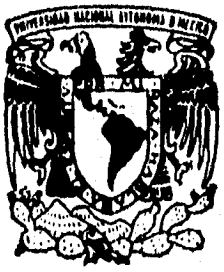


TRABAJO DE TESIS PROFESIONAL

SIMULACION PARA LA SOLUCION DE PROBLEMAS DINAMICOS ESTOCASTICOS Y DE OPTIMIZACION Y CONTROL

Q U E R E P R E S E N T A

GUADALUPE DEL CARMEN RODRIGUEZ MORENO

LICENCIATURA EN MATEMATICAS APLICADAS Y COMPUTACION

1996

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES ACATLÁN

Trabajo de Tesis Profesional:

SIMULACIÓN PARA LA SOLUCIÓN DE PROBLEMAS DINÁMICOS ESTOCÁSTICOS Y DE OPTIMIZACIÓN Y CONTROL

GUADALUPE DEL CARMEN RODRÍGUEZ MORENO

Asesora:

ACT. MARICARMEN GONZÁLEZ VIDEGARAY

LICENCIADO EN MATEMÁTICAS APLICADAS Y COMPUTACIÓN
1996

Por que siempre me han dado lo mejor de ellos
con mucho cariño a mis padres Sofía y Héctor

A mis hermanos Víctor, Mayra y Héctor,
esperando esto sea una forma de inspiración

A la persona que siempre creyó en mí,
mi asesora y jefa: Acc. MariCarmen González Videgaray

A mis compañeros de trabajo: Alejandra, Cuquita, Laura,
pero muy en especial a Sony

A mis profesores y a la UNAM por brindarme
la oportunidad de formarme profesionalmente

Por sus ganas de vivir a mi prima Norma Angélica

A Alex, quien siempre ha estado a mi lado,
tanto en los momentos felices como en aquellos de soledad y tristeza

ÍNDICE

PÁGINA

CAPÍTULO III
TRANSACCIONES Y GPSS/H

CAPÍTULO IV
MODELOS CON UN SOLO SERVIDOR

PÁGINA

INTRODUCCIÓN

En la década de los ochenta, las industrias bajo el proteccionismo del gobierno, no tuvieron la necesidad de competir, pues tenían asegurado el mercado nacional para promover sus productos. Los empresarios no sentían la exigencia de mejorar sus procesos de producción, poseían ventas seguras, sin importar el nivel de calidad que sus artículos contenían, y aunque la legislación vigente marcaba cumplir con ciertas normas de control, el sistema encargado de verificar que se llevaran a cabo no funcionaba adecuadamente, lo que ocasionó que la legislación fuera alejándose de las exigencias que en el mercado internacional se imponen, derivándose así que las exportaciones de nuestro país se limitaran a unos cuantos elementos.

Ante el mal manejo del país, tanto políticamente como económicamente y la creciente demanda de la población por mejores condiciones de vida, durante el sexenio comprendido entre 1988 y 1994, se decide ceder gran parte de las industrias que manejaba el Gobierno a la iniciativa privada y abrir las puertas del mercado mexicano a empresas extranjeras con la firma del Tratado de Libre Comercio con los Estados de Unidos de Norteamérica y Canadá.

Evidentemente las condiciones que se plantearon en dicho acuerdo, dejaban en clara desventaja a las firmas nacionales, dadas las limitaciones que poseían tanto en el nivel técnico como en el económico, y lo que es peor, no se contempló el enorme rezago que se tenía en el nivel administrativo, en donde se observa la apatía y la obstinación de muchos empresarios por seguir con idénticos esquemas y actitudes, al dirigir sus empresas, sin basarse en las nuevas metodologías que se apoyan fuertemente en el proceso de datos de manera electrónica.

Los yerros cometidos recientemente por los responsables de dirigir al país, han dejado aún más claro el hecho de que no podemos seguir con la misma actitud. Es cierto que faltan recursos, que hay más pobreza y una enorme desilusión e incertidumbre en la gran mayoría de la población, sin embargo, no es nada aconsejable tomar una actitud de derrotismo, al estar compitiendo en un mercado libre con empresas extranjeras, tal vez los empresarios ahora se den cuenta de la enorme importancia que tiene innovar, estar al día, en los conocimientos que surgen y decidir cómo aplicarlos a sus actividades, la interrogante que irremediablemente se plantea es: ¿en dónde buscarán estas alternativas de solución y quién las aplicará?

Es necesario entonces realizar, en todos los niveles de la economía nacional, una correcta toma de decisiones, optimizar tiempo y recursos, estimulando al máximo algo que, a veces, parece olvidado por los empresarios y por algunas instituciones de educación superior: la investigación. Esto realmente originaría un vínculo entre las necesidades a nivel industrial del país y lo que se hace en las universidades.

Aún sin existir una estrecha relación, en la vida académica se tiene conocimiento de metodologías adecuadas para auxiliar en la correcta toma de decisiones. Desgraciadamente, pocas empresas saben de la existencia de estos métodos y los que han oído de ellos, desconfían enormemente al no comprender exactamente como funcionan.

Uno de estos métodos es la simulación, la cual es empleada con gran frecuencia fuera de México, dada su confiabilidad en los resultados y su relativa facilidad de uso con la ayuda de las computadoras. Una de las ventajas que presenta es la de brindar una visión futura de un fenómeno de manera muy precisa, hecho que anteriormente parecía imposible dado que sólo era posible experimentar con el sistema real, lo cual acarreaba elevados costos y perdida de tiempo. Es de suma importancia promover su uso correcto y explicar toda la fundamentación teórica que la acompaña a los responsables de la toma de decisiones.

Muchos esquemas manejados en la industria pueden ser programados en una computadora y modificados para analizar ciertos cambios, sin afectar al sistema real, y de esta manera determinar cual es el que mejor se adaptaría a nuestros fines. Es en este momento, al tratar de interpretar la realidad y llevarla a una computadora es donde surge la necesidad de conocer un lenguaje de simulación. Por ello se presenta el GPSS/H, que permite modelar un sistema y modificarlo fácilmente, dado que emplea un lenguaje natural que lo hace más accesible, por su facilidad de comprensión.

Este paquete puede significar una herramienta importante en la administración, es por ello que se considera necesario conocerlo, analizarlo y estudiarlo. El objetivo de esta tesis es: *proporcionar una herramienta para la solución de problemas dinámicos estocásticos, de optimización y control a través de modelos de simulación*, permitiendo así a cualquier persona obtener una visión de como estudiar sistemas a través de los modelos de simulación con el uso del GPSS/H. Sin embargo, es de suma importancia recalcar que este trabajo es tan solo una introducción y que quien trate aprender de él, debe utilizar su creatividad al modelar los sistemas, programando, analizando las diferentes estructuras que se presentan y por último llevando esta experiencia a su campo laboral y verificando la enorme utilidad que puede brindar.

CAPÍTULO I
EL USO DE LA SIMULACIÓN

I.1 INTRODUCCIÓN

El presente capítulo provee algunas definiciones básicas de modelo y sistema, así como el análisis de algunas perspectivas en el área de simulación. Se estudiarán en principio las características generales de la simulación. Posteriormente se verá el campo de aplicación de ésta como una herramienta para investigar las características de los sistemas y se indicará su importancia en la investigación de operaciones. Finalmente se describirán algunas de sus ventajas y desventajas con respecto a otras metodologías.

I.2 SISTEMAS Y MODELOS

I.2.1 DEFINICIÓN DE SISTEMA

Es importante, para el fin del presente trabajo, dar una definición de lo que es un *sistema*, ya que de la correcta interpretación de éste dependerá en gran medida que el uso de la simulación como herramienta, tenga éxito o no. Un sistema se puede definir como un conjunto de elementos o entidades, por ejemplo: máquinas, escuelas, interrelacionados entre sí, que funcionan con un objetivo común.

En la práctica, lo que significa un sistema depende de los objetivos del estudio en particular.

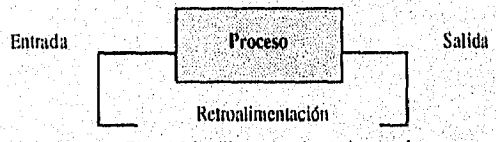


Figura I.2.1 Elementos de un sistema

Un sistema consta de entradas, las cuales se introducen a un proceso y salen como salidas del sistema, estas salidas entonces lo retroalimentan y se convierten en entradas, como se muestra en *Figura I.2.1*. Un estado del sistema se puede definir como un conjunto de variables necesario para describir el sistema en un momento dado, relativos a los objetivos del estudio.

Un sistema, a su vez, puede ser un subconjunto de un sistema compuesto de otra colección de entidades. Por ejemplo, si algún estudioso quiere analizar un banco para determinar el

número de cajeros necesarios, dando así un servicio adecuado a clientes que sólo quieren cambiar cheques o hacer algún depósito, el sistema puede ser definido en base a esa porción del banco, que consiste de cajeros y clientes esperando en filas. Si por otro lado el prestamista y las cajas de depósito son incluidas en este sistema, el estudio abarcaría también esta parte del sistema.

Un proceso de interés dentro del mundo real puede ser en esencia un sistema; para poder estudiarlo científicamente a veces se tiene que formular un conjunto de suposiciones de cómo trabaja dicho sistema (estas suposiciones muy a menudo toman la forma en relaciones matemáticas o lógicas, constituyendo así un modelo, que usualmente es utilizado para entender claramente el comportamiento de un sistema, así como también para explicar, pronostica, controlar, etc.). Un buen sistema debe contener una retroalimentación, es decir, un monitoreo del comportamiento con respecto a ciertos estándares de medición. La mayoría de los sistemas funcionan en un medio ambiente que afecta su conducta.

Los sistemas pueden clasificarse de acuerdo a su construcción en el tiempo en dos tipos, discretos y continuos. Un sistema es discreto si las variables de estado cambian instantáneamente en tiempos separados. Un banco es un ejemplo claro de un sistema discreto, dado que las variables de estado, es decir, los clientes en el banco cambian sólo cuando llega otra persona o cuando un cliente termina de ser atendido o cuando se va. Un sistema es entonces continuo cuando las variables de estado cambian continuamente con respecto al tiempo, un ejemplo de ellos puede ser un avión en el aire, dado que su movimiento es continuo con respecto al espacio. Pocos sistemas en lo general son completamente discretos o completamente continuos, pero tomando en cuenta ciertas características del sistema es posible enmarcarlo como continuos o como discretos.

En algún momento de la vida de la mayoría de los sistemas hay la necesidad de estudiarlos para tratar de comprender las relaciones entre varios componentes o predecir el comportamiento en algunos nuevos escenarios.

I.1.2 Definición de Modelo

Un *modelo* es una representación simplificada de un sistema real con algún objetivo en particular. El objetivo del modelo es permitir al analista la determinación de uno o más cambios del sistema real modelado, que afectan otros aspectos del sistema o inclusive su totalidad. Los modelos siempre deberán ser menos complejos que el sistema real, y si esto no sucede es mejor trabajar con el sistema real.

Un modelo debe contener lo esencial de un sistema real. Aquellos aspectos del sistema que no contribuyen significativamente al comportamiento del sistema no deberán ser incluidos, porque complicarían las relaciones entre las entradas y salidas del modelo; dependiendo del propósito del estudio, el analista deja de incluir realismo de ellos. Los modelos no son verdaderos o falsos, sino más bien son útiles y apropiados para un buen análisis.

Si las relaciones que componen un modelo son lo suficientemente simples, quizá sea posible usar herramientas matemáticas, tales como álgebra, cálculo o probabilidad, para obtener la información exacta de algunas preguntas de interés sobre el sistema; a esto se le llama *solución analítica*. La mayoría de los sistemas reales son demasiado complicados y grandes para permitir construir modelos realistas que puedan ser evaluados analíticamente, y es entonces cuando se deben estudiar bajo la simulación.

Los elementos que constituyen un modelo son los siguientes:

- *Ecuaciones:* están compuestas por variables, parámetros y operadores. Además pueden provenir de alguna definición o pueden estar basadas en la experiencia y el razonamiento, estas ecuaciones están formadas por variables, que representan las distintas partes del sistema y por operadores, que indican la relación que existe entre los elementos.

- *Variables:* relacionan un componente con otro y se clasifican de la siguiente manera:

    - *Exógenas:* son variables independientes o de entrada al modelo; se supone que han sido predeterminadas y proporcionadas, independientemente del sistema que se modela. Existen dos tipos de variables exógenas: *controlables y no controlables*; su clasificación depende de la posibilidad de modificarlas al juicio del analista del sistema o del encargado de tomar las decisiones.
    - *Endógena*: son variables dependientes y de salida del sistema; son generadas por la interacción de las variables exógenas con las de estado, pueden ser variables objetivo y son una función de las variables exógenas.
    - *Determinísticas:* son variables en las que se puede predecir su valor con certeza absoluta.
    - *Aleatorias o estocásticas:* son variables no observables directamente antes de su ocurrencia.
    - *De estado:* describen el estado de un sistema o uno de sus componentes, ya sea al comienzo, al final o durante un periodo de tiempo.

I.1.3 TIPOS DE MODELOS

En la Investigación de Operaciones existen tres clases de modelos:

- Icónicos
- Analógicos
- Simbólicos

Los modelos *icónicos* se parecen físicamente al sistema real, son imágenes a escala del sistema cuyo problema se quiere resolver, como una esfera que modela la tierra o un planetario que se asemeja al sistema solar; algunos otros ejemplos de ellos pueden ser: las fotografías, las maquetas, los dibujos y modelos a escala de barcos, automóviles, aviones y canales.

Los modelos *analógicos* se asemejan al sistema real en su comportamiento y se basan en la representación de las propiedades de un sistema, cuyos problemas se quieren resolver utilizando otro sistema, cuyas propiedades son equivalentes, como un flujo de agua en pipas modela el flujo de electricidad en alambres. Por ejemplo, las propiedades de un sistema hidráulico son equivalentes a los de un sistema eléctrico o inclusive económico.

En los modelos *simbólicos* no hay relaciones físicas o analógicas del sistema real, pero existen relaciones lógicas y matemáticas, que son conceptualizaciones abstractas del problema real a base del uso de letras, números, variables y ecuaciones. Este tipo de modelos son en ocasiones fáciles de manipular y se pueden desarrollar un gran número de experimentos interesantes. Dado que los modelos para simulación deben tener la capacidad para ser implementados en una computadora, deben ser muy explícitos al menos en el nivel lógico.

A continuación se analizarán los modelos simbólicos de acuerdo a sus fines, de arriba a abajo aumenta la dificultad:

- Descriptivos, expresan el comportamiento del fenómeno y dejan el proceso de optimización en manos del analista.
- Explicativos, relacionan el comportamiento causa - efecto.
- De pronóstico, sirven para predecir el comportamiento futuro, utilizando series de tiempo en ellos.
- De optimización, deben tener uno o más objetivos.
- De decisión y control, intentan mantener el fenómeno dentro de cierto límite.

Una clasificación de los modelos simbólicos de acuerdo a su construcción:

- Discretos/Continuos
- Determinísticos/Estocástico
- Lineal/No lineal
- Estáticos/Dinámicos

La clasificación de *modelos discretos* o *continuos* se refiere al tipo de variables del modelo. Las variables continuas pueden tomar el valor de un número real y las variables discretas sólo pueden tomar valores específicos y limitados. Es importante la distribución de las variables de tiempo en el modelo de simulación. Si las alteraciones del modelo ocurren continuamente como el tiempo, entonces se clasifica al modelo continuamente. De otro modo, ocurren en tiempos discretos, y entonces el modelo es discreto. La decisión en usar modelos discretos o continuos para un sistema en particular depende en los objetivos del estudio.

En un proceso químico, la temperatura y la presión experimentan un cambio promedio continuamente en el transcurso del tiempo, entonces un modelo continuo sería apropiado en este caso. En un sistema de colas, como las filas de un supermercado, la gente llega, empieza

a ser atendida y completa el servicio en tiempos discretos, en este caso el modelo es discreto. Debe hacerse notar que los modelos discretos pueden tomar el tiempo como variables continuas. Pero los cambios en el sistema no ocurren continuamente en un promedio dado, sino en eventos de tiempo discretos.

Se distingue entre *modelos determinísticos* y *estocásticos* por el tipo de variable. Si alguna variable aleatoria está presente, se clasifica como modelo estocástico. Las variables aleatorias deberán ser definidas por una función de probabilidad apropiada. Ahora, si las variables toman valores ya dados, y describen el comportamiento del sistema, se clasifica a estos modelos determinísticamente. En todos los sistemas donde hay humanos y máquinas se tiene por lo general variaciones aleatorias.

En los *modelos determinísticos*, ni a las variables exógenas ni a las endógenas, se les permite ser variables al azar, ya que se supone poseen relaciones exactas dentro de las características de operación, en lugar de funciones de densidad de probabilidad. Los modelos determinísticos requieren menos procesamiento en computadoras que los modelos estocásticos y con frecuencia es posible resolverlos analíticamente por medio de la utilización de técnicas como el cálculo de máximos y mínimos. Los resultados en estos modelos son determinados en función de las entradas cuantitativas y su relación en el modelo, sin embargo, en muchos casos toma mucho tiempo de computadora evaluar la solución.

Muchos sistemas, sin embargo, deberán ser modelados teniendo al menos algún componente de entrada aleatorio, es entonces cuando surgen los *modelos estocásticos*. Estos son modelos en los que por lo menos una de las características de operación está dada por una función de probabilidad. Muchos sistemas de inventarios y de colas son modelados estocásticamente. Estos modelos producen datos de salida aleatorios y deben ser tratados como una estimación de las verdaderas características del modelo.

Los modelos son clasificados como *estáticos o dinámicos* dependiendo del cambio de las variables en el tiempo. Los *modelos estáticos* son aquellos que no toman en cuenta, explícitamente, a la variable tiempo, son una representación de un sistema en un momento en particular o puede ser una representación en donde la variable tiempo no juega un papel importante, ejemplos de ellos son los modelos de Monte Carlo. Los modelos matemáticos que tratan de las interacciones que varían con el tiempo, se denominan *modelos dinámicos*, representan un sistema en el transcurso del tiempo, ejemplo de ello puede ser una línea de espera.

I.3 CARACTERÍSTICAS GENERALES DE LA SIMULACIÓN

Simulación es una palabra que a mucha gente le es familiar, al menos en forma general. Un diccionario menciona que la simulación tiene que ver con la imitación o con mimetismo, *dando la apariencia o efecto de*, o *tomando las características de la realidad* además de que en la simulación hay objetos fingidos, replicas o modelos. Para llevar un análisis de

simulación exitoso, no tan solo es necesario analizar *la apariencia o característica*, sino más bien la operación o comportamiento del modelo de simulación. El uso moderno de esta palabra se remonta a 1940, cuando Von Neumann y Ulam acuñaron el término *análisis de Monte Carlo* para aplicarlo a una técnica matemática que se usaba en aquel entonces para resolver ciertos problemas de protección nuclear que eran demasiado costosos para resolverse experimentalmente o demasiado complicados para ser tratados analíticamente. El análisis de Monte Carlo involucraba la solución de un problema matemático no probabilístico, mediante la simulación de un proceso estocástico, cuyos momentos o distribuciones de probabilidad satisfacen las relaciones matemáticas del problema no probabilístico. Pero se puede ver una definición formal:

"x simula a y" si y solo si :
a) x y y son sistemas formales
b) y se considera como el sistema real
c) x se toma como una aproximación del sistema real
d) las reglas de validez de x no están exentas de error.

Raúl Coss Bu, autor del libro: *Simulación, Un enfoque Práctico*, menciona algunas de las definiciones de simulación más claras y prácticas, dadas por diferentes autores:

- Según Thomas Naylor, la *simulación es una técnica numérica para conducir experimentos en una computadora digital. Estos experimentos comprenden ciertos tipos de relaciones matemáticas y lógicas, las cuales son necesarias para describir el comportamiento y la estructura de sistemas complejos del mundo real a través de largos periodos de tiempo.*

- Maisel y G. Gnugnoli la define como *una técnica numérica para realizar experimentos en una computadora digital. Estos experimentos involucran ciertos tipos de modelos matemáticos y lógicos que describen el comportamiento de sistemas de negocios, económicos, sociales, biológicos, físicos o químicos, a través de largos periodos de tiempo.*

- Robert E. Shanon la define como *el proceso de diseñar y desarrollar un modelo computarizado de un sistema o proceso y conducir experimentos con este modelo con el propósito de entender el comportamiento del sistema o evaluar varias estrategias con las cuales se puede operar el sistema.*

- Según Kelton y Law *la simulación es una técnica que se puede implementar en la computadora para imitar las operaciones de varios tipos de procesos del mundo real.*

La habilidad para desarrollar un modelo es fundamental para llevar a cabo un buen análisis de simulación. Además es importante la necesidad de expresar el modelo como un programa computacional eficiente y efectivo. También, dado que los modelos de simulación son descriptivos en naturaleza, debe uno definir los objetivos de la simulación correctamente.

utilizando un procedimiento de experimentación bien ideado para obtener resultados útiles y lógicos. El propósito fundamental de un análisis de simulación es predecir el comportamiento del sistema, de esta forma se ayuda a quienes toman las decisiones a evaluar todas las posibles alternativas.

La simulación requiere de un conjunto específico de aptitudes, entre ellas se encuentran, el modelaje, la programación, la probabilidad, la estadística y los métodos recursivos, muchos de éstos temas los proporciona la carrera de Matemáticas Aplicadas y Computación en la Escuela Nacional de Estudios Profesionales, Acatlán. El área de modelaje es quizá la menos definida de este conjunto de habilidades y es quizá la más importante para llevar a cabo una simulación con éxito.

La simulación en esencia es una técnica que permite construir modelos de situaciones reales, aunadas a la realización de experimentos con estos modelos. Esta definición es sumamente amplia y puede muy bien comprender situaciones aparentemente no relacionadas entre sí. Proporciona una forma de dividir el trabajo de construir un modelo en componentes, es decir, siendo más sencillo formular (por ejemplo, un componente puede ser un sistema de líneas de espera sencillo) y después combinar estos componentes en forma natural.

Cuando un arquitecto construye un modelo en tres dimensiones, mostrando algún diseño de un edificio, el modelo es la imitación de la construcción. Esto es mucho más fácil, rápido y mucho menos caro que construir el edificio físicamente, y es mucho menos costoso modificarlo con el único propósito de cambiar el diseño, el prototipo da al arquitecto la compresión de la estructura, de la apariencia, así como de la factibilidad del trabajo. Estos mismos no estarían disponibles en un dibujo de dos dimensiones, (esto también es simulación en su propia forma) o si se intentara visualizar el diseño mentalmente. El modelo ayuda al arquitecto y al cliente a lograr un mayor entendimiento del diseño propuesto, también sirve como base para la comunicación entre el arquitecto y el cliente, permitiendo el entendimiento de las intenciones de los dos, para que al final llegar a un acuerdo en el diseño final.

El modelo del arquitecto es un ejemplo de una simulación en forma de tres dimensiones, siendo éste estático. El modelo esta inmóvil en el sentido de que la gente no se mueve por el edificio, usa los elevadores o el sistema de aire acondicionado.

Un ejemplo de un modelo de simulación físico, de tres dimensiones y dinámico es el siguiente caso: un fabricante de elevadores tiene un modelo físico de un sistema general de elevadores, el usuario del modelo decide cuántos pisos existen en el edificio y cuántos elevadores debe contener el sistema, dispone del porcentaje de la gente que sube o baja en el ascensor, indica a que piso la gente desea ir, selecciona las reglas de los movimientos (tales como especificar que los elevadores pararán en el piso 20 o en pisos más altos) y luego pone en operación el modelo, observando los grupos de personas que esperan al elevador (representadas por luces y mostradores) en los diferentes pisos, viendo cómo los elevadores recogen y dejan a la gente, luego se obtienen estadísticas, indicando el tiempo promedio de espera, y el tiempo promedio requerido para que la gente llegue a su destino. Cambiando los

puntos de arribo y los porcentajes de llegadas de la gente, el número de elevadores y las reglas de los movimientos, el usuario puede experimentar informalmente con el modelo en una forma *de que pasaría si*, buscando un diseño que equilibre el funcionamiento con el costo.

En el modelo del elevador, no sólo están incluidos los aspectos físicos, sino también los aspectos lógicos. El experimentador provee información del modelo, lógicamente y computacionalmente, para así imitar la operación del sistema. Se puede notar que el modelo no sólo imita la operación del sistema del ascensor sino que también refleja el transcurso del tiempo. En el prototipo, el tiempo simulado puede ser mucho más rápido que en el sistema real. El efecto de reducción del tiempo sirve para movilizar la porción del experimento, lo anterior causa que la operación del modelo sea mucho más eficiente para el usuario.

Algunas simulaciones dinámicas se comportan determinísticamente, otras más contienen elementos probabilísticos. Por ejemplo, en el modelo del sistema del elevador, el tiempo requerido para subir en un ascensor un piso puede ser de un valor determinístico, tal como un segundo; pero por otro lado, cuando un elevador se detiene en algún piso, es algo aleatorio el determinar el número de personas que estarán esperándolo en ese instante.

Gran parte de las simulaciones, especialmente de diversión y de educación, están construidas en base a las computadoras personales. Estas simulaciones basadas en computadoras no son físicas en el sentido de un modelo arquitectónico de tres dimensiones pero pueden ser físicas en el sentido que los aspectos de la simulación son representados gráficamente en una pantalla de dos dimensiones y, como se puede ver en el modelo dinámico del ascensor, puede tomar aspectos lógicos y computacionales, con algunas entradas del usuario. Estos pueden ir más lejos que esto, desafiando la interacción con el usuario, un ejemplo de ellos son los simuladores de vuelo, juegos militares, juegos de gerencia, modelos físicos de ríos, modelos econométricos, diversos dispositivos eléctricos analógicos y pruebas de aeroplanos en túneles aerodinámicos.

Una habilidad importante para llevar a cabo una simulación con éxito, es la necesidad de expresar un modelo en forma útil, para manipularlo con el fin de resolver lo relevante al diseño y operación del sistema. El programa en computadora traduce el modelo en una forma útil, mientras que la teoría de probabilidad define las variables aleatorias, la teoría de estadística asiste en el diseño y análisis de experimentos usados para contestar a las interrogantes planteadas. Sin embargo, ni la estadística, ni el modelo de simulación, son útiles para obtener soluciones óptimas, por lo que el recurso final podría ser el empleo de los métodos iterativos para este fin.

I.4 Campo de Aplicación de la Simulación

Así como una persona puede utilizar la simulación para analizar estrategias de guerra o para diseñar un juego de cualquier tipo, y luego experimentar las características del juego, puede también aplicarse para investigar las características de un diseño o estrategias propuestas

para un alcance amplio de situaciones de una gran importancia práctica. El uso de la simulación no esta restringido a algún tipo de problema, sino más bien la simulación es una herramienta versátil, que tiene una enorme aplicación.

Como se muestra en la *Tabla 1.4.1*, en los Estados Unidos la simulación ha sido aplicada en los sectores de producción, de servicio y público, incluyendo: planeación corporativa, sistemas de control de inventario, líneas de producción, sistemas de tratamiento de materiales y distribución de periodos de trabajo. También se ha aplicado en sistemas de admisión de hospitales, sistemas de transportes y sistemas de elevadores, entre otros.

Se pueden definir las aplicaciones en dos dimensiones:

- Área funcional
- Producción
- Servicio

Algunas áreas funcionales podrían ser: análisis de inventarios, sistemas de distribución de horarios de trabajo, sistemas complejos de líneas, planeación corporativa, sistema de tratamiento de materiales complejos y juegos de guerra o negocios. En el área de producción y servicio pueden ser aplicadas en la manufacturación, cuidado de la salud, justicia criminal y administración publica y educación.

Social	
• Localizaciones de las respuestas de los vehículos en emergencia	• Planificación de la educación
• Diseño de sistema de tránsito masivo	• Estrategias de campañas políticas
• Secuencia de las luces de tráfico	• Control del aire contaminante
• Evaluación para diseño de organizaciones de servicio tales como hospitales, correos o restaurantes de comida rápida	• Diseño y operación de facilidades de transportación tales como carreteras, aeropuertos, metros o puertos
Salubridad	
• Estrategias de control de epidemias	• Diseño del cuarto de emergencia
• Estrategias para transplante de órganos	• Planeación del seguro social
• Admisiones al hospital	
Militar y Espacio Aéreo	
• Estrategias de guerra	• Distribución del equipo
• Estrategias de busca y salvamento	• Sistema de defensa aérea
• Posición de satélites	
Servicio Industrial	
• Administración del seguro y riesgo	• Localizaciones de facilidades
• Estrategias de auditoria	• Horarios de cajeros
• Diseño de redes de comunicación	• Horarios de fletes
Industria	• Diseño y análisis de sistemas manufactureros
• Control de Inventarios	• Pruebas de seguridad de producto
• Negociaciones de labores	• Control de calidad
• Horarios de producción	• Diseño de canales de producción
• Análisis de sistemas financieros o económicos	

Tabla 1.4.1 Situaciones donde la simulación ha sido utilizada con mayor frecuencia en los Estados Unidos

Cabe mencionar que en el país, la simulación no ha sido utilizada con mucha frecuencia aún, pero con el Tratado de Libre Comercio el uso de esta debe cambiar y extenderse más en áreas como las mencionadas anteriormente. Es muy probable que con los problemas económicos, muchos empresarios y el mismo gobierno estén menos dispuestos a *experimentar* con *nuevas técnicas*, pero es necesario crear un ambiente de confianza y seguridad en cuanto al uso de esta técnica, dadas las ventajas que ofrece para la toma de decisiones.

I.5 Importancia de la Simulación en la Investigación de Operaciones

I.5.1 ¿Qué es la Investigación de Operaciones?

Se han dado muchas definiciones de lo que es la investigación de operaciones (I.O.) o investigación operativa como suele llamarse en los países europeos. La I.O. puede describirse como un enfoque científico de la toma de decisiones que requiere la operación de sistemas organizacionales. Sin embargo, con objeto de establecer una base para que se pueda entender la naturaleza de la I.O., se usará la definición de Churchman, Ackoff y Arnoff, la cual dice lo siguiente:

"La I.O. es la aplicación, por grupos interdisciplinarios, del método científico de problemas relacionados con el control de las organizaciones o sistemas (hombre-máquina) a fin de que se produzcan soluciones que mejor sirvan a los objetivos de toda la organización"[1]

Básicamente la I.O. es un método que permite encontrar las relaciones óptimas que mejor operen en un sistema dado y con un objetivo específico. También se puede analizar como la aplicación de la metodología científica a través de modelos, primero para representar el problema real que se quiere solucionar en un sistema y segundo para resolverlo.

La I.O. se usa ampliamente en todo tipo de organizaciones incluyendo la industria, el comercio, la aérea, de proyectiles, automotriz, comunicaciones, computación, energía eléctrica, electrónica, alimenticia, metalúrgica, minera, del papel, del petróleo y del transporte. Las instituciones financieras, gubernamentales y de salud están utilizando también la I.O.

La I.O. se aplica a problemas que se refieren a la conducción y coordinación de operaciones o actividades dentro de una organización. La investigación de operaciones incluye la investigación científica creativa de las propiedades fundamentales de las operaciones. Se ocupa también de la administración práctica de la organización.

---

[1] Churchman C. W., Achoff, R. L. y Arnoff, E. L., *Introduction to Operation Research*, John Wiley and Sons.

Para concluir se puede decir que la I.O. se ocupa de la toma decisiones óptima y del modelado de sistemas determinísticos y probabilísticos que se originan en la vida real.

Algunas de las técnicas de la investigación de operaciones son las siguientes:

- Estadística
- Programación Lineal
- Simulación
- Programación Matemática
- Probabilidad
- Procesos Estocásticos
- Análisis de Redes PERT
- Optimización
- Programación Dinámica
- Teoría de Decisiones
- Teoría de Colas
- Control de Inventarios
- Teoría de la Confiabilidad

Estas técnicas tienen por objetivo, usando métodos científicos, ayudar a resolver problemas complejos de decisión, en lugares organizados.

I.5.2 La Simulación frente a las demás Técnicas de la Investigación de Operaciones

La simulación es tan solo parte de una de las técnicas usadas en la investigación de operaciones y ciencias de la administración. Tiene un alto renombre entre las demás técnicas en términos de importancia. Este renombre ha crecido en base a los numerosos estudios desde mediados de los 60. Las tres técnicas que sobresalen por su importancia son *la estadística, la programación lineal y la simulación.*

A continuación se mencionan algunos estudios sobre la importancia de la simulación entre las técnicas de investigación de operaciones y el perfil de los que la practican en las empresas de Estados Unidos.

1. En 1972, Turban[2] hizo un informe sobre un estudio de las actividades de investigación de operaciones que proporcionó un panorama de dichas actividades durante 1969. Se mandaron cuestionarios por correo a los directores de investigación de operaciones/ciencias de la administración de 475 compañías. Estas se seleccionaron entre las 500 más importantes de la lista del Fortune, usando las 300 corporaciones industriales más grandes, 50 corporaciones dentro del rango entre esas 300 y las 500, y las 25 compañías más grandes de cada una de las categorías de servicios, bancos, servicios públicos (luz, gas, energía eléctrica), comercializadoras, aseguradoras y de transporte. Regresaron 107 cuestionarios. Se encontró que la simulación en niveles de corporación era usada aproximadamente en un 25%.

[2] Turban, E., *A Sample Survey of Operations Research Activities at the Corporate Level,* Operations Research, 1972.

2. En 1977, Ledbertter y Cox[3] publicaron los resultados del estudio de las 500 empresas del *Fortune,* sobre la utilización de técnicas de investigación de operaciones en sus empresas. Hubo 176 respuestas.

3. En 1978, en una encuesta de graduados del Departamento de Investigación de Operaciones en la Universidad Case Western Reserve, Rasmussen y George encontraron que entre los graduados de maestrías, la simulación estaba situada en el quinto lugar entre las 15 áreas en términos de su valor después de su graduación (atrás de lo que ellos llaman métodos estadísticos, pronósticos, análisis de sistemas y sistemas de información). Entre los doctorados, la simulación esta empatado con programación lineal, y en segundo lugar quedaron los métodos estadísticos.

4. En 1979, Thomas y DaCosta[4] informaron sobre un estudio de las actividades de investigación de operaciones, que incluyó 260 empresas de la lista citada anteriormente, preguntando cual de las 14 técnicas eran las más usadas, se recibieron 150 cuestionarios contestados y la simulación obtuvo el segundo lugar con el 84% de uso.

5. En 1980, Shannon Long y Buckles encuestaron a los miembros de la División de Investigación de Operaciones del Instituto Americano de Ingenieros Industriales (Operation Reseach Division of the American Institute of Industrial Engineers), y reportaron que la simulación quedo en segundo lugar en familiaridad (sólo detrás de programación lineal) y en primero en términos de utilidad y de interés entre 12 metodologías.

6. En 1983, Forgionne[5] publicó los resultados de un estudio sobre la utilización de investigación de operaciones/ciencias de la administración en las organizaciones. Enviando un cuestionario por correo de una muestra aleatoria de 500 empresas tomadas de las 1,500 corporaciones más grandes de Estados Unidos, recibiendo 125 respuestas.

7. En 1989, Harpell Lane y Mansoner reportaron que la simulación estaba en segundo lugar de utilización detrás del análisis estadístico entre ocho herramientas en una encuesta a empresas de renombre.

Todas estas encuestas pueden parecer bastante antiguas, pero reflejan el uso y valor de la simulación en la sociedad norteamericana y se nota un incremento considerable por las mejorías en la computadora y en el software. En la *Tabla I.5.3.1*, se muestra un resumen de lo analizado anteriormente.

[3] Ledbetter, W. N. y J. F. Cox *Are OR Techniques Being Used,* Insutrial Engineering, febrero 1977.
[4] Thomas, G. y J. DaCosta, *A Sample Survey of Corporate Operations Research,* Interfaces, 1979.
[5] Forgionne, G. A., *Corporate Management Science Activities: An Update,* Interfaces, 1983.

Técnica	Turban (1969)	Ledbetter y Cox (1975)	Thomas y DaCosta (1977)	Forgionne (1982)
Análisis de Decisión de Bayes	--	--	9	--
Delphi	--	--	13.5	--
Programación Dinámica	6	6	10	7
Métodos Financieros	--	--	13.5	--
Teoría de Juegos	--	7	--	8
Programación Heurística	8.5	--	8	--
Programación Entera y Mixta	--	--	12	--
Teoría de Inventarios	4	--	5	--
Programación Lineal	3	2	3	4
Modelos de Redes	--	4	--	--
Programación No Lineal	7	--	7	6
PERT/CPM	5	--	4	3
Análisis de Riesgo	--	--	11	--
Líneas de Espera	8.5	5	6	5
Simulación	2	3	1	2
Análisis Estadístico	1	1	1	1

Tabla 1.5.3.1 La jerarquía de las técnicas de investigación de operaciones de la administración, la jerarquía 1 denota la técnica que se usa con más frecuencia

Dado su alto rango entre las demás alternativas de la investigación de operaciones y la frecuencia de su uso, para ayudar a desarrollar soluciones efectivas y ahorrativas a problemas reales, se puede afirmar que el campo de la simulación se torna cada vez más vigoroso, dinámico y creciente.

En un estudio hecho con académicos y empresarios se les pidió que calificaran algunas técnicas de acuerdo a su familiaridad y de acuerdo a su deseo de aprender dicha técnica. Entre un grupo de 12 alternativas, la simulación quedó en segundo lugar en familiaridad y en primero por deseo de aprenderla, como se puede ver en la *Tabla 1.5.3.2*.

Técnica	Familiaridad	Utilidad	Deseo de aprender
Programación Lineal	1	2	8
Simulación	2	1	1
Análisis de Redes	3	4	2
Teoría de Colas	4	7	10
Arboles de Decisión	5	3	3
Programación Entera	6	6	6
Programación Dinámica	7	11	7
Programación No Lineal	8	9	4
Procesos de Markov	9	10	11
Análisis de Reemplazo	10	5	9
Teoría de Juegos	11	12	12
Programación Objetiva	12	8	5

Tabla 1.5.3.2 Resultados del estudio hecho a académicos y empresarios norteamericanos

Al mostrar estas estadísticas se ha tratado de señalar que el uso de la simulación a través de las últimas décadas ha sido de gran importancia, entre otras cosas por su fácil aplicación en procesos de todo tipo.

I.6 LA SIMULACIÓN VS. LAS OTRAS ALTERNATIVAS

Al tener un sistema y tratar de resolverlo se tiene dos alternativas, como se muestra en la *Figura I.6.1*:

- Experimentar con el sistema real
- Experimentar con un modelo del sistema

Al experimentar con un modelo del sistema, se puede experimentar con:

- Modelo Matemático
- Modelo Físico

Para darle solución al modelo matemático se tienen las siguiente opciones:

- Solución Analítica
- Simulación

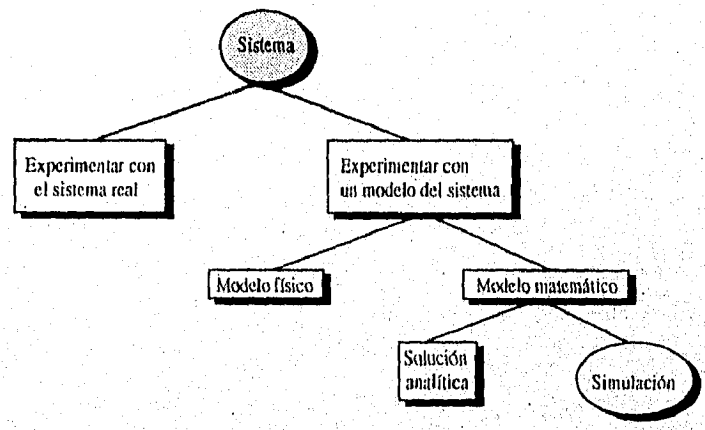


Fig. I.6.1 Diferentes formas de estudiar un sistema y sus formas de ser evaluado

Experimentar con el sistema real vs. Experimentar con el modelo del sistema. Si es posible (y de costo permisible) se puede alterar el sistema físicamente y después dejarlo operar bajo las nuevas condiciones. Pero rara vez es posible hacer esto, por que tal experimento sería costoso o muy desgastaste para el sistema. Por ejemplo, un banco puede contemplar el reducir el número de cajeros para disminuir el costo, pero tratar de hacer esto llevaría a una espera de clientes demasiado grande y quizá a una perdida de ellos. Si un sistema no existe y se desea estudiarlo para analizar su forma de creación, es conveniente experimentar con el modelo del sistema, ejemplos de esta situación podrían ser facilidades flexibles de manufacturación o sistemas de armas nucleares. Por estas razones, es necesario crear un modelo como una representación del sistema y estudiarlo como sustituto de lo que

sucede en la realidad, sin embargo, cuando sea usado, siempre surgirán dudas sobre si realmente el modelo refleja el sistema para propósitos de decisiones a ser tomadas.

Modelo físico vs. Modelo matemático. Para la mayoría de las personas, la palabra modelo trae a la mente imágenes de maquetas, de coches de barro en túneles, cabinas de avión usadas para entrenamiento o cosas así. Ocasionalmente, es de utilidad construir prototipos físicos para el estudio de sistemas de ingeniería o administrativos. La gran mayoría de estos son construidos con propósitos matemáticos, representando el sistema en términos lógicos y relaciones cuantitativas que son manipulados para analizar como el modelo responde y además de como el sistema reaccionaría, suponiendo que el modelo fuera valido. Quizá el ejemplo más sencillo de un modelo matemático es la relación $d=rt$ donde r es la velocidad, t es el tiempo y d es la distancia viajada. Esto da un modelo válido en cierta circunstancia pero en otras puede ser poco útil.

Solución analítica vs. Simulación. Una vez creado el modelo matemático, debe examinarse como puede ser usado para resolver las dudas acerca del sistema que debe representar. Si el modelo es lo suficientemente simple, quizá sea posible trabajar con sus relaciones y valores para obtener una solución exacta y analítica. En la ecuación $d=rt$, por ejemplo, si se sabe la distancia para viajar y la velocidad, se puede trabajar con la siguiente ecuación $t=d/r$ como el tiempo requerido que se puede obtener fácilmente, pero algunas soluciones analíticas pueden ser muy complejas requiriendo bastante recursos computacionales; invertir una matriz de 10x10 es un ejemplo bien sabido de una situación en donde hay una formula analítica conocida, pero para obtenerla numéricamente en un momento dado la situación se puede tornar bastante compleja.

Si la solución analítica de un modelo matemático está a la disposición del usuario y es eficiente computacionalmente, usualmente es mejor estudiar el modelo de esta manera y no con simulación. Sin embargo, muchos sistemas son demasiado complejos y sin ninguna posibilidad de solución analítica. En este caso, los modelos deberán ser estudiados con la simulación. Usar el método de simulación como último recurso puede parecer trivial, pero existen muchas situaciones donde se requiere de la simulación por la complejidad de los sistemas que representan en forma valida.

Figura 1.6.2 Conjunto de tres alternativas de aproximación que puede ser usadas para la resolución de problemas o más específico en el diseño y análisis de sistemas

En el extremo izquierdo de la *Figura 1.6.2* se encuentra la idea de experimentar con el sistema real. En el extremo derecho el del modelaje matemático. En medio de estos dos extremos se encuentra ubicada la simulación.

Las alternativas de la figura anterior pueden ser ilustradas con un ejemplo simple, pero instructivo, descrito en el contexto de flechas y arcos. Se supone que 10 flechas deben ser tiradas al blanco y que cada flecha da o no al blanco con una probabilidad de 0.5. ¿Cuál es entonces, la probabilidad que un número par (tal como el 0, 2, 4, ...) de tiradas al blanco alcance el objetivo? Una forma para estimar el resultado es experimentar con el sistema real. Esto requiere de la obtención de un tirador, flechas y el blanco, también se debe establecer condiciones, tales como que en cada lanzamiento se tiene una posibilidad de dos, de darle al blanco o no darle. Se tendría que lanzar al blanco 10 veces, una por una y apuntar si un número par de tiros al blanco ocurrieron, además de repetir el experimento varias veces y observar el número de veces que un número par de disparos ocurrió. La fracción de los experimentos en los que se obtuvo un número par de tiros puede ser usada para una estimación de la solución a la pregunta propuesta originalmente.

Otra forma para resolver esta pregunta es darle solución por medio del modelado matemático, aplicando algunos conceptos fundamentales de la probabilidad. Sea p la probabilidad que una flecha tire al blanco, y sea P la probabilidad que un número par de tiros al blanco ocurra cuando 10 flechas son tiradas, entonces P y p están relacionadas por la siguiente ecuación:

$$P = \sum_{k=0}^{5} \binom{10}{2k} p^{2k} (1-p)^{10-2k}$$

Esta ecuación puede ser evaluada para $p = 0.5$, dando como resultado la solución exacta a la pregunta, generalmente a esta ecuación se le llama la Binomial.

La simulación da un segundo método que puede ser usado para solucionar la pregunta propuesta anteriormente. La simulación no tiene que ver con la experimentación del sistema real, ni con un modelo matemático, tiene que ver con la experimentación de una alternativa del sistema real.

Si no se desea o puede tirar al blanco, el analista podría fácilmente imitar el proceso en diferentes formas; lo anterior requiere seguir un esquema para interpretar un evento aleatorio de cierto tipo que corresponda a dar al blanco o fallar el lanzamiento. Se podría tirar al aire una moneda, por ejemplo, interpretando sol como dar al blanco y águila como un fallo. Lanzando la moneda repetidamente y tomando nota de los resultados, se podría entonces simular cuantas veces se desee el proceso de tirar 10 flechas al blanco.

En una aproximación con la simulación, se puede reemplazar el tiro al blanco con el lanzamiento de una moneda. Lanzar una moneda puede ser tedioso y puede ser un consumo inútil de tiempo. Para evitar esto, se podría hacer una simulación basada en computadora, pero hay que hacer notar que las computadoras siempre se comportan de manera predecible. De aquí surge la siguiente pregunta: ¿Cómo un resultado impredecible de lanzar flechas al blanco puede ser incorporado a la simulación vía computadora? Brevemente, esto puede ser hecho con el uso de un esquema basado en computadora para generar números aleatorios.

Se supone, por ejemplo, que existe un esquema hecho en computadora para sacar un número aleatorio del conjunto de todos números decimales de 16 dígitos uniformemente distribuidos en un intervalo abierto de 0.0 a 1.0. Para simular el tiro al blanco, una computadora puede generar números aleatorios de este intervalo, si el número aleatorio es menor o igual a 0.5, entonces esto puede ser interpretado como que se tiró al blanco con éxito; de la otra forma, la interpretación sería que se falló al blanco. De aquí, es un paso fácil para los que saben programar el escribir un programa que simule cuantas veces se quiera el proceso de tirar al blanco en un conjunto de 10 flechas. Entonces, así como se experimenta con el sistema real, la fracción de los resultados de los experimentos en el caso de que un número par de veces tire al blanco, puede ser usado para proporcionar una estimación para el resultado a la pregunta propuesta originalmente. Hay que hacer notar que una simulación basada en computadora es mucho más rápida que lanzar una moneda o que tirar al blanco.

Este ejemplo simple ilustra la naturaleza de la diferencia entre experimentar con el sistema real en un extremo y el modelaje matemático en el otro, siendo la simulación la que ocupa la posición intermedia entre estos dos extremos. Problemas de verdadero interés son mucho más complicados y serios que tirar al blanco, muy a menudo no tienen soluciones matemáticas, usualmente el sistema real no existe o no puede ser usado para experimentación, entonces la simulación puede ser la única forma de enfrentar el problema.

I.7 Ventajas y Desventajas de la Simulación

La simulación ha sido utilizada ampliamente por el grado de realismo que puede ser incluido en los modelos de simulación y dada la facilidad con que se estructuran, además pueden ser explicados a personas que no saben del tema y deben tomar algunas decisiones. Comparando la simulación con técnicas analíticas como la teoría de colas, los modelos de simulación no requieren de suposiciones tan rigurosas y necesarias para una solución tratable. También, las relaciones lógicas y matemáticas asociadas a estos diseños son generalmente más obvias. Pero a pesar de todas sus virtudes los modelos de simulación son menos eficientes que los modelos analíticos. Esto deja a los expertos usar la simulación como una técnica de último recurso.

Los modelos analíticos tienen varias ventajas al ser considerados con cuidado cuando se trata de seleccionar alguna técnica para tratar de resolver algún problema. Estas ventajas incluyen que la descripción del problema es muy precisa, soluciones en forma cerrada, facilidad al evaluar los impactos de cambios de entradas a las salidas y en algunos casos, la habilidad para producir un solución óptima.

Hay algunas desventajas como la suposición de que un sistema discreto puede no ser realista y además las formulaciones matemáticas complejas hacen más difícil darle solución al problema. Por ejemplo, muchos sistemas pueden ser modelados como redes de colas, pero las suposiciones requeridas para una solución analítica puede dar una solución poco realista, o quizá la formulación matemática necesaria para reflejar el grado de veracidad no es

posible. Los modelos de simulación compensan estas desventajas sin sacrificar algunas de las ventajas de los modelos analíticos.

Como se indicó en la *Figura 1.6.2*, el realismo fue obtenido experimentando con el sistema real, esto es la mayor ventaja de esta aproximación pero tiene numerosas desventajas, entre éstas se encuentra que el sistema real debe existir antes que algún experimento pueda ser hecho, pero quizá el objetivo es diseñar un sistema que no existe todavía. Puede suceder que si el sistema existe y está en uso, entonces por razones económicas y/o políticas no será posible interrumpir su actividad tan solo con el propósito de observar que sucede si se alterá. Por ejemplo, si un sistema manufacturero es usado para la construcción de productos, entonces puede ser demasiado costoso interrumpir el proceso de producción para investigar los efectos de hacer uno o más cambios al sistema. Aún si el sistema real puede ser usado para experimentación, será necesario mucho tiempo para llevarla a cabo. En un sistema existente cuando un cambio ha sido aplicado, deberá operar por días o semanas mientras es observado, entonces quizá se presenten una o varias alternativas a ser analizadas, hasta entonces los resultados del experimento no podrán estar disponibles en el tiempo requerido.

Además, las pruebas con el sistema real no producen las respuesta exactas a las preguntas planteadas, tan solo arrojan estimaciones, como se ilustró en el ejemplo de las flechas.

No hay una generalidad en los resultados observados en la experimentación con el sistema real. El resultado sólo se aplica al sistema en el estado en donde la prueba fue hecha. Las relaciones explícitas de causa-efecto no son derivadas del ensayo, y es muy difícil y a veces imposible generalizar los resultados. No obstante, algunos sistemas son tan inherentemente complejos que se debe recurrir a la experimentación directa, al menos eventualmente. Por ejemplo, para determinar la efectividad de una droga para combatir el cáncer propuesta en el sistema biológico humano, el uso directo del fármaco en las personas puede ser necesario.

Ahora se considerarán los pros y las contras del modelaje matemático. Como se indicó en la *Figura 1.6.2*, el modelaje matemático implica un alto grado de abstracción. Los modelos matemáticos de los sistemas dan como resultado el uso de ecuaciones para ser expresadas como relaciones cuantitativas y lógicas entre los elementos de los que un sistema consta. El modelaje matemático cae en las categorías de programación lineal, probabilidad, procesos estocásticos, teorías de colas y control de inventarios. Este tipo de recurso se beneficia de la generalidad, elegancia y poder de las matemáticas. La mayor ventaja de los modelos matemáticos es que están sujetos a suposiciones expresadas en ecuaciones de forma exacta, en donde los valores de las variables dependientes de interés pueden variar con valores alternativos con respecto a las variables de decisión. Esto es evidente en la ecuación del ejemplo de las flechas. Si la probabilidad de dar al tiro es cambiada de 0.5 a 0.2, debido a la mala puntería del ejecutante, tan solo sería necesario sustituir los valores correspondientes en la ecuación y calcular la respuesta exacta a la pregunta, de la probabilidad de dar al blanco en un número par de veces.

Los modelos matemáticos también pueden ser usados para determinar si la solución óptima de un problema existe y si es así para determinar si es única dicha solución. En el ejemplo de las flechas y el tiro al blanco se puede suponer que se puede saber el valor de p (la probabilidad que un tiro tiene de dar en el blanco) maximice el valor de P (la probabilidad que un número par de veces de en el blanco con éxito). En la aproximación matemática, la ecuación que relaciona P con p puede ser usada para responder a esta pregunta exactamente. Como contraste, hay que considerar que la experimentación con el sistema real para dar una respuesta aproximada a los objetivos del estudio puede ser tedioso además de consumir mucho tiempo.

La mayor desventaja de los modelos y métodos matemáticos en que se basan es que una o más características importantes de la situación deberán ser modificadas o ignoradas, si es que se desea dar la solución matemática al problema; al grado que la suposiciones hagan que el modelo matemático creado falle al ajustarse a las condiciones del problema, el resultado del modelo puede no dar solución al conflicto original. En este sentido, la solución correcta a un problema incorrecto es la solución incorrecta al mismo. Esta desventaja permite explicar por que la simulación es usada tan frecuentemente como alternativa del modelaje matemático.

Otra desventaja potencial del modelaje matemático es que se requiere de un alto nivel de conocimiento de dicha área. Aún si el protolipo existe para un problema dado, la persona a cargo de enfrentar la problemática puede no saber de la existencia o puede no entender los detalles del modelo a tal grado que le sea posible aplicarlo. Y aún si un modelo matemático es válido, darle solución puede estar fuera de su alcance. Resultaría casi imposible o muy costoso realizar experimentos de validación, en la estructura matemática que describe al sistema.

Ahora se considerará a la simulación, como un experimento en donde se trata de entender, como un problema se comporta en la realidad, con la imitación de su comportamiento en un ambiente artificial que se aproxime lo más posible a la realidad. A continuación se muestran algunas de las ventajas importantes de la simulación:

1. Realismo. Los modelos de simulación pueden ser realistas, en el sentido de que captan las características actuales del sistema. La simulación basada en la experimentación es mucho mejor que un número de sistemas sujetos al realismo del modelaje matemático. La simulación permite estudiar y experimentar con las complejas interacciones que ocurren en el interior de un sistema dado, ya sea una empresa industrial, económica o un subsistema de cualquiera de ellas; la observación detallada del sistema simulado conduce a un mejor entendimiento del mismo y proporciona sugerencias para mejorarlo, que de otro modo no sería posible. Muchos de los sistemas complejos reales con elementos estocásticos no pueden ser descritos en forma correcta por un modelo matemático que tan solo puede ser evaluado analíticamente. Además, con el uso adecuado de la animación, las simulaciones se asemejan más al sistema real.

2. Cambios informativos. Se pueden estudiar los efectos de ciertos cambios informativos, de organización y ambientales, en la operación de un sistema, al hacer alteraciones en un modelo y observar los efectos de estas en el comportamiento del sistema. Los diseños de sistemas alternos pueden ser comparados vía simulación para analizar cual cumple con los requerimientos específicos.

3. La inexistencia del sistema. Los sistemas que son investigados no necesariamente deben existir para ser simulados. La única necesidad es que el sistema exista en la mente del creador del mismo.

4. La reducción del tiempo. El tiempo puede ser comprimido en los modelos de simulación. Lo equivalente a días, semanas y meses de las operaciones de un sistema real, muy a menudo, puede ser simulado en segundos, minutos o horas en la computadora; esto significa que un gran número de alternativas simuladas pueden ser investigadas y los resultados pueden estar disponibles a tiempo para influenciar en el diseño de un sistema.

5. Difiere de las especificaciones de los objetivos. En algunos tipos de modelos matemáticos (es decir, los que usan la programación lineal o algún otro tipo de programación matemática) se tiene que especificar un objetivo como principio de la actividad del modelaje y formular este objetivo en términos matemáticos. Además, bajo este objetivo, se encuentran algunas restricciones. En la simulación, no se requiere de tal formulación de metas y restricciones para alcanzarlas. Esto ofrece al diseñador y al cliente la posibilidad de escoger opciones libres de condiciones y buscar un diseño del sistema que eventualmente sea satisfactorio según las decisiones del ambiente. Esta especificación diferida de un objetivo u objetivos puede ser una ventaja en ciertas circunstancias, cuando a quien se le está haciendo el modelo no se puede delimitar inicialmente el objetivo o rendimiento óptimo o cuando la importancia relativa de varios objetivos no es clara.

6. Control del experimento. En la simulación, cada variable puede ser tomada como constante, excepto las variables que son estudiadas. Como resultado, el posible efecto de variables incontrolables en el comportamiento del sistema no se tomará en cuenta, como muy a menudo ocurre cuando los experimentos se realizan en el sistema real.

7. Reproducción de las condiciones de aleatoriedad. En los sistemas de simulación compuestos de elementos que muestran aleatoriedad pero que están estadísticamente predecibles es posible reproducir los eventos aleatorios idénticamente por medio de una secuencia de números seudoaleatorios, pueden tener las características de verdaderos números aleatorios. Esto permite hacer uso de las técnicas de reducción de varianza para mejorar la precisión en donde las características del sistema pueden ser estimadas por cierto esfuerzo de simulación y para agudizar el contraste entre las alternativas de los diseños del sistema.

8. Vocación. La simulación no requiere de un conocimiento alto de matemáticas, como podría ser el modelaje matemático. Esto le facilita a los practicantes de simulación un desenvolvimiento adecuado.

9. Ganar al cliente. Dado que el concepto de simulación es fácil de entender, con mayor disposición los clientes estarán dispuestos a utilizar la simulación en vez del modelaje matemático.

10. Bajo costo. Ha sido estimado, que los estudios de simulación cuestan 2% o menos del capital que cuesta el construir un sistema.

11. Apoyo pedagógico. La simulación puede ser usada como recurso pedagógico, para estudiantes y practicantes, al enseñarles los conocimientos básicos en el análisis estadístico y en la toma de decisiones.

Sin embargo, la simulación presenta algunas desventajas:

1. No proporciona resultados exactos. Si se supone un sistema compuesto de uno o más elementos que tienen un comportamiento aleatorio, entonces el modelo no da el resultado exacto, sólo estimado, y con cierto error.

2. No hay generalidad en los resultados. Los resultados de la simulación se aplican sólo a situaciones que fueron simuladas y no permite generalidades.

3. No es un método de optimización en sentido estricto. La simulación es usada para responder a preguntas de tipo: *¿Qué pasaría si?* pero no para preguntas como: *¿Qué es lo mejor?* En este sentido la simulación no es una técnica de optimización.

4. Largos tiempos de estudio. El estudio de simulaciones no puede ser hecho a corto plazo. Meses de esfuerzo son requeridos para obtener la infomación, construir, verificar y validar los modelos, diseño del experimento y evaluación e interpretación de los resultados. Cada corrida produce una estimación de las verdaderas características para algún parámetro en particular, varias corridas serán entonces necesarias para cada parámetro, por esta razón los estudios son largos. Los requerimientos propios a la simulación deben cubrirse antes de que los resultados sean necesarios. En la práctica, desafortunadamente, los resultados usualmente son necesitados para *ayer*. Un estudio de simulación puede no estar autorizado hasta que el proyecto principal de cierta empresa lo requiera, es entonces cuando se vuelve una prioridad y para ese instante ya no habrá el tiempo suficiente para recopilar los datos necesarios y efectuarlo.

5. Altos costos para cumplir con los requerimientos necesarios para hacer estudios de simulación. Para establecer y mantener una buena capacidad para hacer estudios de simulación es necesario comprometerse enormemente con el personal de la empresa, realizar la adquisición de software y hardware, así como el entrenamiento y otros costos de soporte. Algunas empresas pueden tener una o dos personas que posean cierta experiencia con proyectos de simulación, tales personas pueden ser ayudadas por consultores externos cuando se presente algún proyecto de simulación. Algunas otras pueden simplemente contratar consultores, sin embargo, se puede crear cierta dependencia con estos agentes.

6. **Mal uso de la simulación.** Hay varias facetas para balancear y comprender el estudio de la simulación como resultado, una persona debe ser educada en una variedad de áreas (análisis de entradas, diseño y experimentación, análisis del resultado) antes de ser un practicante de la simulación. Este hecho muchas veces es ignorado, resultando así situaciones donde el analista sabe construir modelos de simulación pero no sabe analizar dichos resultados finales. Este tipo de personas no podrán conducir un estudio de simulación balanceado y comprensivo. Como resultado esos estudios estarán erróneos o incompletos, y presentarán alguna falla. Algunos errores comunes son:

- No se logra obtener un conjunto de objetivos bien definidos.
- Nivel de detalle del modelo inapropiado.
- Falta de comunicación con la administración en forma continua durante el estudio.
- Trato del estudio como si fuera un ejercicio complicado de programación.
- Falta de personal capacitado con conocimientos de investigación de operaciones y estadística.

Puede parecer que no hay una respuesta tajante a la pregunta de cúando utilizar la simulación. Técnicamente la respuesta correcta es que la simulación debe ser usada cuando las ventajas son más que las desventajas. Esto no es muy operacional, en la práctica, la simulación debe ser seleccionada cuando ninguna de las siguientes dos condiciones prevalecen:

- Las suposiciones requeridas para un modelo analítico apropiado no son lo suficientemente satisfactorias para definir el sistema real.
- O cuando el modelo no puede ser analizado analíticamente.

Quizá esto puede reducir severamente el rango de problemas para los que la simulación es apropiada, pero como se vio, no necesariamente, por la complejidad de las situaciones que se presentan en la realidad y la dificultad en asociarles un modelo analítico.

Los modelos analíticos y de simulación pueden ser dos alternativas. Uno de los usos de la simulación más fructíferos pero limitado ha sido en conjunción con modelos analíticos. Esta aproximación toma ventajas de las mejores características de los dos métodos mientras minimiza las desventajas. Uno de estos estudios tiene que ver con la planeación de uso de una ambulancia. Utilizando la aproximación recursiva de optimización-simulación, un modelo de programación mixta primero genera la solución óptima del personal y planeación de la ambulancia, y luego un modelo de simulación evalúa su efectividad basada en las soluciones dadas por el modelo de programación mixta tales como la espera de un paciente y la utilización del personal. Si esta medidas no son satisfactorias, las condicionantes del modelo de programación mixta serán alteradas o serán construidas utilizando el análisis de regresión. Este proceso continúa de esta forma hasta que una solución aceptable sea encontrada.

En conclusión, la simulación no es el realizar un experimento más de optimización, ofrece enormes ventajas, pero no hay que olvidar, que también posee algunas desventajas. Afortunadamente la mayoría de los inconvenientes asociados a su empleo no serán tan importantes con el tiempo, gracias a que las herramientas de simulación, metodología, educación y operaciones computacionales han presentado y seguirán experimentado, enormes mejorías.

CAPÍTULO II
CONSTRUCCIÓN DE MODELOS PARA EVENTOS CON TIEMPOS DISCRETOS

II.1 PLANEACIÓN DE LA SIMULACIÓN

Se debe reconocer que el modelaje y codificación detallada son tan solo una parte de la acción necesaria para comprender o diseñar un sistema complejo; ya que hay que tomar en cuenta otros aspectos tales como el diseño de experimentos estadísticos así como de los costos administrativos y de personal. De igual importancia es mencionar que para llevar un estudio de simulación con éxito se necesitan muchos conocimientos, no es suficiente dominar el área de simulación ya que además se deben contemplar las áreas de probabilidad, estadística, muestreo, así como estar totalmente inmerso en los detalles del problema. Por lo que es conveniente seguir estos pasos al momento de hacer un análisis de simulación:

1. Formular el problema
2. Conceptualizar el modelo
3. Obtener y procesar los datos
4. Formular el modelo matemático
5. Estimar los parámetros necesarios
6. Implementar el modelo
7. Evaluar el modelo
8. Validar los resultados
9. Diseñar el experimento de simulación
10. Analizar los resultados
11. Conclusiones y recomendaciones

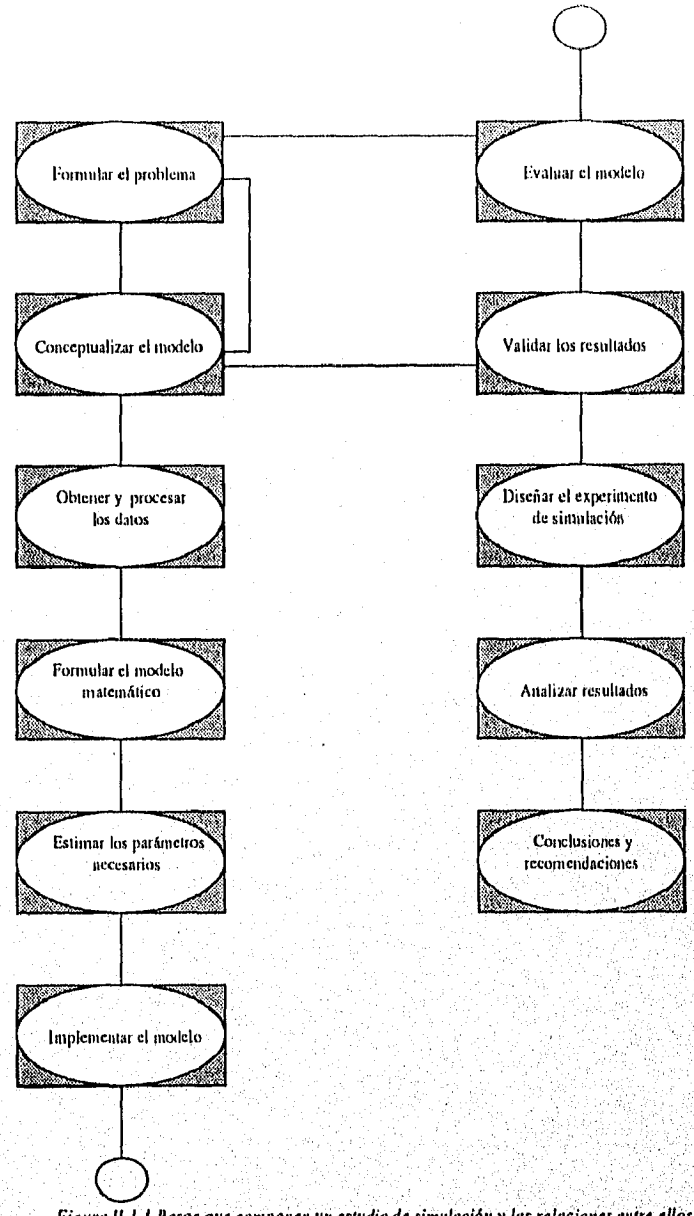


Figura II.1.1 Pasos que componen un estudio de simulación y las relaciones entre ellos

No todos los estudios de simulación incluyen íntegramente estos pasos ni tampoco el mismo orden; quizá algunos contengan más detalle en ellos o presenten diferente arreglo, sin embargo, la mayoría de los autores coinciden en los mismos conceptos, además de que una simulación no es un simple proceso secuencial, muchas veces es necesario regresar a un paso anterior como se muestra en la *Figura II.1.1*, para que realmente el sistema sea parecido al modelo que se intenta simular. También se debe considerar que conforme se avanza en el estudio, el conocimiento sobre el sistema mejora, obteniéndose así una mayor comprensión sobre el sistema y la simulación

Lo que en seguida se intenta, es dar un esquema general de cómo una simulación se debe o se puede llevar a cabo con el fin de obtener los mejores resultados, por lo que se analizarán los pasos mencionados anteriormente, sin profundizar demasiado en ellos, señalando sólo los factores más importantes que deben tomarse en cuenta.

II.1.1 FORMULACIÓN DEL PROBLEMA

Aunque pudiera parecer obvio, muchas veces es difícil formular un problema, para un buen resultado de simulación es necesario detallarlo de forma correcta, siendo este paso el más crítico para alcanzar el éxito en la tarea emprendida. El estudio debe comenzar con una exposición clara de los objetivos y la especificación del problema, sin esta será difícil obtener resultados claros de la simulación. Se encontrará en consecuencia que la exposición inicial del dilema varia considerablemente de su versión final, ya que la formulación de la problemática es un proceso secuencial que generalmente requiere una reformulación continua y progresiva y un refinamiento de los objetivos que se pretenden lograr en el experimento durante su realización. Soluciones apropiadas, como es de esperarse, para problemas mal formulados no pueden ser obtenidas, así mismo hay que indicar, aunque parezca obvio, que para que un problema pueda ser formulado correctamente debe ser identificado. En este paso es importante que el analista se familiarice con los detalles, así como con el propósito del sistema. Muchas veces los objetivos toman la forma en:

- Preguntas que se deben contestar
- Hipótesis que se deben probar
- Efectos a futuro

Pero, no hay que limitarse a especificar preguntas, sino también hay que distinguir los criterios y objetivos necesarios para evaluar las posibles respuestas a estos cuestionamientos. Adicionalmente se debe tomar en cuenta la existencia de sistemas alternos así como de un criterio para evaluar la eficiencia de estas opciones. Dentro de este criterio se pueden encontrar las facilidades económicas, técnicas y políticas.

Muchas veces los objetivos podrían consistir en estimar los efectos que ciertos cambios en los parámetros del modelo ocasionan, así como el impacto a las características operacionales o la influencia que las variables exógenas puedan tener sobre las variables endógenas. Un ejemplo de esto podría ser experimentar con diferentes valores de la esperanza matemática

de los tiempos de llegada a una estación de servicio (fila de un banco, de una gasolinería, ect.) y estimar la esperanza del tiempo de espera y la del tiempo de ocio, para cada uno de dichos valores.

Sin embargo, en muchas ocasiones el analista necesita interpretar correctamente lo que sucede en la empresa o institución, acción que de no llevarse correctamente dará como resultado que los datos obtenidos disten de la realidad, para evitar esto se tiene que relacionar y recopilar información con los responsables de esta, es decir, la persona de emular el sistema tendrá que ser capaz de comunicarse abiertamente y de manera clara con personas ajenas a su disciplina, en ocasiones al analista le puede costar trabajo lograr la comunicación necesaria, es por esto que se requiere de mucha paciencia para lograr comprender la situación.

II.1.2 Conceptualización del Modelo

Es quizás este paso, para muchos autores, el más difícil dado que intuitivamente y de forma general se deben conceptualizar los componentes del sistema, así como su entorno y los objetivos. Es en este apartado donde hay que especificar el propósito del modelo y las restricciones bajo las cuales se crea la abstracción o el modelo formal, es decir, se torna necesario efectuar un análisis cuidadoso del ambiente del sistema, de tal forma que se puedan definir de forma exacta cuales son los objetivos específicos, si éstos pueden expresarse en forma cuantitativa, cuales son los parámetros y las variables relevantes, cuales son las relaciones causa-efecto, que datos se requieren, etc. Se debe crear una representación matemática precisa del sistema. Para llevar a cabo un buen análisis se debe preguntar por ejemplo, lo siguiente:

- ¿Cuáles son los objetivos específicos del estudio?
- ¿Pueden ser expresados en forma cuantitativa, en términos del criterio de ejecución del sistema?
- ¿Cuáles son los parámetros relevantes?
- ¿Cuáles son las variables de estado, las variables de decisión y los parámetros?
- ¿Cuáles son las relaciones de causa-efecto?
- ¿Qué datos se requieren? ¿Puede disponerse de ellos en forma inmediata? Si no ¿Cuánto esfuerzo se requiere para obtenerlos?
- ¿Con qué recursos de cómputo se cuenta? (hardware y software)
- ¿Con qué recursos humanos se cuenta?

II.1.3 Obtención y Procesamiento de los Datos

Muchas veces este paso es tomado en segundo término en el análisis de simulación, debido a que algún tipo de recolección de datos se debió haber llevado a cabo en la formulación del problema. En este paso se deben capturar los datos disponibles que se requieren para la simulación del comportamiento del sistema y que a partir de ciertas actividades se

transforman en información; se debe también reducir y analizar los datos para lograr que el modelo sea más explícito. Es muy importante definir con claridad y de forma precisa los datos necesarios para obtener los resultados deseados.

En general existen tres fuentes que generan información y se dividen en tres grupos:

1. Series de tiempo.
2. Opiniones de expertos.
3. Estudios de campo (estadísticas).

Para estudios de campo los métodos de colección de datos pueden variar dependiendo del problema, desde métodos muy simples, como coleccionar la información manualmente hasta utilizar técnicas modernas. Para determinar como coleccionar la información es necesario tomar en cuenta los siguientes puntos:

- La habilidad para registrar la información lo suficientemente rápido y obtener así el nivel de precisión deseado.
- El impacto del proceso de colección de información en los atributos del comportamiento del sistema que son medidos.
- La facilidad en la conversión de la información a una forma de proceso computarizado.
- El costo por coleccionar la información.

En muchas situaciones la observación directa y el registro manual será suficiente, pero muchas veces cuando una persona es analizada, la presencia de un observador puede influenciar su comportamiento, como en el caso de un cajero en un supermercado. O quizá suceda que la acción sea demasiado rápida para que alguna persona pueda llevar el registro de forma eficiente, como por ejemplo cuando se realiza la colección de información del trafico vehícular; muchas veces los observadores cometen errores al estar registrando la información. Existen otros tipos de colección de datos entre los que se encuentran los métodos automatizados que pueden incluir videocámaras, luces automáticas, equipo de presión sensorial y lectores ópticos.

Para determinar el tamaño de la muestra se deben tomar en cuenta los costos y la precisión de la información. Tamaños de muestra grandes permiten estimar las características de la población con mayor exactitud, pero esto requiere de mayor tiempo para coleccionar los datos y transportarlos a archivos que puedan ser analizados en la computadora, ocasionado así mayor costo. La teoría estadística puede definir la precisión de una estimación con un tamaño de muestra dado, permitiendo determinar cualitativamente si el costo vale la pena.

Cierta información utilizada para definir los modelos puede ser determinística, es decir, se sabe con seguridad, pero mucha más es probabilística. Dada cierta información probabilística, existen básicamente dos formas de incluirlas en los modelos, una de ellas es utilizar la muestra para representar la distribución de probabilidad o determinar una distribución teórica de probabilidad que sea apropiada para la información.

Existen varias razones por las cuales es necesario disponer de un sistema eficiente para el procesamiento de datos y alcanzar el éxito al estar haciendo un estudio de simulación, por ejemplo:

- La información descriptiva y cuantitativa (datos) referentes al sistema que se va a investigar, constituye un requisito previo a la formulación del problema.
- Los datos que hayan sido reducidos en forma significativa pueden sugerir hipótesis con cierta validez, las cuales se usarán en la formulación de los modelos matemáticos que describen el comportamiento de un sistema dado.
- Los datos también pueden sugerir mejoras o refinamientos en los modelos matemáticos que existen en el sistema dado.
- Es necesario que los datos, reducidos a una forma final, se utilicen para estimar los parámetros de las características de operación relativas a las variables endógenas, exógenas y de estado del sistema.
- Hay que considerar que sin tales datos es imposible probar la validez de un modelo de simulación.

Es importante mencionar que existen seis funciones importantes del procesamiento de datos que forman una parte integral del procedimiento para implantar los experimentos de simulación en computadoras: recolección, almacenamiento, conversión, transmisión, manipulación y salida.

Recolección. Es el proceso de captación de los hechos disponibles, con lo cual estos pueden ser procesados posteriormente, cuando sea necesario.

Almacenamiento. En realidad el proceso de recolección y almacenamiento de datos ocurren simultáneamente, pues el primero implica que los datos sean o hayan sido almacenados. Esta tarea es sumamente costosa y laboriosa, dado que comprende la identificación, revisión, edición, codificación (asignación de claves), transcripción y verificación de ellos.

Conversión. Este proceso es crucial en la determinación de la eficiencia del procesamiento, pues la mala conversión puede ocasionar una mala interpretación del problema. El proceso trata de modificar la información de cierta manera para después ser manipulada.

Transmisión. Bajo ciertas circunstancias, existen problemas adicionales en la conversión de los datos de una forma a la otra, que implica una transmisión de ellos, esto es, el transporte de la información desde una localidad hasta el lugar en donde será procesada.

Manipulación. Una vez que los datos han sido recolectados, almacenados, convertidos a una forma eficaz y transmitidos al lugar de procesamiento final, resulta posible entonces comenzar con las operaciones de manipulación de la información como sería el realizar la estadísticas, ajuste de curvas de probabilidad, etc. y la preparación de estos para su salida final.

Salida. Una vez manipulada la información, es necesario para continuar con el análisis de simulación un reporte de esta operación.

II.1.4 FORMULACIÓN DEL MODELO MATEMÁTICO

Es en este momento cuando se debe llevar el problema del mundo real a un modelo matemático, es decir, interpretar de la realidad las características que puedan ser reflejadas por medio de las matemáticas, analizar su comportamiento y exponerlo de alguna manera en un modelo. Para evaluar la efectividad del sistema es necesario identificar alguna(s) medida(s) de comportamiento para que de esta manera se logre juzgar su efectividad. Estas medidas pueden ser seleccionadas de las variables endógenas del sistema. Algo que puede ayudar a formular el modelo de mejor manera es tener conocimiento del problema, así como de las operaciones matemáticas, experiencia en cuanto al trato con modelos, procedimiento de prueba y error, y lo más importante, algo de suerte.

El modelo debe contener el detalle suficiente para capturar la esencia del sistema y quizá no sea necesario tener una correspondencia de uno a uno entre los elementos de la realidad con los elementos del modelo. Un diseño con demasiado detalle seguramente elevará los costos al tratar de programarlo y ejecutarlo.

Es conveniente señalar algunas consideraciones que no se deben olvidar al realizar la construcción del modelo:

Variables del modelo. No es difícil saber cuantas variables endógenas debe contener, dado que esto se determina al comienzo del análisis, es decir, los objetivos del modelo. La parte complicada del estudio es cuando se debe decidir la elección de las variables exógenas, porque muy pocas variables pueden llevar a un modelo no válido y la abundancia de ellas hacer imposible la simulación, o bien complicar el diseño. Como ya se señaló, detallar demasiado, usar más variables de las necesarias hará, muy probablemente, más costoso el modelo y la simulación, y se puede complicar al estar programando. También es importante identificar cuales variables pueden ser estocásticas.

Complejidad de los modelos. Se puede suponer que los problemas reales son muy complicados y que los modelos matemáticos que tratan de representarlos también lo serán de alguna manera, sin embargo, esto no quiere decir que todos lo sean y por lo tanto hay que tomar con calma este hecho. Como prioridad se debe estar interesado en formular modelos matemáticos que produzcan descripciones o predicciones, razonablemente exactas, referentes al comportamiento de un sistema dado y reduzcan a la vez, el tiempo de computación y programación. No hay una completa interdependencia de estas características en los modelos matemáticos, ya que tanto el número de variables en un modelo, como su complejidad, se encuentran directamente relacionadas con los tiempos de programación, cómputo y validez.

Eficiencia de computación. Se entiende por eficiencia computacional la cantidad de tiempo de cómputo necesario para lograr algún objetivo experimental específico. Es importante marcar ciertos objetivos, como por ejemplo: es posible que se desee reducir el tiempo de cómputo requerido para generar los valores de las variables endógenas sobre un periodo específico o se requiere simular el comportamiento de las variables endógenas en un periodo *x*; quizá el analista este interesado en la reducción del tiempo de computación requerido para lograr algún nivel de precisión estadística previamente determinado, al estimar los valores de ciertos parámetros estadísticos generados por el modelo de simulación.

Tiempo consumido en la programación de la computadora. El tiempo requerido para escribir un programa que genere los tiempos planificados para las variables endógenas de un conjunto particular de modelos matemáticos, depende en parte del número de variables utilizadas en los modelos y de su complejidad. Si algunas variables usadas en los modelos son estocásticas, entonces tanto el tiempo de programación como el de computación deben equilibrase con los aspectos de validez y velocidad del cálculo. También existe la posibilidad de que el analista formule su modelo de forma que satisfaga los requerimientos de algún lenguaje de simulación.

Realismo en los modelos. Al hablar del realismo en los modelos se hace referencia a la validez del problema. Es decir, *¿el modelo describe adecuadamente al problema?, ¿proporciona predicciones razonablemente buenas acerca del comportamiento del sistema, en periodos futuros?* A menos que la respuesta a una o ambas de estas preguntas sea afirmativa, entonces la dificultad de los modelos se reducirá considerablemente y el experimento de simulación se convertirá sólo en un ejercicio de lógica deductiva.

Compatibilidad con el tipo de experimentos que se van a realizar. Dado que el objetivo principal al conceptualizar un modelo matemático es el de dirigir experimentos de simulación, se debe pensar en qué forma particular se tomarán las características del diseño de los experimentos que deben incorporarse en los modelos.

Sin embargo, pocas veces estas propiedades se cumplen en su totalidad al tratar con problemas del mundo real, dadas ciertas dificultades que se presentan, ejemplo de ellas son las siguientes:

- Quizá es imposible cuatificar o medir ciertos tipos de variables que afectan el comportamiento del sistema.
- Puede suceder que el número de variables por considerarse al describir un sistema dado, exceda la capacidad de nuestra computadora.
- Es posible que se desconozcan algunas variables exógenas significativas que afectan la salida del sistema.
- También se pueden pasar por alto algunas de las relaciones entre las variables exógenas y las endógenas del sistema.
- Las relaciones entre las variables que afectan el comportamiento del sistema, son en muchos casos tan complejas que no pueden expresarse como una o más ecuaciones matemáticas.

Existen dos tipos básicos de diseño para formular modelos matemáticos:

Diseños generalizados. Este diseño trata de describir el comportamiento de un sistema completo.

Diseños modulares o de bloque. En este diseño se intenta sintetizar un modelo generalizado a partir de un conjunto de modelos que describen los componentes importantes de un sistema.

El objetivo principal de cualquier modelo de simulación es ayudar en la toma de decisiones y por ende en esto hay que basar su conceptualización.

Como se ha visto, para formular un modelo matemático no existen leyes o reglas que aseguren su éxito. Sin embargo, se requiere de la utilización adecuada de ciertas técnicas, métodos o marcos de referencia que dan criterios útiles en este tópico tan complejo. Así mismo se requiere:

- Especificar variables y parámetros.
- Especificar relaciones lógicas de causa - efecto.
- Especificar sus componentes (en forma de diagramas de flujo)

II.1.5 ESTIMACIÓN DE LOS PARÁMETROS

Ya que se ha recolectado la información y se han formulado quizá varios modelos matemáticos que describen el comportamiento del problema, es necesario estimar los valores de los parámetros de dichos modelos y probar su significación estadística. Como se mencionó anteriormente, no tendría mucho sentido utilizar la simulación en problemas de tipo determinístico, ya que en realidad lo que interesa al usarse un modelo de simulación es la obtención de información *confiable* de fenómenos generalmente estocásticos.

Por ello, es necesaria la obtención de parámetros o características que definan al sistema, las cuales son imprescindibles en distribuciones teóricas o empíricas de probabilidad.

Antes de intentar la estimación de los parámetros de las características operacionales de un sistema, se debe tener conocimiento amplio, cuando menos de las técnicas ordinarias de estimación por mínimos cuadrados y de los procedimientos clásicos de pruebas estadísticas. Para estimar estos, por lo general se utiliza alguno de los siguientes métodos:

- Métodos de máxima verosimilitud (Maximiza la probabilidad de ocurrencia de una muestra)
- Método de los momentos (Iguala los momentos muéstrales a las poblaciones)
- Método de los mínimos cuadrados (Minimiza las distancias entre los puntos de tal forma que se ajusten a una curva).

Los dos primeros son estadísticos y el último es determinístico. Algunas veces no es posible utilizar los métodos mencionados anteriormente, en esos casos debemos utilizar procedimientos iterativos para encontrar buenos estimadores.

II.1.6 IMPLEMENTACIÓN DEL MODELO

Con el modelo ya definido, si es necesario, se deberá transcribirlo de manera que sea posible llevarlo a la computadora y así obtener los resultados deseados.

Los pasos a seguir para formular un programa de computación son:

- Elaboración del diagrama de flujo.
- Indicar datos de entrada y condiciones iniciales.
- Codificar o diseñar el programa en algún lenguaje de propósito general o de propósito específico, o quizá se pueda realizar en un paquete estadístico o en una hoja de cálculo.
- Verificar errores.
- Probar el programa desechando todos los errores.
- Generar resultados.
- Emitir reportes de salida.

En el primer paso se requiere de una formulación de diagrama de flujo o de un diagrama de bloques que bosqueje la secuencia lógica de los eventos que realizará la computadora.

Al momento de seleccionar el lenguaje para escribir el código para computadora, cabe señalar que existen lenguajes de simulación muy potentes como el GPSS, Simula, Gasp, SIMNET, SIMSCRIPT, Dynamo, Simulate, Siman, Slam, etc. De igual manera existen lenguajes de propósito general como Fortran, Algol, Cobol, C, Pascal que también son muy potentes, pero se requiere más tiempo de programación, dado que los lenguajes de propósitos específicos fueron escritos para facilitar la programación, corrección y experimentación de ciertos tipos de sistemas, aunque el tiempo de ejecución es mayor en los lenguajes de propósitos específicos. Más adelante se analizará las ventajas de los diferentes lenguajes, así como los de propósito general y de propósito específico.

Un aspecto más en la fase de programación del desarrollo de un experimento de simulación en computadora, es relativo al problema de los datos de entrada y las condiciones iniciales para el experimento, ya que los ensayo de simulación son dinámicos, surgen las interrogantes respecto a los valores que se les deberían asignar a las variables y parámetros del modelo en el momento en que se comienza a simular el sistema, es decir, se deberá forzar la entrada al sistema en un punto particular del tiempo, pero al hacer esto hay que analizar cuales son las suposiciones a considerar con respecto a las condiciones de equilibrio o las condiciones iniciales del sistema.

Un problema que surge al programar los modelos es el del desarrollo de las técnicas numéricas para la generación de datos. Si alguna o algunas variables exógenas en el modelo son estocásticas y tienen una distribución de probabilidad, se debe seleccionar algún método numérico para generar los valores con dicha distribución de probabilidad. Para generar números aleatorios existen varias formas, estas son: métodos manuales (utilizar una moneda, dado, etc.), tablas de libros, métodos de computación analógica (impulsos electrónicos aleatorios, tiempos, etc.) y métodos de computación digital. Para obtener valores uniformes, existen algunos métodos como los métodos congruenciales y los métodos de los cuadrados medios. Para valores de distribución de probabilidad no uniforme existen varios métodos que se basan en valores uniformes como: método de la transformación inversa, método del rechazo, método de composición, entre otros.

II.1.7 EVALUACIÓN DEL MODELO

La evaluación de un modelo consiste en recolectar sistemáticamente los datos producidos por la simulación tales como parámetros de los modelos, sus medias y sus varianzas, utilizando técnicas estadísticas se evalúa la significancia de estos, así como hacer un juicio inicial sobre la adecuación de los modelos, es decir, pruebas sobre la bondad de ajuste de las distribuciones simuladas. Estas pruebas y técnicas estadísticas podrían ser:

- Estudios referentes a las medias: pruebas de una muestra relativa a las medias y las diferencias entre ellas.
- Análisis en torno a las varianzas: pruebas de la Ji cuadrada y pruebas F.
- Exámenes basados en el conteo de datos: pruebas referentes a las proporciones, diferencias entre k proporciones, tablas de contingencia y pruebas de bondad de ajuste.
- Pruebas no paramétricas: prueba del signo, pruebas basadas en sumas de rangos, la prueba de la mediana, la prueba U, pruebas de corridas y pruebas de correlación en serie.

Es importante que se apliquen pruebas que permitan detectar las violaciones en las suposiciones fundamentales de los modelos, estas podrían comprender las pruebas para detectar:

- errores en las variables
- colinearidad múltiple
- heterosedasticidad
- autocorrelación
- identificación

En este paso es importante que el analista se pregunte:

- ¿Se incluyeron variables que no eran necesarias, en el sentido que no contribuyen a la capacidad para predecir el comportamiento de las variables endógenas del sistema?
- ¿Faltó incluir una o más variables exógenas que pudieran afectar al comportamiento de

las variables endógenas en el sistema?
- ¿Se formuló incorrectamente una o más relaciones funcionales entre las variables?
- ¿Se apreció debidamente las estimaciones de los parámetros de las características operacionales del sistema?
- ¿Son estadísticamente significativas las estimaciones de los parámetros en el modelo?

Si el analista puede contestar lo anterior satisfactoriamente entonces se continúa con el análisis, y sino, se debe regresar a la conceptualización del modelo.

II.1.8 VALIDACIÓN DE RESULTADOS

Para validar los resultados arrojados por el programa de computadora se requiere una combinación de suposiciones prácticas, teóricas, estadísticas y filosóficas complejas. Como se puede ver en la *Figura II.1.8.1*, el analista debe preguntarse y analizar que tan bien coinciden los valores simulados de las variables endógenas con datos históricos conocidos y qué tan exactas son las predicciones del comportamiento del sistema real hechas por el modelo de simulación para periodos futuros. La verificación se enfoca en la consistencia interna del modelo, mientras que la validación tiene que ver con la correspondencia entre el modelo y la realidad. Lo más usual para validar un modelo es basarse en los siguientes aspectos:

- La opinión de expertos sobre los resultados de la simulación.
- La exactitud en la predicción del futuro.
- La comprobación de falla del modelo de simulación, al utilizar datos que hacen fallar al sistema real.
- La aceptación y confianza en el modelo que tenga la persona que hará uso de los resultados que arroje el experimento de simulación.

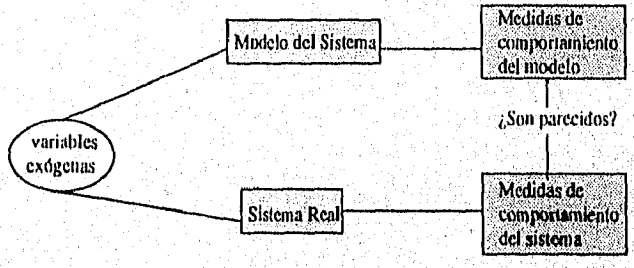


Figura II.1.8.1 Aproximación de la validación del modelo

II.1.9 DISEÑO DEL EXPERIMENTO

Una vez que se obtiene un modelo aceptable, hay que diseñar los experimentos deseados para simular. Se deberán evaluar los diferentes diseños del sistema propuesto, de acuerdo a las medidas de comportamiento producidas por el modelo. En esta fase se contestará a las preguntas del diseño experimental. Es importante elegir las diferentes alternativas y las combinaciones de estos, así como el orden de la experimentación. También se deberá tratar de minimizar los errores puramente aleatorios, de manera que resalten los efectos de los diseños, esto se hace a través de técnicas de reducción de varianza.

Básicamente esta fase consiste en generar los datos deseados y realizar el análisis de sensibilidad de los índices requeridos. Así mismo, se diseña la aplicación del alternativas o políticas para medir sus efectos, utilizando intervalos de variación. Es importante una vez más que el analista se conteste las siguientes preguntas:

- ¿Cuántas alternativas se deben tomar en cuenta?,
- ¿Cuánto debe durar la corrida de las distintas alternativas?,
- ¿Qué condiciones iniciales utilizar para cada alternativa?
- ¿Cuál debe ser la precisión necesaria en las medidas de comportamiento del sistema?
- ¿Cuántas corridas deberán ser hechas para cada alternativa?

II.1.10 ANÁLISIS DE RESULTADOS

Este paso consiste en interpretar y analizar los resultados que se obtuvieron de la simulación, esta acción se encuentra descrita en los siguientes pasos:

- *Recolección y proceso de los datos simulados*, en forma parecida a la usada para los datos reales, aunque resulta mucho más compleja, ya que la parte aleatoria interviene en forma complicada, además de la influencia del tiempo.

- *Cálculo de pruebas estadísticas*, en este caso para determinar la significancia del efecto de los distintos diseños alternos, así como sus combinaciones. Típicamente estas pruebas consisten en un intervalo de confidencia para medir el funcionamiento de un diseño de sistema en particular.

- *Interpretación de resultados*, observando los efectos de los cambios en las variables endógenas.

Es importante dar un orden de calidad entre los diferentes diseños basándose en las medidas de funcionamiento del sistema. No se debe olvidar comparar las políticas alternativas y sugerir las acciones pertinentes con las cuales se toma la mejor decisión, esperando con ello obtener los mejores resultados de la simulación.

II.1.11 Conclusiones y Recomendaciones

Es importante que los resultados matemáticos no se sólo queden como números, sino que exista una interpretación precisa con respecto al problema. El analista después de hacer una simulación y un análisis profundo a un sistema, deberá tener en este momento la capacidad de dar recomendaciones, es decir, para que el sistema estudiado trabaje de mejor manera que es lo que se necesita hacer.

Siguiendo paso a paso este esquema, se aumentan las posibilidades de obtener un modelo de simulación que arroje los mejores resultados.

II.2 Conceptos Básicos de Modelos para Eventos con Tiempos Discretos

Los modelos para eventos con tiempos discretos tienen mucho que ver con las líneas de espera, la formación de éstas es un fenómeno común que ocurre siempre que la demanda presente de un servicio exceda a la capacidad actual de proporcionarlo. En la industria y en otros sectores, deben tomarse decisiones respecto a la cantidad de capacidad que debe proporcionarse. Muchas veces es difícil predecir con exactitud cuándo llegarán las unidades que buscan ser atendidas y/o cuánto tiempo será necesario para dar ese servicio, proporcionar demasiada atención implica costos mayores y por otro lado no tener la suficiente capacidad de servicio también causa esperas excesivamente largas en ciertos momentos, esto también puede ser costoso en el sentido de que, ya sea por un costo social, por un costo causado por la pérdida de clientes, por el costo de empleados ociosos o por algún otro gasto importante. Muchas veces al hacer un estudio de simulación para este tipo de modelo, lo que se busca es lograr un balance económico entre el costo de servicio y el costo asociado con la espera de ese servicio.

Un *evento* es un punto particular en el tiempo donde el estado del sistema cambia, por ejemplo, cuando una persona llega a una línea de espera y se une a ella, la variable de estado que lleva la cuenta del número de personas en la fila es alterada. Incorporando al modelo la facilidad para cambiar estas variables conforme el tiempo avanza y las secuencias de eventos, es como se puede describir el comportamiento del sistema. Un evento está definido como el instante de la ocurrencia que puede cambiar en el estado del sistema.

Se enuncian a continuación algunas definiciones de simulación de modelos de eventos con tiempos discretos, propuestas por algunos autores:

"Una simulación de modelos de eventos con tiempos discretos puede ser analizada como un modelo de la interacción de eventos discretos que ocurren en el sistema y en las

variables de estado del sistema".[1]

"Una simulación de un modelo de eventos con tiempos discretos es donde el estado del modelo (o del sistema siendo modelado) cambia en un conjunto de puntos del tiempo discretos que posiblemente son aleatorios. Por ejemplo cuando un doctor empieza a examinar a un paciente en una sala de emergencia, el acto de comenzar a examinar al paciente es un evento y ocurre un punto del tiempo. Después cuando el doctor termina la examinación, el acto de terminar la examinación es un evento que también ocurre en un momento dado del tiempo. En una simulación continua uno o más aspectos de el estado del modelo cambian continuamente en el tiempo. Casi siempre existe la posibilidad de discretizar un procesos continuos, es por esto que una simulación de eventos discretos puede ser usada para modelos de sistema continuos exitosamente". [2]

"La simulaciones de modelos de evento con tiempos discretos, en contraste con las simulaciones continuas, son en las que los cambios en el es estado del sistema ocurre en puntos aleatorios del tiempo (al contrario de continuamente) como resultado de la ocurrencia de estos son discretos. Los bloques básico para la construcción de un modelo para un simulación de evento discretos son los estados posibles y los eventos, un reloj simulador para registra el paso del tiempo simulado, un mecanismo para generar aleatoriamente los distintos tipos de eventos y un mecanisma para después generar las transiciones de los estados. Por ejemplo los eventos que cambian el estado del sistema pueden ser lanzar una águila o un sol". [3]

"Una simulación de eventos discretos tiene que ver con el modelaje de un sistema conforme pasa el tiempo bajo una representación en donde las variables de estado cambian instantáneamente en puntos del tiempo separadamente, en términos matemáticos se puede decir que el sistema puede cambiar en puntos del tiempo contables". [4]

En estas definiciones uno se puede percatar que los modelos de eventos con tiempos discretos tienen que ver con la ocurrencia de cierto evento en un punto del tiempo, siendo estos discretos y generalmente aleatorios.

Los componentes básicos de estos modelos son los siguientes: los *clientes* (sea una persona o producto, también llamados transacciones) que requieren de un servicio y se generan a través del tiempo en una fuente de entrada. Estos clientes entran al sistema de colas y se unen a ella, como se muestra en la *Figura II.2.1*. En determinado momento se selecciona un miembro de la *fila*, para proporcionarle el servicio, mediante alguna regla conocida como *disciplina de la cola*. Después, en un *mecanismo de servicio*, se lleva a cabo el servicio o servicios requerido por el cliente. Finalmente, el cliente se retira del sistema de colas.

---

[1] Hoover, Stewart V., Perry, Ronald F. *Simulation a Problem Solving Approach*, pág. 44
[2] Schiber, Thomas J. *An Introduction to Simulation Using GPSSIII*, pág. 16
[3] Hiller, Frederick S., Lieberman, Gerald J., *Introducción a la Investigación de Operaciones*, pág. 858
[4] Law, Averill M., Kelton W. David, *Simulation Modeling and Analysis*, pág. 7

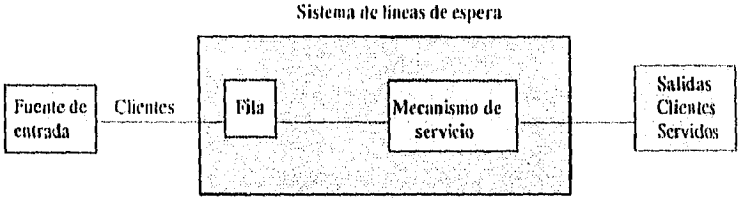


Figura II.2.1 Esquema del proceso de modelos para eventos con tiempos discretos

Existen distintos elementos del proceso de colas que se analizan a continuación:

Fuente de Entrada. Es la descripción de la población de la cual se originan los clientes, es decir, el número total de clientes que pueden requerir el servicio en determinado momento, es en otras palabras el número total de clientes potenciales. Puede suponerse que el tamaño es infinito o finito (de modo que también se dice que la fuente de entrada es ilimitada o limitada). Muchas veces se toma a la fuente de entrada como ilimitada, por su facilidad. También en muchas ocasiones el patrón estadístico mediante el cual se generan los clientes a través del tiempo es de acuerdo a un proceso Poisson, Una suposición equivalente es que la distribución de probabilidad del tiempo que transcurre entre dos llegadas consecutivas es exponencial.

Cola. Se caracteriza por el número máximo permisible de clientes que puede admitir la línea de espera. Las colas pueden ser finitas o infinitas. La suposición de una cola infinita es la estándar para la mayor parte de los modelos, incluso en situaciones en las que de hecho existe una cota superior sobre el número permisible de clientes.

Disciplina de la cola. Se refiere al orden en el que se seleccionan sus miembros para recibir el servicio. Un ejemplo de esto es el primero en llegar será el primero en ser atendido.

Mecanismo de Servicio. Consiste en una o más instalaciones de servicio, cada una de ellas con uno o más canales paralelos de servicio llamados *servidores*. Si existe más de una instalación de servicio, puede ser que se atienda al cliente a través de una secuencia de ellas (canales de servicio en serie). En una instalación dada, el cliente entra en uno de estos canales y el servidor le presta un servicio completo. Se debe especificar el arreglo de las instalaciones y el número de servidores (canales en paralelos) en cada una. Los modelos más elementales suponen una instalación, ya sea con uno o con un número finito de servidores.

El tiempo que transcurre desde el inicio del servicio para un cliente hasta su terminación en una instalación se llama **tiempo de servicio** (o duración del servicio). Un modelo de un sistema de eventos con tiempos discretos debe especificar la distribución de probabilidad de los tiempos de servicio para cada servidor, lo más usual es que se suponga la misma distribución para todos los servidores.

Un Proceso de Colas Elemental. Como ya se vio este modelo se aplica a diferentes situaciones. El tipo que más prevalece es donde existe una sola línea de espera (que puede estar vacía en ciertos momentos) los clientes se forman frente a una instalación de servicio, dentro de la cual se encuentra uno o más servidores. Cada cliente generado por una fuente de entrada después espera un poco en la cola (línea de espera) y recibe la atención de uno de los servidores.

Un servidor no tiene que ser un solo individuo, puede ser un grupo de personas, es más, ni siquiera tiene que ser una persona o personas, pueden ser máquinas o una pieza de equipo. Siguiendo esta misma idea los clientes tampoco tiene que ser personas, pueden ser maquinas, mensajes de transmisión u objetos.

Mecanismos de Tiempo de Avance. Dada la naturaleza dinámica de estas simulaciones, es importante llevar el control del tiempo, para avanzar de un valor a otro. Se llama *reloj de simulación* a la variable en el modelo que da el valor al tiempo simulado. Esta variable por lo general se inicializa en cero, y va avanzando al tiempo de las ocurrencias de los eventos futuros; puede ser discreto o continuo, dependiendo del tipo de simulación en este caso nos referimos a discretos.

Los componentes de simulaciones de modelos de eventos discretos son:

Estado del sistema. La colección de variables de estado necesarias para describir el sistema en un momento en particular.

Reloj Simulador. Una variable a la que se le da el valor del tiempo simulado. Puede ser absoluto o relativo, la diferencia entre estos dos, depende si se hace una sola simulación o varias en un mismo proceso.

Lista de Eventos. Una lista que contiene los tipos de eventos a ocurrir en la simulación, y sus características.

Contadores Estadísticos. Variables utilizadas para guardar información estadística del comportamiento del sistema. Algunas información estadística podría ser:

- Tiempo de espera promedio
- Tiempo de ocio esperado
- Longitud promedio
- Longitud esperada de las colas no vacías (personas, choches, etc.)
- Número esperado de unidades en el sistema
- Tiempo de espera para una llegada al servicio
- Tiempo esperado de duración de una unidad en el sistema
- Probabilidad de que se tengan *n* unidades en el sistema
- Longitud máxima de cola

Generalmente, para diseñar un programa de un modelo para eventos con tiempos discretos

de simulación se requiere de lo siguiente en terminos de programación:

Programa Principal. Un programa que llama a la rutina de tiempo para determinar el siguiente evento y luego transfiere el control a la rutina de evento correspondiente para poner al corriente al estado del sistema. También puede revisar el final y llama al generador del reporte cuando una simulación ha terminado.

Rutina de Inicialización. Un subprograma que inicializa el modelo de simulación en el momento cero.

Rutina de Tiempo. Un subprograma que determina el siguiente evento de la lista de eventos y luego hace que el reloj simulador avance al siguiente momento.

Rutina de Evento. Un subprograma que actualiza el estado del sistema cuando un tipo de evento en particular ocurre (existe una rutina para cada evento).

Rutinas de Librería. Un conjunto de subprogramas utilizados para generar valores aleatorios de ciertas distribuciones de probabilidad.

Generador de Reportes. Es un subprograma que estima los contadores estadísticos de ciertas medidas de comportamiento y produce uno o varios reporte cuando la simulación termina.

La simulaciones de este tipo se pueden hacer manualmente, sin embargo por la cantidad de información que debe ser procesada y manipulada, es recomendable utilizar una computadora personal. Para hacer un análisis de simulación en base a un modelo de este tipo se recomienda seguir los pasos vistos en la sección II.1. A continuación se dará una introducción a un ejemplo de un modelo de eventos discretos.

II.4 Introducción a un Ejemplo de Simulación

Ahora se analizará una situación práctica de la vida real, enfatizando la planeación de la simulación. Es decir, se verán los primeros pasos de la simulación en este apartado para después en los siguientes capítulos hacer nuevamente referencia al ejemplo y analizarlo de manera más amplia. El tomar este ejemplo para desarrollarlo a lo largo de la tesis, permitirá efectuar comparaciones, y su análisis puede orientar a cualquier empresa similar que preste un servicio a hacerlo más eficas y eficiente.

Formulación del problema

Existe un convenio entre las autoridades mexicanas y el gobierno de Canadá, en el que se les dan facilidades a trabajadores mexicanos para ir a Canadá como agricultores. Para ello es necesario que los trabajadores realicen una serie de trámites entre los que se encuentran la obtención de su pasaporte. Todos estos trámites se llevan ahora a cabo en la Secretaría del

Trabajo y Previsión Social, en una sucursal ubicada en la calle de Lucas Alamán; con el apoyo de las Secretarías de Relaciones Exteriores, Hacienda, Gobernación y Salubridad. Anteriormente, los trabajadores tenían que ir a diferentes oficinas, en distintos sitios a realizar sus trámites; esto les ocasionaba una pérdida de tiempo y dinero pues muchos no conocían la ciudad por venir de otros estados de la República, y el trámite duraba aproximadamente tres meses en llevarse a cabo.

Actualmente, a los trabajadores se les ha simplificado la realización de sus trámites, instalando pequeñas oficinas de las respectivas Secretarías, en un solo lugar. Con estas oficinas los trámites se llevan a cabo con mayor rapidez, dándoles menos molestias a los trabajadores. Con esta ayuda, ahora sus trámites terminan en aproximadamente una semana. Hay que hacer notar que la temporada fuerte de trabajo ocurre durante los meses de junio, julio y agosto, que es cuando el gobierno de Canadá solicita más trabajadores.

El trámite más tardado es el de la obtención del pasaporte. La Secretaría de Relaciones Exteriores tiene a una persona encargada de la recepción de documentos. Para dar un mejor y rápido servicio se ha decidido analizar cómo se lleva a cabo la recepción de este trámite. Esta oficina abre su ventanilla a las 8:00 a.m. y cierra a las 2:00 p.m., es decir, trabaja seis horas continuas. El análisis propuesto se hizo desde las 9:00 a.m. hasta medio día. Se pudo observar que los trabajadores llegaban y si había alguien en la fila esperaban y si no había nadie eran atendidos inmediatamente, es decir, el primero que llegaba era el primero en ser atendido, y no había preferencias.

Siendo el objetivo como ya se mencionó, dar un mejor y rápido servicio a los trabajadores, se deseaba agilizar el proceso de documentación, reduciendo así los costos, de tiempo y dinero de los trabajadores, para ello se considera necesario traer otra recepcionista, darle capacitación y modificar el área de trabajo para que la mayoría de los trabajadores terminen sus trámites a tiempo.

Conceptualización del modelo

Como se puede suponer el problema es tan solo una pequeña parte de todo un sistema. Con base en todo lo anterior se puede conceptualizar al modelo de la siguiente manera:

1. El modelo es de optimización, dado que se desea dar un mejor servicio al menor costo.
2. El modelo será numérico, porque sería demasiado difícil resolverlo analíticamente.
3. El modelo es discreto pues se atiende a personas y estas personas llegan en puntos separados del tiempos.
4. El modelo es dinámico, dado que clientes llegan y se van.
5. El modelo es estocástico porque no se sabe con exactitud cuando llegará un trabajador o cuanto tiempo será necesario para que la recepcionista atienda al trabajador.
6. El ejemplo se puede analizar como una línea de espera.

Obtener y procesar los datos

Primero se debe tomar en cuenta qué datos son los que se necesitan, después, obtenerlos y procesarlos. Como primera propuesta se pueden identificar los siguientes:

- Número de recepcionistas
- Tiempo entre llegadas
- Tiempo de recepción
- Tiempo de espera de los trabajadores para ser atendidos.
- Disciplina, es decir, el primero que llega será el primero en ser atendido
- Horarios de trabajo
- Distribuciones de probabilidad asociadas
- Costos por perdidas de trabajadores
- Costos por utilidad del servicio

En el anexo del Capítulo II se puede ver toda la información recabada durante tres horas continuas de trabajo.

Formulación del modelo matemático

Es importante definir las variables, como pueden ser las variables de estado y las variables aleatorias no controlables. De igual manera, se debe analizar y definir los parámetros, las relaciones causa-efecto, así como un criterio de ejecución.

- *Variable de estado*

La variable de estado es el número de trabajadores presentes en el sistema al minuto t, a la variable la se le puede definir con la siguiente simbología $n(t)$.

Los eventos que modifican este número son:

- Llegue un trabajador, es decir $n(t)$ aumenta en 1.
- Se termine el servicio, es decir, $n(t)$ disminuye en 1.

- *Parámetros*

Los parámetros del modelo son los siguientes.

- Número de recepcionistas: 1
- Tiempo promedio entre llegadas: 4.16 min.
- Tiempo promedio de servicio: 5.26 min.

- *Relaciones Causa-Efecto*

Las relaciones de Causa-Efecto serían de la siguiente manera:

- Para pasar de $n(t_1)=n$ a $n(t_{1+1})=n+1$ debe llegar un trabajador.
- Para pasar de $n(t_1)=n$ a $n(t_{1+1})=n-1$ se requiere de la conclusión de una recepción.
- Un cliente deja de esperar cuando comienza a ser atendido, esto depende de la variable aleatorio del tiempo de servicio.

- *Variables aleatorias no controlables*

Estas variables se pueden definir de la siguiente manera:

- $t(a)$ = tiempo de la siguiente llegada
- $s(t)$ = tiempo de servicio

- *Criterios de Ejecución del Sistema*

Al evaluar los resultados, del número de clientes y/o el tiempo de espera. $n(t)$ se puede mostrar como la gráfica en el anexo.

- $n(t)$: número de clientes en el sistema al tiempo t
- R: número de llegadas
- T: longitud de tiempo observado
- n: número promedio de clientes por unidad de tiempo (minutos)
- w: tiempo promedio de un cliente en el sistema.

El resto de los pasos se analizarán conforme se avance en el trabajo y se verán diferentes modelos utilizando este ejemplo como base.

CAPÍTULO III
TRANSACCIONES Y GPSS/H

III.1 LENGUAJES DE SIMULACIÓN

III.1.1 LENGUAJES DE PROPÓSITO GENERAL

Es posible programar modelos de simulación en lenguajes tales como Fortran, Basic, Pascal, Prolog, Lisp o C, pero para hacer esto se requiere de una gran habilidad. Es importante que el analista se enfoque tanto en la comprensión completa del modelo, como en la del sistema; y quizá no tenga toda su atención en detalles técnicos como los que se involucran a un lenguaje de programación. Los lenguajes de simulación facilitan la programación ya que el analista no necesita ser un experto programador.

Los elementos y estructura de un lenguaje de propósito general no están muy ajustados con los principios de los sistemas que se desea simular, por ejemplo, estos lenguajes no tienen la estructura de datos conveniente para la progresión temporal de un evento, siendo esto básico para los modelos de simulación. En ningún lenguaje de éstos existe un comando que incremente el número de personas u objetos en una fila o que lleve el tiempo de una simulación, para poder incluir estas funciones y otras que son esenciales en la estructura del modelo se requiere de un programa extenso, en el que muchas veces es difícil detectar errores. Los motivos para utilizar lenguajes de simulación son que el tiempo necsario para crear modelos validos es menor, permiten detectar errores más fácilmente y que los reportes estadísticos automáticos que se generan permiten tomar una decisión óptima, entre otros.

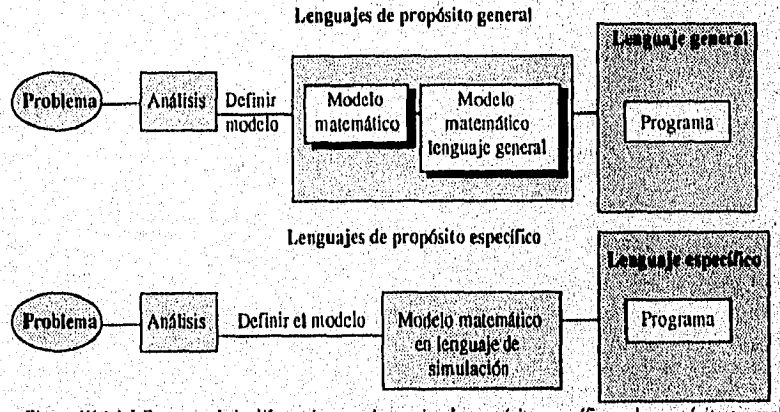


Figura III.1.1.1 Esquema de la diferencia entre lenguajes de propósito específico y de propósito general

Como se puede ver en el la *Figura III.1.1.1*, el analista al exponer el modelo en un lenguaje de propósito general, primero define el modelo matemático y luego el programa de ese modelo, en cambio con los lenguajes de propósito específico al modelar el sistema, se está definiendo el modelo y el programa.

En general no existe un lenguaje óptimo para cada problema, es importante analizar en cada caso cual es el más conveniente. Flexibilidad, es la palabra clave que se utiliza al comparar un lenguaje de propósito general con uno de propósito específico. Al programar con un lenguaje de propósito general se tiene la opción de llamar a una gran variedad de subrutinas, que permiten generar varios tipos de reportes estadísticos, gráficas e inclusive animaciones.

III.1.2 Características de los Lenguajes de Propósito Específico

El objetivo principal de los lenguajes de propósito específico es simplificar la labor de escribir programas de simulación para diversos tipos de modelos y sistemas. Existen actualmente dos clases de paquetes de simulación: los lenguajes y los simuladores. Un lenguaje de simulación es un paquete diseñado para computadora que es muy general y que contiene algunas características importantes de ciertos tipos de aplicaciones. Los modelos pueden ser desarrollados escribiendo un programa en un lenguaje específico, la capacidad principal en la mayoría de los lenguajes es su habilidad para modelar cualquier tipo de sistema, sin importar sus procedimientos de operación ni su lógica.

Los lenguajes de simulación que se encuentran disponibles en el mercado mundial, a la fecha difieren considerablemente por el grado en el que se hace posible su aplicación a tipos particulares de sistemas y el grado en el que pueden suministrar procedimientos de simulación más o menos automáticos.

En cuanto a un simulador, es un paquete computacional que permite simular un sistema sin la necesidad de programar, su ventaja mayor es esta también, pero su desventaja es que se basa únicamente en un sistema, ejemplos de ellos pueden ser los simuladores de vuelo.

Cada lenguaje de simulación tiene diferente visión del comportamiento de los sistemas, sin embargo se ha encontrado que en general tienen tres partes esenciales:

- Identificación del objeto y las características del mismo
- Relaciones entre objetos
- Generación de los objetos

Los objetos pueden ser clientes en un banco o partes de un coche en una línea de montaje. Los lenguajes se refieren a los objetos de diferentes formas, por ejemplo, los lenguajes SIMAN y SIMSCRIPT se refiere a estos como entidades y GPSS y SIMNET como transacciones.

Existen varios elementos de modelos que son utilizados en los paquetes de simulación, entre ellos se encuentra:

- Una *entidad o transacción, (transaction o xact)* la cual llega a un sistema dado, en el que obtiene un servicio de alguna manera y luego se retira. Ejemplos de éstos pueden ser un cliente formado en una fila de una caja de un mercado, una pieza en una fábrica, o un mensaje en un sistema de comunicación.

- Un *atributo o parámetro* es la información que describe o caracteriza a la entidad, ejemplos de estos pueden ser el tipo de mecánica que lleva el cliente en la tienda, el día de entrega de cierta pieza, o el tamaño de un mensaje transmitido.

- Una *fila o línea de espera* es un conjunto de entidades con cierta característica en común, como clientes que esperan que se les cobre por la mercancía que llevan. Las entidades en una fila pueden ser procesadas en forma PLPS (Primero en Llegar, Primero en Salir) o ULPS (Último en Llegar, Primero en Salir) o en base a un valor del atributo de cierta entidad (prioridad).

- Un *recurso o servidor* es una persona o máquina que da cierto servicio a alguna entidad mientras se encuentra en el sistema. Ejemplos de ello puede ser una cajera, una máquina o un nodo en un sistema de comunicación.

Muchas veces es importante analizar el tamaño del modelo en términos de entidades, porque si este es grande, puede ser difícil correr el programa que represente al sistema, dado que se tomaría demasiado tiempo de computadora. Es por esto que el lenguaje de simulación debe contener un generador de objetos y que cuando éstos terminen su labor en el modelo exista la forma de eliminarlos; los distintos lenguajes manejan lo anterior de diferentes formas. Conforme los lenguajes de propósito general han mejorado también los lenguajes de simulación, los cuales han sido codificados en lenguaje ensamblador y muchos más siguen siendo desarrollados en lenguaje Fortran, la estructura de la generación de un objeto y la destrucción del mismo continúan siendo las mismas utilizadas en los lenguajes de simulación antiguos.

Los lenguajes de simulación tienen ciertas características, entre ellas:

Código de Desarrollo del Modelo. En muchos lenguajes todavía es necesario codificar un programa para implementar el modelo, es por ésto que algunos contienen un editor, además con las capacidades gráficas de las computadoras los lenguajes tienen la opción de mostrar gráficas de los resultados estadísticos.

Corrección del Modelo. Una vez que el modelo ha sido codificado en el lenguaje de simulación, el siguiente paso es encontrar los errores del mismo. Los errores de sintaxis son los primeros en salir al estar revisando la codificación y para diagnosticar estos errores la mayoría de los lenguajes tienen implementado un corrector. Los errores más difíciles de

corregir son los que ocurren en la lógica de la codificación, el localizar este tipo de falla todavía no está bien diseñado en la mayoría de lenguajes de simulación. Al encontrar un error de sintaxis o ejecución el programa generalmente se da por terminado, sin dar una razón en términos lógicos del modelo, para estos casos muchos lenguajes permiten hacer un rastreo al momento de correr el programa.

Generación de variaciones aleatorias. Para cualquier simulación probabilística, la habilidad de extraer muestras aleatorias de una distribución de probabilidad específica es muy importante, esta labor es facilitada en un lenguaje de simulación. En general cada variable aleatoria del sistema necesita ser modelada con una distribución de probabilidad y no tan solo utilizando su media, por esta razón es muy importante que el lenguaje de simulación contenga la forma de obtener muestras de algunas distribuciones de probabilidad, así como muestras basadas en la observación de un sistema.

Generación de estadísticas. La ejecución de un modelo de simulación sin la generación adecuada de estadísticas, puede causar que se llegue a una mala interpretación de un sistema. Un lenguaje de simulación debe permitir la generación de estadísticas, durante y después de la ejecución del modelo, y para mejorar la interpretación de los resultados, pueden incluirse gráficos en los paquetes de simulación. Los lenguajes de simulación contienen ciertos comandos para hacer réplicas independientes de un modelo, con cada una se pueden utilizar diferentes números aleatorios, empezando en el mismo estado inicial y guardando el valor cero en los contadores estadísticos, siendo esto parte importante de un estudio de simulación, la generación de varias muestras de diseños alternos del sistema da un mejor panorama de la solución optima del problema, así como permite con el uso de estas muestras hacer un análisis estadístico sobre los resultados.

Diseño de experimentos. Dado que el análisis de simulación es descriptivo, su aplicación y éxito depende de la experimentación del modelo. Los diseños experimentales deben ser efectivos y eficientes para obtener una solución con calidad. Un lenguaje de simulación puede facilitar esta labor, permitiendo hacer cambios al modelo.

Animación y resultados dinámicos. Los lenguajes de simulación en computadoras permiten que la capacidad de gráficos este presente para ilustrar de alguna forma la corrida y las salidas de la simulación. La ilustración de objetos se refiere a la animación. En la animación usualmente se necesita de un monitor a color y de algún símbolo que represente los objetos y los elementos del modelo. Es importante mencionar que el uso de animación por lo general hace que las corridas de las simulaciones sean más lentas, por esta razón es que es recomendable utilizar este tipo de paquetes en computadoras con mayor potencia. Más adelante se tratará este tema con mayor énfasis sobre el paquete *Proof Animator*, este permite hacer animaciones en base a un programa generado de un lenguaje de simulación.

Existen algunas características importantes que los paquetes deben contener, como: un subsistema de entradas, la facilidad para recolectar las estadísticas, la facilidad para definir diferentes diseños, un subsistema de salidas, un editor de programas y un compilador.

Las características necesarias para programar modelos de eventos discretos deben ser entre otras las siguientes:

- Generación de números aleatorios de la distribución de probabilidad uniforme U(0,1)
- Generación de números aleatorios de alguna distribución de probabilidad como la exponencial, normal, etc.
- Generación de tiempos de avance simulados
- Determinación del siguiente evento de la lista de eventos y posibilidad de permitir pasar el control al bloque apropiado.
- Posibilidad de agregar o eliminar datos de cierta lista.
- Posibilidad de coleccionar y analizar datos
- Generación de reporte de resultados
- Detección de condiciones de error

La mayoría de los lenguajes de simulación para modelar sistemas discretos utilizan las siguiente aproximaciones:

Aproximación por programación. Un sistema es modelado identificando los eventos característicos del modelo y luego se escriben un conjunto de rutinas de eventos que permiten dar una descripción detallada de los cambios de estado que ocurren en el momento de que cada evento sucede. La simulación transcurre ejecutando los eventos en orden ascendiente de acuerdo al momento de su ocurrencia. Este tipo de aproximación está presente en los siguientes lenguajes: SIMSCRIPT y SLAM.

Aproximación por orientación a objetos o entidades. Un proceso es una secuencia de tiempo de eventos interrelacionados, separados por pasajes de tiempo, que siguen un orden y que describen la situación completa de una entidad conforme fluye por el tiempo. Un proceso puede corresponder a una entidad que llega a una cola y que es atendida por un servidor. Los lenguajes más usuales que utilizan este tipo de aproximación son: GPSS/H, SIMAN, SIMSCRIPT II.5 y SLAM II.

Estos dos tipos de aproximación son muy parecidos, dado que utilizan entre otras cosas un reloj simulador, una lista de eventos y una rutina de tiempo; pero su principal diferencia es la construcción del programa que permite modelar un sistema. Los comandos utilizados en los lenguajes que se basan en una aproximación por programación con poca facilidad se traducen al lenguaje cotidiano. Existen algunas ventajas de los lenguajes que usan orientación a objetos, una de ellas es que los comandos son más naturales, dado que una rutina de proceso describe la experiencia completa de la entidad en proceso; otra es que en general se requiere menor número de líneas al programar el modelo, y la desventaja mayor es su poca flexibilidad.

Es importante que los paquetes de simulación contengan:

- Flexibilidad de modelaje, por lo general no existen modelos iguales.

- Facilidad para desarrollar el modelo, esto es importante pues en muchos proyectos que contienen simulación y que requieren de un lenguaje de simulación, los resultados son requeridos con cierta rapidez.
- Rapidez de ejecución del modelo, esto es importante para ciertos modelos de gran tamaño.
- Un tamaño máximo de líneas de un modelo, esto quizá es importante cuando el modelo es ejecutado en una microcomputadora.
- Disponibilidad para diferentes tipos de computadoras, entre ellas pueden ser microcomputadoras, estaciones de trabajo o mainframes.
- Capacidad para simulaciones combinadas, tanto discretas como continuas.

Los lenguajes de simulación tienen en general los siguientes objetivos:

- Producir una estructura generalizada para el diseño de modelos de simulación.
- Proporcionar una forma rápida para la conversión de un modelo de simulación a un programa de computadora.
- Proveer una forma rápida para la introducción de cambios en el modelo de simulación, que puedan reflejarse fácilmente en el programa de máquina.
- Proporcionar una forma flexible para la obtención de reportes de salida, que sean útiles cuando se sujetan a su análisis correspondiente.

III.1.3 Ventajas y Desventajas de los Lenguajes de Simulación

Existen algunas ventajas importantes de los lenguajes de simulación con respecto a los lenguajes de propósito general, entre ellas se encuentran las siguientes:

- Facilitan el modelo del sistema, dado que automáticamente contienen las características necesarias para modelos de simulación, permitiendo que el tiempo de programación sea menor. Su construcción básica en forma de bloques está más relacionada con el análisis. Los programas de los modelos de simulación son más fáciles de cambiar cuando están escritos en algún lenguaje de simulación.
- Movimiento automático en el tiempo (reloj).
- Muchos tienen la opción de obtener automáticamente estadísticas, dado que contienen un lugar de almacenamiento dinámico.
- Generación automática de variables aleatorias con distintas distribuciones de probabilidad
- Permiten tener un diagnóstico de errores eficiente, pues muchos errores típicos son detectados y analizados automáticamente. También, dada la estructura de los lenguajes, por lo general se requiere menor número de líneas al programar el modelo, haciendo menos posible cometer errores.
- Generación de diagramas de bloques.
- Generación de animaciones gráficas (esto existe en algunos lenguajes).

Algunas desventajas importantes son:

- Por lo general el analista conoce algún lenguaje de propósito general, pero no un lenguaje de simulación.
- Es mucho más fácil conseguir un lenguaje de propósito general, que uno de propósito específico.
- Un programa escrito eficientemente en un lenguaje de propósito general requiere menos tiempo de ejecución que uno escrito en algún lenguaje de propósito específico, ocurre esto porque los lenguajes de simulación están diseñados para modelar una gran variedad de sistemas, pero con la existencia de computadoras de mayor capacidad, esta desventaja tiene menor importancia.
- Los lenguajes de propósito general ofrecen una máxima flexibilidad para el programador, en cuanto al diseño y formulación de los modelos matemáticos para el sistema bajo estudio, al tipo y formato de los reportes de salida que se generan y a la clase de experimentos de simulación que se realizan con el modelo. No obstante la dificultad que se experimenta al escribir programas de simulación utilizando un lenguaje de programación de propósito general, constituye la principal desventaja de estos lenguajes.
- Los lenguajes de simulación son relativamente más caros que otros tipos de software.

III.1.4 Algunos Lenguajes de Simulación

III.1.4.1 SIMAN

SIMAN (SIMulation ANalysis) es un lenguaje de simulación que es utilizado para analizar modelos de eventos discretos, continuos o ambos. Este paquete fue desarrollado por Dennis Pegden en 1982. Cuando salió la primera versión de este lenguaje ganó la aceptación en el mundo pues fue el primero que estaba disponible para microcomputadoras. Este paquete utiliza un conjunto de diagramas en bloques diferentes, teniendo cada bloque una función específica. El lenguaje supone que las entidades fluyan a través de estos bloques, siendo estos los componentes del sistema. SIMAN reconoce y separa las fases más importantes de un análisis de simulación: la definición del modelo, su experimentación y el análisis de los resultados. Permite así analizar varios sistemas alternos sin modificar el programa del modelo de computadora, tiene varios bloques especiales para modelar modelos de tratamiento de materiales. Los bloques fundamentales de SIMAN son: CREATE, SEIZE, RELEASE, DELAY, QUEUE y TALLY, cada uno de los anteriores tiene un símbolo especial. *Cinema* es una versión mejorada del SIMAN que tiene capacidad para producir animaciones. Este lenguaje contiene un procesador de resultados que permite calcular ciertas estadísticas tales como intervalos de confianza o pruebas de hipótesis en los resultados.

III.1.4.2 SIMSCRIPT

Después de la aparición del GPSS, la Corporación RAND, bajo la tutela de Harry Markowirtz en 1962, desarrolló un lenguaje de simulación llamado SIMSCRIPT. Este lenguaje contiene una visión más general del mundo y permite el modelaje de sistemas más complejos que no son del tipo de línea de espera. Para hacer esto los elementos del lenguaje son menos parecidos con los elementos del mundo real. Este es un lenguaje de programación general que tiene la capacidad de desarrollar modelos de simulación discreta, continua o combinada. El uso de conjuntos, eventos, procesos y recursos es la base del lenguaje, es por esto que el analista requiere de mayor información sobre la estructura del modelo, necesitando así mayor tiempo de entrenamiento para ser eficiente en la programación del lenguaje.

La versión más reciente es la SIMSCRIPT II.5 y pertenece a CACI, el lenguaje es utilizado aproximadamente por más de 4000 empresas alrededor del mundo, entre ellas se encuentran Air Canada, IBM, Autoridades del Puerto de Singapur, NASA, FAA, AT&T, Hexale, UNISYS, entre otros. Contiene un ambiente de programación interactiva, un corrector simbólico, una base de datos simulados con interfase SQL y un paquete completo con gráficas de resultados. Este paquete es utilizado para modelos militares de combate.

Este lenguaje es más de propósito general y en contraste con los demás lenguajes, permite hacer cualquier tipo de estructura de datos, declarando variables de tipo entero, real o texto, incluyendo también subrutinas hechas por el analista, así como funciones. No existe un formato dado para los comandos de SIMCRIPT. Por el tipo de sintaxis utilizada en el lenguaje, siendo menos obvia que los demás lenguajes de simulación, muchos posibles usuarios del lenguaje se desaniman, pero vale la pena aprender el lenguaje, ya que es un recurso de modelaje extremadamente poderoso. Es importante mencionar que el paquete contiene varias funciones estadísticas y matemáticas, pero no contiene una colección automática de datos y no despliega estadísticas. Para hacer esto el usuario debe especificarlo en cierta parte del programa.

SIMSCRIPT se basa en una descripción del sistema y para ello tiene que emplear los conceptos de entidad, atributo, conjunto, estado y evento. Fue implementado en un procesador FORTRAN.

III.1.4.3 SIMNET

El SIMNET permite hacer una modelación a detalle, con flexibilidad y simplicidad, es completamente interactivo, está basado en redes con únicamente 4 nodos. Tiene asignaciones especiales para controlar el flujo de población en la red, así como de modo interactivo permite el análisis de transitoriedad y estabilidad del sistema, no requiere de rutinas externas en Fortran u otro lenguaje. Contiene una simple lógica *If then else end if*, así como una poderosa manipulación de archivos dentro de la red. Permite utilizar

procedimientos para modelar secuencias repetitivas, en serie o en paralelo. El sistema de rastreo se encuentra en modo interactivo o proceso por lote, y localiza y explica errores de una manera sencilla. Tiene un fácil acceso a datos de entrada desde archivos externos, contiene reportes comprensibles en forma estándar y en forma diseñada por el modelador, las estadísticas globales son hechas durante la ejecución del programa.

Este lenguaje fue diseñado con el objetivo de ayudar en el diseño de fábricas, manufactura flexible, análisis de capacidad y cuellos de botella, flujos de materiales y control de inventarios, programación de órdenes de trabajo, diseño de instalaciones de servicio, análisis de recursos, análisis del desempeño humano y en sistemas de mantenimiento y confiabilidad.

III.1.4.4 GPSS

Fue uno de los primeros lenguajes de simulación, el GPSS (General Purpose Simulation System) fue creado por Geoffrey Gordon, siendo publicado por primera vez en 1961. Existen varias versiones de este lenguaje, entre ellas se encuentra el GPSS/H; tiene la característica de permitir simular modelos de líneas de espera. En las décadas de los 60 y los 70, el GPSS fue muy popular por el tipo de sistemas que modelaba, las líneas de espera. Los elementos del lenguaje son muy parecidos con los elementos de un sistema discreto. Se caracteriza por su proceso de interacción o flujo de transacción. Los bloques fundamentales de este lenguaje son: GENERATE, TERMINATE, SEIZE, RELEASE, ADVANCE, QUEUE y DEPART. Los comandos de control necesarios para que un simple programa corra son: SIMULATE, START y END. Estos comandos y bloques se analizarán posteriormente con base en lenguaje GPSS/H versión 2.0 para estudiantes.

III.1.4.4.1 GPSS/H

El GPSS/H fue desarrollado por James Henriksen en 1977 y se distribuye por Wolverine Software. Es un lenguaje de compilación; comparado con otras versiones del GPSS, como es el GPSS V corre cinco veces más rápido. Los problemas a que se orienta el GPSS/H son de tipo de líneas de espera, sin embargo, debe notarse que muchos problemas reales siguen este comportamiento: inventarios, procesos de manufacturación, sistemas de salud, redes de comunicaciones, sistemas de transportes, sistemas de defensa, sistemas de líneas, entre otros. Este lenguaje es una opción buena al trabajar con modelos de unidades de tráfico que compiten entre ellos para un uso adecuado de recursos. Un ejemplo de ello puede ser que en un proceso de manufactura.

El GPSS/H versión 2.0 para estudiantes es utilizado para simulaciones de eventos discretos, es decir, de eventos con tiempos discretos. Esta versión del GPSS contiene más de 60 comandos. Algunas características que este paquete contiene y que versiones anteriores no, son por ejemplo un reloj de valor real, así como la habilidad para leer y escribir en archivos externos, reportes de salida *eficientes*, mejores comandos de control, funciones matemáticas y un ilimitado número de rutinas para generar valores de distribuciones de probabilidad. Por

estas capacidades y la naturaleza básica del GPSS en general, la mayoría de los modelos del GPSS/H no requieren del uso de rutinas externas, sin embargo existe la opción de hacer uso de rutinas externas hechas en Fortran. Además este paquete se encuentra disponible en el Centro de Cómputo de la ENEP Acatlán.

El mercado estadounidense ya salió la nueva versión del GPSS/H la cual permite generar muestras de 23 distribuciones de probabilidad, la versión anterior solo permite tener acceso a cinco diferentes distribuciones, así como hacer uso de rutinas externas elaboradas en C y Fortran.

En este lenguaje los clientes o entidades que requieren servicio de algún tipo en el sistema son llamados *transacciones* y sus atributos son llamados *parámetros*. Los servidores o los recursos que dan ese servicio requerido por las transacciones son llamadas *facilities* o *storages*, que corresponden a un servidor o a un grupo de servidores en paralelo.

Los modelos de este paquete expresan las reglas que gobiernan la operación de un sistema y no formulan y resuelven un conjunto de ecuaciones; los modelos toman forma en series de comandos. Los resultados de simular estos modelos están dados por la información que describe el estado actual del modelo como transcurre el tiempo y/o detallando el conjunto de estados en forma ordenada en el tiempo conforme el modelo llega a su estado presente, y /o en forma resumida del comportamiento al final de una simulación. Los modelos en GPSS/H no proporcionan automáticamente una representación gráfica de la situación a ser modelada. En base al uso de gráficos y animaciones, estos modelos pueden ser visualizados objetivamente. Los elementos de estos modelos pueden contener comportamiento aleatorio, siendo estadísticamente predecibles.

III.1.5 Otros Lenguajes de Simulación

Algunos lenguajes que conviene conocer, además de los anteriores para hacer una elección correcta son:

20SIM. Es un lenguaje que sirve para modelos de sistemas electrónicos, mecánicos e hidráulicos. Permite simular con modelos discretos o continuos, pertenece a la Universidad de Twente, y aproximadamente cuesta 800 dólares para académicos y 2300 para corporativos, su dirección electrónica es www.rt.el.utwente.nl/20sim/product/20sim.htm

Arena. Sirve para modelos industriales, electrónicos, textiles y de alimentación, permite hacer simulaciones discretas o continuas, esta hecho por Systems Modeling Corporation y su dirección electrónica es www.sm.com./arena.htm

Archilles, the factory model. El propósito de este paquete es modelar, simular y proveer una herramienta detallada para análisis y diseños de fabricas u operaciones de manufactura. Tiene capacidades de estudios de planeación, identificación de cuellos de botella y producciones mixtas.

ACSL (Advanced Continuos Simulation Language). Este lenguaje fue desarrollado para modelar sistemas continuos descritos con tiempos dependientes, ecuaciones diferenciales no lineales y funciones de transferencia. Estos modelos tienen diferentes aplicaciones en el mundo real como en el aeroespacio, biológico, químico, ambiental, farmacéutico, energético y electrónico.

C++SIM. Es utilizado para simular modelos discretos, pertenece a la Universidad de Newcastle y esta basado en el lenguaje C y se encuentra en Internet en la siguiente dirección: ulgham.ncl.ac.uk/C++SIM/homepage.html

COMNET III. Pertenece a la CACI Products Company, permite hacer simulaciones discretas de operaciones de redes LAN, MAN y WAM. Este paquete cuesta aproximadamente unos 900 dólares a académicas y 2600 a corporativos, su dirección en Internet es: www.caciasl.com/comnet.html.

CREATE!. Permite hacer simulaciones discretas de modelos logísticos y de producción, pertenece a la Universidad de Magdeburg, Alemania y se encuentra en la siguiente dirección: simsrv.cs.uni-magdeburg.de/~create.

PROPHESY. Sirve para simulaciones discretas de redes y flujos, es de bajos costos y se encuentra en la siguiente dirección: www.esn.net/abstraction/ Pertenece a Abstraction Software.

SIMEX 3.0. Pertenece a la Universidad de Minnesota, hace simulaciones discretas, se encuentra disponible gratuitamente en la Web, esta basado en el lenguaje de propósito general C, se encuentra en la siguiente dirección:
www.nmsr.labmed.umn.edu/nmsr/readme.htlm

TAYLOR II. Permite hacer simulaciones discretas, pertenece a F & H Simulations cuesta aproximadamente 15,500 dólares y se encuentra en la siguiente dirección:
www.xmision.com:80/~ taylor

VisSim. Permite hacer simulaciones continuas, cuesta 17,000 dólares y pertenece a Visual Solutions, se encuentra en la siguiente dirección: www.ultranet.com/biz/vissim/

MODSIM. Es utilizado en el área militar y fue originalmente desarrollado por la armada de los Estados Unidos, esta orientado a objetos y su dirección electrónica es
ftp://max.cecer.army.mil/ftp/isle.

Must. Es un software basado en Pascal para simulaciones de procesos discretos, es utilizado en gran variedad de organizaciones industriales, que incluyen el transporte, la manufactura y el análisis de colas, entre otros. Tiene ayudas en rutinas estadísticas, rutinas para hacer filas, rutinas de entrada y salida, así como para comandos de procesos. La documentación es en Inglés, y su costo es de 5000 dólares.

Gasp. Fue desarrollado por Philip J. Kiviat. Las principales características son su independencia de máquina y sus características modulares, las cuales facilitan el desarrollo y alteración de los programas de simulación para ajustarse a las necesidades de un sistema dado, fue escrito en FORTRAN.

DYNAMO. Es un lenguaje de simulación de propósitos especiales, fue diseñado con el objetivo de simular ciertos tipos de sistema dinámicos de información y retroalimentación, que se pueden describir en términos de un conjunto de ecuaciones de diferencias finitas. Es utilizado para simular modelos econométricos, aunque también se puede utilizar para modelar sistemas biológicos, físicos y sociales. Existe la posibilidad de simular problemas de líneas de espera y de planeación DYNAMO, sin embargo el utilizar el paquete en este tipo de problemas puede ser costo en términos de tiempos de programación.

ALSS II. Es un simulador de sistemas de líneas de montaje, el cual permite desarrollar, simular modelos de este tipo, esta basado en SIMAN.

BEST-NETWORK. Son una familia de simulaciones que analizan el fluido de información en sistemas de comunicación, no se requiere de una programción extensa.

CETRAN. Es un simulador de plantas nucleares, permite construir completamente modelos de estas plantas, al configurar y unir bloques que representan equipo interactivamente.

GSS (General Simulation System). Permite simular sistemas dinámicos, utiliza un lenguaje de modelaje de alto nivel, el usuario tiene la facilidad de definir modelos y modificarlos sin mayor esfuerzo, en este paquete se pueden hacer simulaciones de grandes tamaños en escala del tiempo.

SIMPAC. Fue desarrollado por M. R. Ñackner y J. Kagdis. Consta de cuatro componentes básicos: actividades, transacciones, colas y recursos operacionales, es un lenguaje con incrementos fijos de tiempo, una de sus características principales es el rango flexible de reportes de salida, sin embargo es más difícil de aprender que el GPSS. Se desarrolló para manipular problemas de líneas de espera y de planeación .

QSNAP 2 (Queuing Network Analysis Package 2). Es un lenguaje orientado a objetos y algoritmos que permiten el diseño de modelos complejos de alto nivel, y su análisis utiliza teoremas de teoría de colas para dar soluciones a modelos. Tiene rutinas de control de archivos y letras, contiene gráficas y un interactivo corrector. Permite analizar cadenas de Markov.

SIMEX. Este software es utilizadb para la simulación de micropoblaciones, hace énfasis en poblaciones humanas y la propagación de enfermedades infecciosas y crónicas, contiene algunos ejemplos de modelos de líneas de espera. El autor de es Michael Altmann con la siguiente dirección electrónica: http://www.nmsr.labmed.umn.edu

Stella. Es bastante fácil de utilizar, el usuario dibuja el modelo en términos de flechas y rectángulos. Su dirección electrónica es http://www.valley.net/~hps/hps.html

Como se pudo estudiar el analista tiene muchas opciones del tipo de lenguaje a utilizar, existen lenguajes para una gran variedad de problemas, así sean discretos o continuos, es por esto importante estudiar detalladamente que tipo de sistema se esta profundizando para poder determinar cuál es el mejor lenguaje a utilizar, a continuación se hará un breve análisis sobre como seleccionar el mejor lenguaje.

III.1.6 SELECCIÓN DE UN LENGUAJE DE SIMULACIÓN

El seleccionar un lenguaje de simulación es análogo a elegir el modelo de simulación correcto para cierto tipo de problema. La selección depende mucho de las circunstancias del analista, así como del tipo de problema y del costo del lenguaje, sin embargo, si se puede contestar a las siguientes preguntas entonces se garantizará una mejor selección:

- ¿Existe un manual de usuario disponible?
- ¿Es compatible con la computadora existente?
- ¿Tiene documentación suficiente y diagnósticos de los errores?
- ¿Los tiempos de: organización, programación, compilación y ejecución son adecuados?
- ¿Cuáles son los costos?
- ¿Es conocido o fácil de aprender?
- ¿Es compatible con otro software?
- ¿Para que tipo de problemas se aplica?
- ¿Cuáles es su flexibilidad en los reportes de salida?

En general, el determinar cual es el lenguaje de simulación más adecuado para un estudio en particular depende de la naturaleza del sistema y la habilidad para programar que tenga el individuo que lleva a cabo la investigación. Como regla general, se requiere una mayor comprensión del procedimiento de programación para obtener un incremento en la flexibilidad de un programa de simulación.

Por otro lado es importante tomar en cuenta lo siguiente:

Facilidad de apreudendizaje. Muchos lenguajes son más difíciles de aprender que otros y quizás se enfocan a ciertos sistemas. Mientras más parecidos sean al lenguaje natural, más fáciles serán para aprender.

Facilidad para ser explicados a individuos sin experiencia. Esto es muy parecido a la facilidad para aprender el lenguaje. Es importante dado que muchas veces a quien se debe explicar el modelo es a la persona que esta pagando y quizá no esté muy interesada en el lenguaje, es por esta razón que se debe analizar a quien se está tratando de explicar.

Costo. Este costo debe ser incluido al costo total del análisis. Es importante analizar si un lenguaje no muy costoso puede ser tan útil como uno caro (análisis de costo-beneficio).

Análisis del tipo de problemas que se pueden resolver por el paquete. Para ser lo suficientemente útil, un lenguaje de simulación debe modelar varios tipos de sistemas: administración de inventarios, lugares de trabajo, sistemas de tratamiento de materiales y redes de comunicación, entre otros.

III.2 ANIMACIÓN CON PROOF ANIMATOR

La animación fácil de usar es una de las razones principales por las cuales la popularidad de los modelos de simulación ha crecido. En una animación, los elementos claves de un sistema, es decir, máquinas, partes de coches, clientes, cajeras, mensajes, etc son representados con íconos que cambian de tamaño, de forma, de color o de posición cuando hay un cambio de estado; el sistema se puede mostrar gráficamente durante un lapso de tiempo y observar como va cambiando durante ese tiempo. El paquete Proof Animator despliega la simulación después de que ha sido ejecutada; el lenguaje de simulación genera un programa en base a comandos de salida que después se corre en el Proof Animator, es por esto que tanto GPSS/H como SIMNET o cualquier otro lenguaje permite hacer animaciones dado que contienen comandos de salida. Para generar una animación se requieren de dos archivos: uno con extensión *lay* y otro con extensión *atf*. El archivo de extensión *atf*, es el programa generado por el lenguaje de simulación y el de extensión *lay*, contiene información sobre el fondo de la simulación y el tipo de objeto utilizado, siendo estos los clientes, las partes de un automóvil, etc. Para mayor informes sobre el uso de este paquete de animación buscar el libro *Using Proof Animation*, su bibliografía se encuentra al final de este trabajo.

La razón más importante de la popularidad de la animación es su facilidad para comunicar la esencia del modelo de simulación a administradores o personal con capacidad de decidir, incrementando así la credibilidad del modelo. Otros beneficios de la animación pueden ser:

- Corregir un programa de simulación.
- Mostrar que el modelo de simulación no es válido.
- Sugerir mejoras en los procedimientos operacionales o de control lógica para un sistema.
- Entender el comportamiento dinámico del sistema.

Existen ciertas desventajas de las animaciones, entre ellas se pueden mencionar a las siguientes:

- No son un sustituto para un análisis cuidadoso estadístico de los resultados de la simulación.
- No se pude concluir que un sistema está bien definido con tan solo observar una animación durante un periodo de tiempo.
- Utilizar la animación incrementa el tiempo de desarrollo de una simulación y

muchas veces los paquetes de simulación que contienen animación son demasiado caros.

- Además de que tan solo cierta lógica del modelo de simulación puede ser observada con la animación.

Las características siguientes son algunas de las deseables en un paquete de animación:

- Dado que la animación es una herramienta de comunicación, es importante que se vea realista. Por tal razón es importante que el usuario tenga la posibilidad de crear objetos y fondos de buena calidad.
- Es también deseable que estos íconos no brinquen en la pantalla o con movimientos bruscos.
- De igual manera es importante considerar la posibilidad de almacenar los objetos y los fondos para un uso posterior en otros modelos; además deben existir algunos íconos y fondos estándares para facilitar el desarrollo de la animación.
- La animación debe ser fácil de desarrollar, basándose más en gráficos que en programación.

El paquete Proof Animator contiene en general estas caracterfsticas, sus animaciones no son cien por ciento realistas, pero dan una idea de cómo funciona el sistema. Existen algunos comando de salida en el GPSS/H como BPUTPIC y PUTPIC los cuales permiten generar un archivo que contenga la información que uno desea en él. La información se ordena de cierta forma y al momento de ejecutar el modelo, el lenguaje almacena un archivo el cual puede ser corrido en el Proof Animator. Un ejemplo puede ser el *Programa III.2.1*, muestra un sistema de una éstación de lavacoches.

```
            SIMULATE
ATF     FILEDEF 'LAVACO.ATF' (Se generá un archivo LAVACO con extensión ATF)
            GENERATE 10,2
            BPUTPIC FILE=ATF,LINES=3,(AC1,XID1,XID1) (En el archivo aparecerán las tres líneas sig.)
TIME *,**
CREATE COCHE *
PLACE * ON ENTRY
* Los coches llegarán en ciertos momentos, los asteriscos son los valores de las variables AC1 y XID1
            ADVANCE 0
            SEIZE LAVAR
            BPUTPIC FILE=ATF,LINES=2, (AC1,XID1)
TIME *,**
PLACE * ON LAVAR
            ADVANCE 5,2
            RELEASE LAVAR
            BPUT FILE=ATF,LINES=2,(AC1,XID1)
TIME *,**
PLACE * ON SALIDA
            ADVANCE 0,25
            BPUT FILE=ATF,LINES=2,(AC1,XID1)
TIME *,**
DESTROY *
            TERMINATE 1
            START 100
            PUTPIC FILE=ATF,LINES=2,(AC1)
TIME *,**
ENJI
            END
```

Programa III.2.1 Programa en GPSS/H que generá un archivo para ser ejecutado en Proof Animation

Al correr el modelo anterior en GPSS/H se generará un archivo el cual incluirá la información sobre la simulación que fue obtenida a lo largo de la misma, esta información consta de lo que aparece en el programa después de PUTPIC y BPUT. Al ejecutar el archivo *Lavaco.atf* se tendrá que utilizar algún archivo que contenga el fondo de la animación así como un icono identificado con Coche .

III.3 TRANSACCIONES

En el GPSS/H la estructura del sistema que se va a simular se describe en forma de diagramas de bloques, los cuales se dibujan de acuerdo con un conjunto fijo dado, de tipos de bloques, como se ver en la *Figura III.3.1*, muestra un sistema simple de una sola línea de espera con un servidor. Cada tipo de bloque representa una acción específica la cual es característica de alguna operación básica que ocurre en un sistema. Las conexiones entre los bloques del diagrama indican la secuencia de las acciones que ocurren en el sistema.

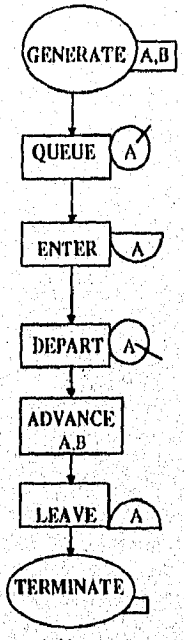


Figura III.3.1 Diagrama de bloques de una línea de un solo servidor

Un modelo de GPSS/H puede ser expresado en forma de diagrama de bloques o con el comando equivalente en este esquema. Un diagrama de bloques es una colección de figuras con un camino que las conecta (*Figura III.3.1*). Cada estructura juega un papel importante en el modelo.

Las unidades de tráfico se mueven a través de los diagramas de bloque se llaman transacciones (*Xact*) en GPSS/H, un ejemplo son las personas que esperan a ser atendidos por una cajera. El movimiento de las transacciones de bloque a bloque conforme una simulación se lleva a cabo, forma la parte vital de una representación hecha en GPSS/H.

La construcción de un modelo se puede hacer al seleccionar los bloques apropiados, ponerles orden y estructurarlos en forma de un diagrama para crear una correspondencia entre lo programado y el esquema que representa al sistema real, este diagrama no necesariamente se debe diseñar para hacer un estudio de simulación utilizando el GPSS/H. Los bloques pueden ser utilizados para representar recursos o servidores del sistema, la forma de recolectar información y las capacidades de toma de decisión. Los aspectos físicos y lógicos del sistema modelado y el tipo de información qué la representación debe obtener, determinan que tipos de bloques es conveniente utilizar al construir el modelo.

Este lenguaje interpreta a los sistemas en forma de unidades de tráfico que se mueven de cierta manera, de bloque a bloque, conforme la simulación avanza. Cada bloque representa una acción a ser llevada a cabo siempre que éste sea ejecutado. En forma general, el movimiento de una transacción a un bloque permite que este último sea ejecutado.

Las transacciones simulan a las unidades de tráfico de un sistema; ejemplos de ellas pueden ser: un sistema de control aéreo, donde las unidades de tráfico son los aviones que se mueven en la secuencia de las zonas de control, conforme el avión viaja de un aeropuerto a otro; en un sistema hospitalario, las transacciones son las personas que llegan a la sala de emergencias; y en el ejemplo de la oficina de pasaportes, éstas son los trabajadores que desean ir al Canadá que son atendidos por la recepcionista.

Es evidente que las transacciones pueden tener un rango de posibles significados en un modelo, en muchos casos éstas representan personas y en otras pueden ser objetos, o en otras situaciones ninguno de los dos. Dado que muchas veces una transacción representa personas que se mueven de un lado a otro, pueden ser imaginadas como personas con movimiento.

Se debe enfatizar que pueden haber diferentes y variadas transacciones, dentro de un modelo en un solo instante. Cada persona u objeto del sistema puede ser identificado con un número, cada transacción en un modelo tiene un número de identificación único, representada por un número entero positivo.

LA NATURALEZA DEL MOVIMIENTO DE LAS TRANSACCIONES EN UN MODELO

Cuando una simulación da inicio no existen transacciones en el modelo, conforme la representación se lleva a cabo las transacciones son creadas e introducidas en cierto instante, de acuerdo a la lógica del modelo, lo mismo sucede cuando una transacción abandona el esquema, es decir, cuando es destruida en algún lapso de la estructura simulada. El número de transacciones varía durante el curso de una simulación.

Las transacciones dependen en gran medida de los bloques, conformando así un modelo. En general, cada transacción posee un bloque actual (*current block*), donde se sitúa en un momento dado y un bloque siguiente al que pretende avanzar. Entre el tiempo de su creación y destrucción, una transacción lleva un ciclo de vida, que usualmente consume tiempo simulado, durante este ciclo pasa por diferentes bloques.

Durante el experimento pueden existir varias transacciones, pero sólo una se mueve en un momento dado. Cuando llega el turno de una transacción para tratar de avanzar de posición actual (*current block*) al siguiente bloque, (si la acción puede ser lograda) la transacción así lo hace y ese estado se convierte en su bloque actual, llevando a cabo ahora la acción correspondiente a esa ubicación. Esta forma de actuar de las transacciones continúa hasta que alguna de las siguientes condiciones la forza a dejar de avanzar por un momento si:

- La transacción entra a cierto bloque que tiene por objetivo mantenerla allí por durante cierto intervalo de tiempo simulado.
- El siguiente bloque de una transacción no permite su entrada a él, si este es el caso la transacción permanece en el bloque actual. Después, está transacción intentará entrar al siguiente estado, logrando, dependiendo del modelo, con éxito su entrada. Existen cinco tipos de circunstancias a las que se les permite bloquear una transacción, en los siguientes capítulos se mencionará algo sobre estos bloques.
- La transacción entra a un bloque que la destruye, si esto sucede la transacción sale permanentemente del modelo.

Cuando una transacción ha sido forzada a detenerse, entonces será el turno de otra para salir de un bloque actual y entrar al siguiente, hasta que ocurra alguna de las condiciones anteriores. Existen, como se ha hecho referencia a ellos sin definirlos, los bloques de creación y destrucción, el primero sirve para crear e introducir una transacción a un modelo y el segundo para sacarla permanentemente del modelo.

EL RELOJ SIMULADO

En un sistema real, cuando ciertos eventos ocurren inevitablemente y aunque suene obvio, el tiempo transcurre, por esta causa el GPSS/H contiene un reloj simulado que es usado para describir y guardar el desarrollo de una simulación. Este reloj registra valores en forma de números reales positivos, siendo el más pequeño de 0.0, ejemplos de estos valores simulados pueden ser 0.5, 0.7, entre otros. En reportes impresos, estos valores se expresan con cuatro

valores decimales, pero internamente los tiempos simulados consisten de 16 dígitos decimales. El modelador escoge la unidad base para el tiempo del modelo. El valor registrado de un reloj simulado sólo se puede incrementar conforme la simulación se lleva a cabo.

ESTRUCTURA DE LOS BLOQUES

Cada bloque en un modelo ocupa un espacio específico, tiene una palabra que identifican su operación y contiene desde cero hasta varios operandos. Las características de éstos se muestran en seguida detalladamente:

Espacio. Cada bloque ocupa un espacio específico en un modelo, estos espacios son numerados. El modelador no numera la localización de los bloques, pero tiene la opción de etiquetar uno o más bloques en el modelo, estas etiquetas identifican el sitio de los bloques. El poner este tipo de inscripción a los bloques facilita el hacer referencia a ellos en otras partes del modelo.

Operación. La operación de un bloque es una palabra clave que indica el papel que éste juega en el proceso del modelo. Cada bloque se caracteriza por una palabra única de operación. Algunas de ellas pueden ser las siguientes: GENERATE, TERMINATE, QUEUE, DEPART, ADVANCE, SEIZE RELEASE, ENTER, LEAVE y TRANSFER. Estas palabras en ocasiones se presentan abreviadas con sus cuatro primeras letras, por ejemplo GENERATE es posible representarla con GENE. Las palabras son semenates al lenguaje natural (en este caso el Inglés).

Operando. Los bloques generalmente contienen uno o más operandos, estos proveen información básica para llevar a cabo una acción. Para ciertos bloques los valores de todos los operandos deben ser dados explícitamente, para otros no se necesitaran valores ya que existen valores determinados para ellos (valores por omisión o default).

II.3.1 CREACIÓN DE TRANSACCIONES: BLOQUE GENERATE

Las acciones en un modelo de GPSS/H están basadas en movimientos de las transacciones a través de los bloques. El bloque GENERATE (GENE), es utilizado para crear e introducir una transacción al prototipo que se esta implementando en la computadora. Una transacción es creada en un momento simulado, pero en general una transacción no sale del bloque GENERATE en ese instante, sino hasta otro momento dado. A los instantes entre movimientos de transacciones consecutivas de un bloque GENERATE se les llama tiempos entre llegadas, este tiempo es una variable aleatoria y se ilustra en la *Figura III.3.1.1*:

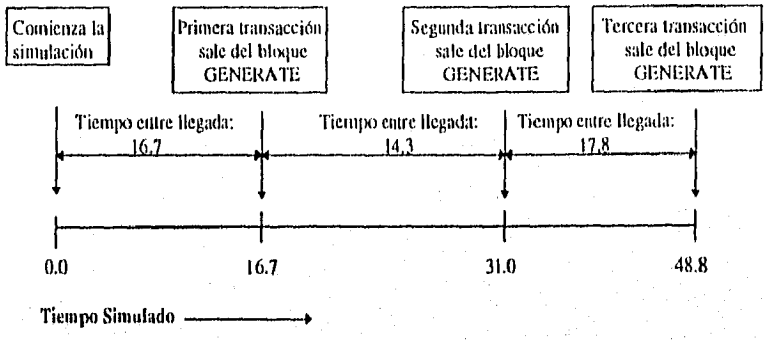


Figura III.3.1.1 Tiempos entre llegadas de un ejemplo

Por ejemplo, en una oficina se supone que en cierto día llega un trabajador en el momento 16.7 y luego llega el siguiente empleado en el momento 31.0 y así consecutivamente, al siguiente día se presenta un trabajador en el momento 18.6 y luego otro arriba en el momento 34.5, como se puede ver, los tiempos entre llegadas son completamente distintos de un día a otro. En general, el tiempo entre arribos que se aplica a cada transacción es determinado por una muestra obtenida de la distribución de probabilidad del tiempo entre llegadas tomadas del sistema real, en este caso se propone analizar al sistema de forma que se pueda determinar cada cuando llega un trabajador. Generalmente los tiempos entre llegadas son probabilísticos, y muchas veces estos comportamientos se fundamentan de una distribución de probabilidad, para determinar cual es la adecuada es necesario hacer un analisis estadístico de pruebas de bondad de ajuste correspondientes a los datos obtenidos del sistema real, se sugiere utilizar algún paquete como el *Statgraphics* en alguna de sus múltiples versiones.

Una distribución uniforme en el bloque GENE puede ser descrita indicando su media y el semigrango. Si se supone que una variable continua es distribuida de forma uniforme con una media de 15.0 y con un semigango de 4.5, entonces los valores podrán estar en el intervalo abierto de 15.0±4.5, es decir, que los valores pueden ser desde 10.5 (15-4.5=10.5) hasta 19.5 (15+4.5=19.5) Existen una infinidad de valores que pueden ser obtenidos de este intervalo. Cuando los tiempos entre llegadas se distribuyen uniformemente, la media entre llegadas en el bloque GENE utiliza el operando A y el semirango utiliza el operando B.

GENERATE A,B

Operando	Significado	Valores por default
A	Media entre tiempos de llegada	0.0
B	Semirango	0.0

Tabla III.3.1.1 Resumen del Bloque GENERATE, con los operandos A y B

Esto se puede ilustrar así: GENERATE 15.0, 4.5, con un intervalo abierto de 10.5 a 19.5.

Existen tres condiciones que deben cumplir los operandos A y B para que el bloque GENERATE sea válido, estas son:

- Si el operando A es mayor que cero y el operando B es igual a A entonces el tiempo entre llegada es cero, evento que no puede ocurrir. Es una consecuencia de que A ± B describe un intervalo abierto, entonces un ejemplo claro de lo que no puede suceder es GENERATE 25.0,25.0.

- El operando B no puede ser mayor que el operando A, si ésto no ocurriera así, daría un intervalo negativo, un ejemplo es GENE 30.0, 45.0

- Los operandos A y B no pueden tener valores negativos, un ejemplo es GENE -25,2.

Como se analizó anteriormente, es bastante fácil especificar tiempos entre llegadas uniformemente, después se verán otras distribuciones utilizadas en el GPSS/H. Los operandos pueden ser expresados en forma de números enteros. Cuando una transacción avanza del bloque GENE esto causa que el bloque genere otra transacción y la ponga a disposición para un movimiento futuro del bloque.

Como ya se mencionó las transacciones tienen una identificación, ésta se lleva a cabo en forma numérica, empezando con el número uno, luego el dos y así consecutivamente; esta anotación es completamente independiente del bloque que las crea.

LOS OPERANDOS DE LA PRIMERA SALIDA DE LA PRIMERA TRANSACCIÓN Y EL CONTEO LÍMITE DEL BLOQUE GENERATE

Este bloque además de tener a los operandos A y B puede tener a los operandos C y D. El operando C es un intervalo de compensación, utilizado para indicar el tiempo simulado en que la primera transacción debe salir del bloque GENERATE, las transacciones que se generen del bloque GENERATE consecuentemente serán determinadas de la forma usual utilizando los operandos A y B. La letra D indica un operando de conteo límite. Si es utilizado restringe el total de transacciones que deben entrar al modelo del bloque GENERATE, si se llega a la restricción este bloque deja de generar transacciones.

GENERATE A,B,C,D

Operando	Significado	Valores por default
A	Media entre tiempos de llegada	0.0
B	Semirrango	0.0
C	Tiempo determinístico para la llegada de la primera transacción	opcional
D	Conteo límite (número máximo de llegadas)	opcional

Tabla III.3.1.2 Resumen del Bloque GENERATE, con los operandos A, B,C y D

OPERANDO DE PRIORIDAD DEL BLOQUE GENERATE

Cuando dos transacciones se mueven al mismo tiempo, lo que se toma en cuenta es el orden de aparición en el programa, este orden puede ser controlado al utilizar el nivel de prioridad, este nivel es un atributo o característica de cada transacción. La prioridad toma la forma de una señal numérica, que puede situarse en valores comprendidos entre -2,000,000,000 y 2,000,000,000.

La preferencia inicial está determinada en el momento de creación de una transacción. El operando E indica la prioridad dada a las transacciones creadas cuando el bloque GENERATE es ejecutado. Su valor por default es cero, la prioridad de cierta transacción se muestra en la ventana de estado actual (esta ventana se analizará posteriormente), cuando una simulación se lleva a cabo en forma de interactiva (*test*). El papel que juega este operando es importante al momento de decidir el orden de movimiento, es decir, cuál de dos transacciones debe moverse primero cuando su tiempo simulado es el mismo. Mientras más alta (grande en número) sea la prioridad de una transacción con respecto a otra más pronto se moverá. Por ejemplo si una transacción tiene prioridad cinco y otra transacción nueve, entonces la primera en avanzar sería la transacción con prioridad nueve.

GENERATE A,B,C,D,E

Operando	Significado	Valores por default
A	Media entre tiempos de llegada	0.0
B	Semirrango	0.0
C	Tiempo determinístico para la llegada de la primera transacción	opcional
D	Conteo límite (número máximo de llegadas)	opcional
E	Nivel de prioridad en una transacción	0 (cero)

Tabla III.3.1.3 Resumen del Bloque GENERATE, con los operandos A, B, C, D y E

III.3.2 DESTRUCCIÓN DE TRANSACCIONES: BLOQUE TERMINATE

Existe un bloque llamado TERMINATE utilizado para destruir transacciones, en otras palabras, eliminarlas del modelo; este tipo de estructura nunca niega la entrada a una transacción y es factible presentarla en variadas ocasiones dentro de un mismo modelo.

El operando del bloque es una disminución del *contador de terminación* del modelo, es decir, indica la cantidad en que el valor del contador terminal del modelo debe ser reducido cada vez que una transacción es eliminada al llegar a este bloque. El contador es una variable entera que es inicializada con el valor de uno o más dependiendo del comando de control START. Conforme las transacciones llegan al bloque TERMINATE, el contador va

decrementa y en el momento que sea cero o menos la simulación se da por terminada. Aunque este contador puede ser negativo, en general el programa del modelo debe ser diseñado para que el contador llegue a un valor cero. Por default, este contador tiene el valor de cero, si una transacción llega a este bloque con un operando cero el valor del contador no será cambiado.

Este bloque juega dos papeles importantes, por un lado destruye las transacciones y por otro es una herramienta para controlar la duración de una simulación. En un modelo pueden existir varios bloques de este tipo pero sólo existirá un contador.

TERMINATE A

Operando	Significado	Valor por default
A	Disminución del contador de terminación del modelo	0

Tabla III.3.2.1 Resumen del Bloque TERMINATE, con el operando A

III.3.3 COMANDOS DE BLOQUE, CONTROL Y COMENTARIOS

Para llevar a cabo una simulación, los comandos de los bloques deben ser anexados al programa. De igual manera se debe proceder con otros tipos conocidos como *comandos de control*. La lista de comandos deberá ser ordenada en un archivo y luego corrida bajo GPSS/H. Aparte de las instrucciones de control y de bloques, en un archivo del modelo es importante incluir comentarios, dado que permite un mayor entendimiento del modelo. Este tipo de comandos puede ser resumidos de la siguiente manera:

1. ***Comandos de Bloques.*** Estos son los únicos que corresponden directamente a los bloques y son ejecutados sólo cuando una transacción llega a ellos.

2. ***Comandos de Control.*** Existen varios comandos de este tipo. Algunos son utilizados para controlar la ejecución de un archivo de un modelo. Otros son utilizados para definir ciertas características de los componentes de un modelo, y otros tantos son empleados para leer entradas o escribir salidas de los modelos de GPSS/H. No existen bloques que correspondan a este tipo de comandos, además que las transacciones no pueden entrar a ellos, éstas sólo se mueven de bloque a bloque y no es válido el movimiento de comandos de control.

3. ***Comentarios.*** Proveen a un analista la facilidad de incluir en un modelo la documentación necesaria como parte de un archivo del modelo. Esto permite que las personas que trabajen con el prototipo y su programación, tengan una mayor comprensión del modelo programado. Este tipo de comandos son opcionales pero siempre es recomendable su uso.

III.3.3.1 Estructura de Comandos

Estos comandos son bastante similares en su estructura a los de bloque. Tienen tres características básicas: una etiqueta, una operación y operandos. En general se pueden mencionar estas características:

1. **Etiquetas.** En algunos comandos son necesarias, en general funcionan como un identificador. Para algunos comandos de este tipo, el uso de etiquetas es opcional y para otros el uso no es permitido.

2. **Operación.** Es una palabra clave que sugiere el efecto del resultado de la ejecución de ese comando. Ejemplos de estas palabras pueden ser SIMULATE (simular), START (comienzo) y END (fin).

3. **Operandos.** Los comandos de control pueden tener cero o más operandos. Estos operandos proveen información en lo que la acción de un comando de control se basa. El número de operandos depende del tipo de comando de control, en general se representan con las letras A, B, C, entre otras. En ciertos casos los valores de estos deberán ser explícitos, pero en otros se podrá utilizar valores por default.

Formato de los Comandos de GPSS/H

Cada tipo de comando está compuesto por una o varias piezas de información. Cada parte de esta información es expresada como una secuencia de uno o más caracteres consecutivos. Por ejemplo: el comando correspondiente al bloque TERMINATE está compuesto por tres partes importantes: un bloque opcional de etiqueta, una palabra de operación (TERMINATE) y un valor opcional para el operando A. El formato de un comando se refiere a la forma como éste debe ser escrito en una línea de un programa, ya sea con una etiqueta, una palabra de operación y los valores de los operandos.

Los comandos de bloque y control llevan el mismo formato. Un comando contiene tres campos: etiquetas, operación y operandos. Un campo es una secuencia de columnas consecutivas de un comando. En la *Tabla III.3.3.1.1* se muestra un formato de como escribir un comando o bloque en GPSS/H. La etiqueta del comando, si es que la hay, se coloca en la columna dos. La palabra de operación es puesta en la columna 11; y sus operandos, si es que los hay, en la columna 22, sin extenderse a la columna 72. En el caso de que exista más de un operando, estarán separados por comas sin espacios. El primer espacio que aparece después de los operandos de cierto comando marcan el final del mismo, se pueden escribir ciertos comentarios después de ese espacio, estos comentarios no deben exceder la columna 80.

Columnas de un comando que forman los campos	Tipo de información tecleado en ese campo
2-9	Etiqueta o identificador de bloque*
11-20	Palabra de operación
22 y siguientes con un máximo de 72	Operandos*
> (la última columna del operando,+ un espacio)	Comentarios*

*Esta información es opcional

Tabla III.3.3.1.1 Formato de como los comandos son escritos en un programa de GPSS/H

Si un asterisco (*) está colocado en la columna uno de un comando, el lenguaje lo tomará como comentario, otra forma para indicar un comentario es escribirlo después del primer espacio del último operando. Las columnas 10 y 21 siempre estarán en blanco, dado que una etiqueta no puede exceder de 8 caracteres y ninguna palabra de operación que empiece en la columna 11 puede extenderse hasta la columna 21. Cualquier caracter alfabético que tiene parte de algún campo de una etiqueta, operación y operando deberá ser escrito en MAYÚSCULA. Para ilustrar esto se presenta un ejemplo:

I	Etiqueta (2-9)	Operación (11-20)	Operando (22-72)
		GENERATE	15,0,4,5 Se generan las transacciones
		TERMINATE	1 Las transacciones son destruidas
*			una por una, reduciendo el valor
*			del contador terminal en 1.

Tabla III.3.3.1.2 Ejemplo de los comandos en un programa de GPSS/H

III.3.4 Comandos de Control: SIMULATE, START y END

Los programas siempre deben contener un comando SIMULATE, uno o más START y exactamente un comando END. Se analizará cada comando en forma breve.

Comando SIMULATE

El comando de control SIMULATE nunca será etiquetado, tiene un operando opcional A. Esta instrucción le dice al programa que ponga el archivo del modelo en ejecución después de haber sido compilado. Si falta este comando, el archivo sólo será compilado. El operando A es un operando de tiempo límite. Pone un tiempo límite de computadora para que el paquete GPSS/H compile y ejecute un archivo. Esto permite que a modelos no estructurados de forma adecuada o con errores, se les ponga un límite en tiempo para que no ocurran loops infinitos. Este operando puede ser una constante entera o decimal, tiene por default unidades en minutos, sin embargo, cuando se desean utilizar segundos, el último carácter del operando deberá ser S para segundos. Cuando se llega al límite de tiempo, la simulación se detendrá y mostrará un mensaje de error. Cuando el modelo es corrido bajo DOS GPSS/H el operando de este comando será ignorado. Se ven ejemplos de esto en la *Tabla III.3.4.1.*

1	Etiqueta (2-9)	Operación (11-20)	Operando (22-72)
		SIMULATE	3S Ejemplo 1: el tiempo de CPU será 3 segundos
		SIMULATE	0.5 Ejemplo 2: el tiempo de CPU será de 0.5 minutos
		SIMULATE	Ejemplo 3: no hay tiempo límite

Tabla III.3.4.1 Varios ejemplos de uso del comando SIMULATE

Para determinar el mejor valor para este operando debe notarse que esto depende del tamaño del archivo, del número de bloques ejecutados y del tipo de computadora a ser utilizada. El DOS GPSS/H ejecuta mil o más bloques por segundo de tiempo de computadora.

1	Etiqueta (2-9)	Operación (11-20)	Operando (22-72)
	nunca será utilizada	SIMULATE	A

Operando o Etiqueta	Significado	Valores por default o resultado
Etiqueta	Nunca será utilizada	Deberá ser por default de otra manera habrá un error de tiempo de compilación
A	Operando de tiempo límite (número entero o decimal; unidades en minutos de tiempo de CPU, con utilización del carácter S al final del operando lo cambia a segundos)	No hay un número por default, no existirá límite de tiempo.

Tabla III.3.4.2 Resumen del comando de control SIMULATE

COMANDO START

La palabra de operación de este comando es START tiene un operando A, éste especifica el valor inicial del contador de terminación del modelo. Este operando debe tener valores enteros de uno o más; si no se especifica un valor se tendrá como resultado un error. Cuando este comando es ejecutado, ocurren tres de las siguientes situaciones:

1. Se le da inicio al contador de terminación del modelo.
2. Los bloques GENERATE son inicializados.
3. El movimiento de bloque a bloque de las transacciones da inicio.

Ejemplos del uso del comando START son:

1	Etiqueta (2-9)	Operación (11-20)	Operando (22-72)
		START	25 Ejemplo 1: valor inicial del contador en 25
		START	1 Ejemplo 2: valor inicial del contador en 1
		START	Ejemplo 3: causará un error compilación

Tabla III.3.4.3 Ejemplos del uso del comando START

1	Etiqueta (2-9)	Operación (11-20)	Operando (22-72)
		START	A

Operando o Etiqueta	Significado	Valores por default o resultado
Etiqueta	Identificador de comando START	Comando sin etiqueta
A	Valor inicial del comador de terminación del modelo	Sin valor, dará como resultado un error de tiempo de compilación.

Tabla III.3.4.4 Resumen del comando START

COMANDO END

El comando END tiene por objetivos en un archivo lo siguiente:

- Indicar el final físico de un archivo
- Su ejecución es finalizada y da el control al DOS o al sistema operativo en uso.

Este comando no tiene operandos, ni tampoco los requiere, por lo cual no están en forma opcional. Cada archivo deberá contener un solo comando END y deberá ser el último comando en el archivo.

1	Etiqueta (2-9)	Operación (11-20)	Operando (22-72)
		END	

Operando o Etiqueta	Significado	Valores por default o resultado
Etiqueta	Identificador de comando END	Comando sin etiqueta

Tabla III.3.4.5 Resumen del comando END

III.3.5 ORDEN BÁSICO DE UN PROGRAMA EN GPSS/H

En la *Figura III.3.5.1* se muestra como puede ser un programa en GPSS/H. Primero se debe escribir el comando de control SIMULATE, luego los comandos de bloques para el modelo, luego el comando START y finalmente el comando de control END.

Figura III.3.5.1 Estructura de un programa en GPSS/H

ESCRITURA DEL PROGRAMA

Antes de que un programa pueda ser ejecutado, deberá ser creado en un archivo de tipo texto. Cualquier editor de palabras o procesador que permita guardar archivos en formato *txt*, puede ser utilizado para crear estos programas. El editor de DOS servirá para crear archivos para luego ser ejecutados en el lenguaje GPSS/H, todos los programas hechos en este trabajo están elaborados con este editor. Se verá un ejemplo (*Programa III.3.5.1*) de un programa en el que se generan 25 transacciones de una distribución uniforme y se destruyen, es importante notar que los comandos deberán ser escritos en MAYÚSCULAS.

1	Etiqueta (2-9)	Operación (11-20)	Operando (22-72)
		SIMULATE	3S Pone como tiempo límite 3 segundos
*			
*	********	**********	**********
*			Creación y destrucción del las transacciones
		GENERATE	15,0,4,5 Llega las transacciones
		TERMINATE	1 Las transacciones son destruidas
*			una por una, reduciendo el valor
*			del contador terminal por 1.
*	********	**********	**********
*			Comandos de control
		START	25 pone como valor inicial al contador en 25
*			inicializa el bloque GENERATE
		END	terminación de la ejecución del programa

Programa III.3.5.1 Un ejemplo de un programa en GPSS/H

FORMAS DE EJECUCIÓN DEL PROGRAMA

Los programas se pueden ejecutar en dos formas: modo de proceso por lotes (*batch*) y modo interactivo (*test*). En la forma *batch* el usuario hace el programa, lo ejecuta y obtiene los resultados después de haber ejecutado el programa en forma de lotes. Cuando se utiliza el modo *test*, el analista puede interactuar con la simulación de varias formas, una de ellas es ver como las transacciones van avanzado en los bloques; de esta forma, el paquete proporciona una herramienta para corregir el modelo de interés.

Para correr un programa en GPSS/H en el sistema operativo DOS en modo interactivo se emplean los siguientes comandos:

```
gpssh nombrearchivo.gps tv type nowarn
gpssh nombrearchivo.gps tvtnw
```

Estos comandos siempre serán un requisito para ejecutar un programa en el GPSS/H. El *nombrearchivo* es el nombre del archivo donde se encuentra el programa y el *gps* es la extensión. El uso de **tv** indica que el archivo será corrido en forma interactiva, el uso de **type** muestra que el reporte de la simulación deberá ser escrito en la pantalla y **nowarn** manifiesta que, si ocurren mensajes de advertencia de tiempo, no serán mostrados en la pantalla. El **tvtnw** es la misma expresión que la anterior, pero más compactada.

Al utilizar el modo de *test*, después de la compilación, aparecerá un mensaje en la pantalla **"Ready!"** luego el usuario podrá introducir comandos de modo interactivo. En este modo el analista no podrá hacer algún cambio al programa al ser ejecutado. Estos comandos pueden ser introducidos en minúsculas o mayúsculas.

RASTREANDO TRANSACCIONES EN SU CREACIÓN Y DESTRUCCIÓN EN EL MODO INTERACTIVO

El primer comando al ejecutar un programa en modo de *test* es **step** (ste st o s, presionando la tecla F10). Esta instrucción significa que se puede continuar con la simulación, procediendo al punto de la simulación donde más de un bloque haya sido ejecutado y luego la interrumpe. Ahora se planteará como se usa este comando con el ejemplo *Programa III.3.5.1*. Como primer paso se debe dar una instrucción al sistema operativo para ejecutar este programa. Al escribir:

```
gpssh ejemplo1.gps tvtnw
```

dará como resultado la *Pantalla III.3.5.1*.

Como se puede observar aparece la palabra **Ready!:**. Al escribir el comando **step** aparecerá el siguiente mensaje:

```
XACT1: POISED AT BLOCK 2. RELATIVE CLOCK: 12.9327
```

Para dejar esto más claro se analizará la *Pantalla III.3.5.1*, la cual se encuentra dividida en tres partes:

- La pantalla de origen
- La pantalla de estado
- La pantalla de diálogo

```
BLOCK CURRENT   TOTAL      ejemplo1.gps SOURCE CODE
----- -------   -----  ------------------------------------

1         1          GENERATE 15.0,4.5 Llega las transacción
2         0          TERMINATE 1
                        Pantalla de origen
=============================================================
S/C: OFF  ABS CLOCK:  12.9327    REL CLOCK: 12.9327    TTG: 25
=============================================================
XACT: 1     CURBLK: 1     NEXTBLK: 2     CHAINS: CEC     PC:
                        Pantalla de estado
MARK-TIME: 12.9327    MOVE-TIME: 12.9327    PRIORITY: 0
=============================================================
AN INTRODUCTION TO SIMULATION USING GPSS/H
          by Thomas J. Schriber (Wiley, 1990)

     Copyright (c) 1990, WOLVERINE SOFTWARE CORPORATION
          ANNANDALE, VIRGINIA 22003-2500, USA
Ready!
: step
XACT 1 POISED AT BLOCK 2.  RELATIVE CLOCK: 12.9327
:                       Pantalla de diálogo
```

Pantalla III.3.5.1 Muestra como se verá al ejecutar un programa en modo interactivo

Pantalla de origen (Source). La pantalla superior es la de origen, en ella se muestra, en la parte derecha el nombre del archivo del programa y la etiqueta **SOURCE CODE**. Debajo

de esta se encuentran cinco comandos de bloques consecutivos. En la parte izquierda, se encuentran unas columnas etiquetadas con **BLOCK, CURRENT** y **TOTAL**. La columna **BLOCK** muestra la localización en el modelo que ocupa los comandos de bloques que se muestran en la parte derecha. Las columnas **CURRENT** y **TOTAL** muestran los conteos de las transacciones para el bloque correspondiente. El conteo **CURRENT** indica cuántas transacciones (*Xacts*) están en ese bloque, y el conteo de **TOTAL** muestra cuantas veces cierto bloque ha sido ejecutado.

Pantalla de estado (Status). Se encuentra localizada a la mitad de la pantalla completa, muestra información que describe el estado actual del modelo. Las etiquetas **ABS CLOCK** (contabiliza el tiempo transcurrido desde que dio inicio la simulación) y **REL CLOCK** (contabiliza el tiempo transcurrido a partir de cierto punto de interés) muestran los valores actuales de los relojes absoluto y relativo, respectivamente. Generalmente estos valores siempre serán iguales a menos que se simulen varias réplicas del mismo modelo contínuas con el comando RESET. La etiqueta **TTG** permite visualizar el valor actual del contador de terminación. En las siguientes dos filas se muestra información sobre los movimientos de las transacciones. En la fila dos, la **XACT,** etiqueta a las transacciones con su número de identificación, las etiquetas **CURBLK** y **NEXTBLK** corresponden al bloque actual de una transacción y al siguiente bloque al que intentará avanzar; la información se despliega en términos de número de localización de un bloque. **CHAINS** etiqueta la cadena donde reside esta transacción. La cadena de eventos actuales está compuesta de las programaciones de transacciones para avanzar de alguna forma durante el modelo en el tiempo simulado. En la fila tres, **MARK-TIME,** es la etiqueta el tiempo de reloj absoluto de donde la más reciente transacción salió del bloque GENERATE. **MOVE-TIME** etiqueta el tiempo simulado en que esta transacción se está moviendo. Y la etiqueta **PRIORITY** da la información del nivel de prioridad de esta transacción.

Pantalla de diálogo (dialog). La parte inferior muestra la pantalla de diálogo, en ella se muestran los comandos de modo interactivo tecleados y los mensajes como resultado. En ella puede aparecer información dada por el GPSS/H como:

XACT 1 POISED AT BLOCK 2, RELATIVE CLOCK 12.9327

El mensaje *poised at* significará que una transacción en movimiento está a punto de tratar de situarse en un siguiente bloque. El enunciado anterior indica que la transacción uno es la que está en movimiento y la que intentará avanzar al bloque que se encuentra localizado en la posición dos. El movimiento de la misma ocurrió en un tiempo simulado de 12.9327.

XACT 1 DESTROYED AT BLOCK 2, RELATIVE CLOCK: 12.9327

Es fácil comprender que este mensaje indica que la transacción ha sido destruida y que fue eliminada en el momento 12.9327.

FORMA DE MANDAR A UN ARCHIVO LA INFORMACIÓN OBTENIDA DE UNA SIMULACIÓN

Cuando se ejecuta un programa en modo interactivo y se desea guardar la pantalla de diálogo, se debe teclear lo siguiente:

```
gpssh ejemplo1.gps tv nowarn
```

Es importante hacer notar la importancia de no utilizar la opción type. Después de teclear el comando anterior aparecerá la frase Ready!: y se le debe dar la siguiente instrucción;

```
sel tlog on
```

Al hacer lo anterior la pantalla de diálogo se guardará en un archivo en el directorio donde se encuentra el programa llamado *nombrearch.lis*, siendo este caso: *ejemplo1.lis*.

III.3.6 SUSPENSIÓN DEL MOVIMIENTO DE TRANSACCIONES EN EL TIEMPO: BLOQUE ADVANCE

Cuando existe tráfico en un sistema, usualmente llega a puntos donde se hace una pausa y toma cierto tiempo antes de continuar con su movimiento. Usualmente estos puntos corresponden a lugares donde el sistema recibe algún servicio. Un ejemplo de ello es cuando un doctor que examina a un enfermo en los pasillos de un hospital antes de llegar a una sala de emergencia.

En el ejemplo anterior, el tiempo transcurre mientras una unidad de tráfico permanece en cierto lugar, en ciertos puntos, en el sistema; mientras recibe cierto servicio, se toma un tiempo, así como cuando el tráfico se mueve de un punto a otro. Existen algunas ocasiones en el modelaje de GPSS/H en las que es apropiado que cierta transacción avance a un bloque y que permanezca allí por un momento para simular, por ejemplo, el tiempo requerido para recibir un servicio en cierto instante. El bloque ADVANCE permite hacer esto, cuando alguna transacción avanza a él, esta transacción se atrasará por cierto tiempo; después avanzará al siguiente bloque. Por lo general el tiempo que la transacción permanece en ella es aleatoria. Al igual que el bloque GENERATE, este bloque contiene los operandos A y B, los cuales son utilizados para especificar la distribución de probabilidad de tiempo que permanecerá en ese bloque (es importante recordar que estos valores son uniformes, para especificar otro tipo de distribución se recomienda ver un análisis más amplio que se desarrolla en el Capítulo VI). Estos operandos juegan el mismo papel que el del bloque GENERATE.

```
ADVANCE A,B
```

Operando	Significado	Valores por default
A	Tiempo medio de retención	0.0
B	Semirrango de retención	0.0

Tabla III.3.6.1 Resumen del bloque ADVANCE

UTILIZACIÓN DEL CONTADOR EN EL COMANDO STEP

En forma general el comando step toma la siguiente forma:

```
step número de pasos
```

El *número de pasos* debe ser un entero positivo que indica cuántas ejecuciones de bloques se deben llevar a cabo. Este valor por default es 1.

IDENTIFICACIÓN DE BLOQUES

Puede ser de interés el referirse a cierto bloque desde otro bloque, esto se hace a través de dos procesos:

- Etiquetación del bloque
- Hacer referencia al bloque por su etiqueta

Una etiqueta es un identificador para la localización que cierto bloque ocupa. Para etiquetar cierto bloque se utiliza el campo de etiquetación, y debe constar de uno a ocho caracteres alfanuméricos, siendo el primero una letra del abecedario. Se debe evitar el uso de palabras reservadas para nombrar las etiquetas. Un ejemplo de cómo se debe hacer la identificación se puede ver en el ejemplo de la *Tabla III.3.6.2.*

1	Etiqueta (2-9)	Operación (11-20)	Operando (22-72)
	REPETIR	ADVANCE	250

Tabla III.3.6.2 Un ejemplo de la utilización de una etiqueta

III.3.7 MOVIMIENTO NO SECUENCIAL DE TRANSACCIONES EN UN MODELO: BLOQUE TRANSFER

Algunas veces una transacción necesita llegar al siguiente bloque de manera no secuencial, para hacer esto, se utiliza el bloque sin condición TRANSFER. Contiene dos operandos, el operando A no debe ser usado al utilizar el bloque TRANSFER incondicionalmente. El operando B *indica* en donde se encuentra el siguiente bloque al que se trata de llegar cuando una transacción llega al bloque incondicional TRANSFER. Este señalamiento se lleva a cabo etiquetando el siguiente bloque y a utilizar esa etiqueta como operando B.

TRANSFER A,B

Operando	Significado	Valores por default
A	Deberá ser omitido en forma de incondicional	El usuario deberá poner un valor por default
B	Apuntador a un bloque (copia de la etiqueta asociada a ese bloque)	Causará error de ejecución al no poner nada

Tabla III.3.6.1 Resumen del bloque TRANSFER

EL COMANDO DISPLAY EN MODO INTERACTIVO

El usuario puede pedir información utilizando el comando display, esta instrucción se lleva a cabo, de la forma general:

display *tipo de inf.*

donde *tipo de inf.* representa el tipo de información que el usuario desea observar. Estos tipos son los siguientes:

- *blo.* Despliega información de los conteos de bloques actuales y totales.
- *cec.* Despliega información de la cadena actual de eventos.
- *clocks.* Despliega información de los valores de los relojes absolutos y relativos.
- *fec.* Despliega información de la cadena futura de eventos.
- *xact="identificación de transacción".* Las características de cierta transacción identifica (xact=25).

La forma más breve de utilizar el comando display es usar la letra d. Es posible desplegar más de un solo tipo de información, esto se hace usando espacios entre tipos de información.

Función de tecla	Efecto
F1	Muestra la línea que se encuentra hasta arriba de diálogo que todavía se encuentra en la memoria de la computadora
F2	Sirve como comando set tv off
F3	Muestra la última línea reciente de diálogo
F4	Sirve como comando set tv on
F5	Sube 20 columnas de la izquierda de las ventanas de diálogo y de origen
F6	Sube 10 columnas de la izquierda de las ventanas de diálogo y de origen
F7	Sube una línea en la ventana de origen
F9	Baja una línea en la ventana de origen
F10	Sirve como comando step 1

Estos comandos pueden ser recordados al teclear:

display pf

Es importante que a lo largo de la tesis el lector programe y analice las diferentes formas de ejecutar un modelo, así como los bloques vistos hasta este momento.

CAPÍTULO IV
MODELOS CON UN SOLO SEVIDOR

IV.1 CONCEPTO DE SERVIDOR

El lenguaje GPSS/H es adecuado para modelar sistemas compuestos por unidades de tráfico, que compiten entre sí para hacer uso de un conjunto limitado de recursos. Existe una gran cantidad de sistemas reales que se ajustan a esta descripción. Por ejemplo: sistemas de manufactura o de producción, sistemas de cuidado de la salud, sistemas de transporte, redes de comunicación, sistemas de defensa, sistemas de seguridad social y líneas de espera en general.

Dado que el GPSS/H es un lenguaje especialmente diseñado para los sistemas de líneas de espera, es necesario definir alguna terminología básica. Vale la pena recalcar el hecho de que gran cantidad de problemas de simulación pueden plantearse como líneas de espera.

El diseño de líneas de espera debe tomar en cuenta dos costos diferentes. El primer costo involucra el de proveer un servicio. El segundo implica el de los elementos que permanecen esperando en la fila. Generalmente el valor de espera está en función del costo de la deserción de la fila, debe existir un punto de equilibrio entre estos dos costos, como se puede ver en la *Figura IV.1.1*. El diseño óptimo pretenderá minimizar la suma de los gastos de servicio y de espera. En algunos casos resulta difícil o imposible estimarlos, por ejemplo, piénsese en una ambulancia donde no es posible asignar un valor monetario a la salud o vida de una persona.

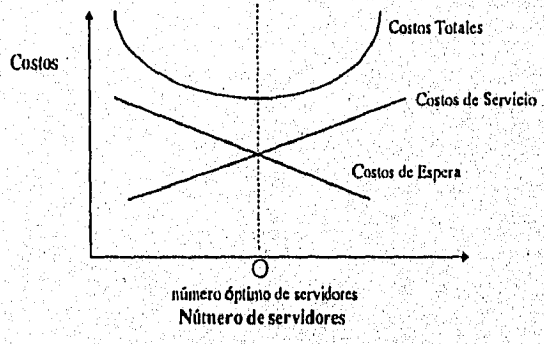


Figura IV.1.1 Costos del sistema de líneas de espera

Las filas pueden describirse según sea su número de canales (o servidores) y la cantidad que posea de fases. El número de canales describe el número de filas en el sistema, puede ser único o múltiple. El conjunto de fases se refiere al número de servicios (*facilities*) que debe recibir cada elemento antes de completar el sistema. Puede ser único o múltiple y también considerarse el caso de las filas múltiples con servidores en paralelo o cualquier mezcla de estos casos.

Por ejemplo, un canal de una fase, puede ser la fila para comprar boletos en una taquilla única, como se puede analizar en la *Figura IV.3.1*. Un canal de varias fases, pueden ser los clientes de una cafetería de auto servicio o un restaurante con servicio de buffet, un ejemplo puede ser la *Figura IV.6.3*. Canales múltiples de fase única, pueden ser las cajas de un banco: puede tenerse en este caso una fila con varios servidores, o varias filas con varios servidores, un ejemplo puede ser el de la *Figura V.3.2* o *Figura V.3.3*. Canales múltiples de fases múltiples puede ser la recepción, archivo y envío en una oficina grande de correo, en la *Figura IV.1.2* se muestra un ejemplo de lo anterior.

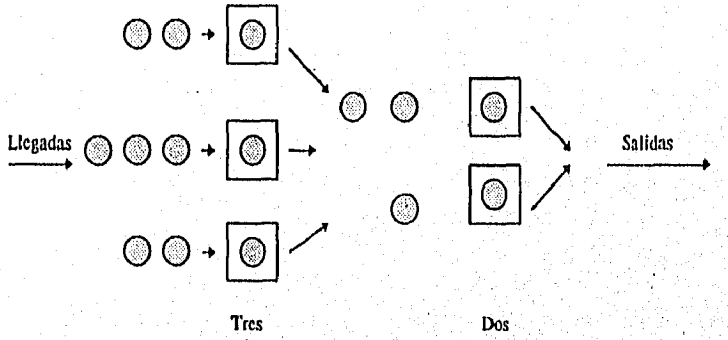


Figura IV.1.2 Sistema de dos fases con múltiples canales

Una secuencia de unidades de tráfico o transacciones *(transactions)*, por ejemplo clientes, se genera en forma aleatoria y debe ir pasando por una serie de entidades *(process entities)* o bloques *(blocks)*. Se llamará fila *(queue)* a los elementos que esperan en una línea para recibir un servicio *(facility)*. Pueden asociarse retardos aleatorios con cada una de estas entidades. El sistema incluye la fila, el servidor (o los servidores), y los elementos que están recibiendo el servicio.

Por otro lado, existe una distribución de probabilidad asociada a la forma en que se incorporan elementos a la fila, así como una distribución asociada a los tiempos de espera y de servicio. Las distribuciones más utilizadas son la uniforme y la exponencial, dado que estas se acoplan bastante bien a los modelos de este tipo.

Existe una disciplina en la estructura, esto se refiere al orden en que se atiende a los elementos en la fila, la más común y aparentemente la más justa es la regla PEPS(Primero en Entrar Primero en Salir) o FIFO (First In First Out). Pueden o no existir prioridades *(priorities)*; finalmente es factible el cambio de líneas de espera y la deserción.

IV.2 NATURALEZA DE LOS SERVIDORES

En su forma más simple, un servidor es una persona u objeto (pudiendo ser también combinación de los dos) que provee algún servicio en particular. Algunos ejemplos claros de un servicio que una persona brinda son:

- Un trabajador en una línea de montaje
- Un doctor
- El cajero de un banco
- El cajero de un supermercado
- La recepcionista de la oficina de Relaciones Exteriores

Algunos ejemplos de servidores que son objetos:

- Una máquina que corta papel
- Una impresora conectada a una computadora
- Un salón de operación en un hospital
- Un tanque de gas que infla globos

Los servidores son componentes esenciales de sistemas, se les puede denominar como recursos, dado que muchas veces estos servidores están limitados en su cantidad, usualmente son escasos en los sistemas.

El término *servidor* es utilizado para describir un recurso que únicamente puede responder a una solicitud de servicio a la vez. Por ejemplo, una sala de operación solamente puede ser ocupada para operar a un individuo, otro caso es el de una impresora que exclusivamente puede imprimir un archivo a la vez, en el ejemplo de la recepcionista de pasaportes visto en el Capítulo II, a quien se nombrará SOFIA que justamente puede atender a un trabajador a la vez.

El que un servidor pueda dar cierta atención depende de si se encuentra en servicio y también de si está trabajando sin tomar tiempos libres. Por ejemplo, la empleada de Relaciones Exteriores, Sofía entra a trabajar a las 9:00 a.m., y sale a las 3:00 p.m., podrá atender al trabajador únicamente en ese lapso de tiempo. O la impresora, si se usa en cierto periodo, y se llega a descomponer, entonces el servidor no estará trabajando en orden, hasta que sea compuesta o se cambie por otra.

Aun cuando lo anterior ocurra, el servidor en ocasiones no podrá responder inmediatamente a cierta solicitud de servicio. O quizá pueda ocurrir que, cuando el servidor se encuentra dando el servicio a alguien, algo suceda que haga que no finalice el servicio en ese momento. Un ejemplo de ello puede ser el caso de la oficinista de pasaportes, cuando se encuentra atendiendo a un trabajador, llega el jefe y la manda llamar para darle ciertas instrucciones.

Ahora, si se supone que un servidor termina de dar un servicio y dos o más solicitudes de trabajo se encuentran en espera, es entonces cuando éste debe decidir a quien atender después. Existen varias alternativas que pueden ser utilizadas para tomar esta decisión, y son llamadas, en forma general, *reglas de servicio* o *disciplina de una línea*. El orden en que los clientes son atendidos es llamado orden de servicio, algunas alternativas pueden ser las siguientes:

Primero en llegar, primero en ser atendido. Las solicitudes son atendidas en orden cronológico.

Primero en llegar, primero en ser atendido, con niveles de prioridad. Es una variación de la anterior, con niveles de prioridad asignados a ciertas solicitudes, un ejemplo de ello puede ser el caso de una persona gravemente herida de bala en una sala de operación. Las solicitudes con mayor prioridad son las primeras en ser atendidas y así las siguientes.

Tiempo de procesamiento más pequeño. En esta alternativa las solicitudes son jerarquizadas en base al tiempo esperado de servicio, es decir, en el tiempo promedio de atención, por lo general para obtener este valor se requiere de una variable aleatoria en la que el valor no se sabe hasta que el servicio ha sido finalizado. Mientras más pequeño sea el tiempo requerido de servicio, mayor prioridad tendrá dicha solicitud.

Ciertamente no todas las solicitudes de servicio esperan a que llegue su turno de ser atendidas. Por ejemplo, si hay muchas personas que esperan a ser atendidas, quizá la siguiente persona en llegar cambie de planes y no se quede a esperar. También es posible que después de cierto tiempo un individuo que se encuentra en la fila, decida retirarse de ella.

Algunos de los recursos en un sistema pueden tomar la forma de un solo servidor y otros pueden tomar la forma de grupos de dos o más servidores idénticos, es decir, que los servidores tienen las mismas características, el tiempo de servicio de uno es igual al otro. Por ejemplo, en un hospital, pueden existir dos doctores que tengan las mismas características, haciéndolos así idénticos en su forma de operar, dos o más servidores idénticos pueden ser modelados como servidores individuales o pueden ser modelados en forma grupal.

En el GPSS/H, un solo servidor puede ser modelado en forma individual, así como los servidores en grupo idénticos pueden ser modelados en forma grupal o individualmente, casi cualquier tipo de servicio puede ser modelado en GPSS/H.

IV.3 Consideraciones Lógicas en un Modelo de un Solo Servidor

Un sistema de un solo servidor puede ser visto como en la *Figura IV.3.1*.

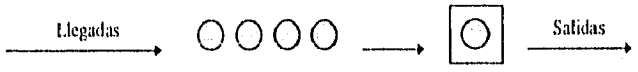


Figura IV.3.1 Un sistema de un solo servidor

Los círculos son las unidades de tráfico (clientes o trabajo en proceso) que requieren de ser atendidas, el cuadrado es un servidor y el círculo dentro del cuadro es la unidad de tráfico que es atendida en ese instante. A la línea que está formada por estas unidades de tráfico que esperan a ser atendidas, se le llama línea de espera. La combinación del servidor, la unidad de tráfico que es atendida y los que esperan el servicio se le llama sistema de línea de espera (**queuing system**). Quizá sea oportuno visualizar este sistema en forma de tiempo orientado como en la *Figura IV.3.2*. De esta manera se puede imaginar una unidad de tráfico e identificar los pasos que se llevan a cabo durante todo el sistema. Estos pasos fueron ordenados en forma cronológica. Para ilustrar esto se puede tomar el ejemplo de la recepcionista Sofía, en donde se debe pasar por cinco pasos seguidos, como se muestra en la *Figura IV.3.2*.

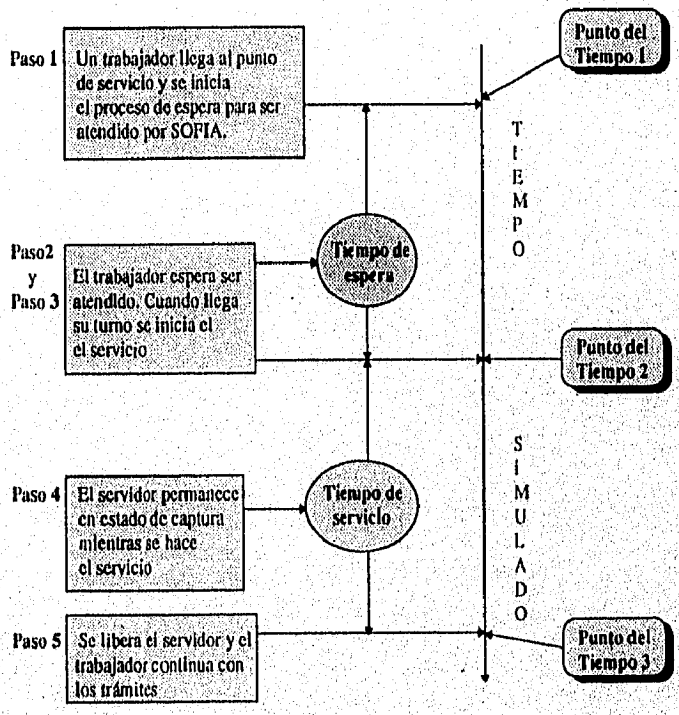


Figura IV.3.2 Visualización en forma orientada de un servicio de un solo servidor

En el primer paso, el trabajador llega a la línea, así es como se inicia el proceso de espera, ocurriendo en cierto punto del tiempo en este caso Punto del Tiempo 1. En el segundo paso, el trabajador espera ser atendido, lo cual ocurrirá en cierto momento. En el tercer paso se finaliza el tiempo de espera y se inicializa el proceso en el que se recibe el servicio. En el paso cuatro, se mantiene a SOFIA en servicio y transcurre cierto lapso, es decir, el tiempo de servicio. Finalmente, en el paso cinco, es cuando se termina de atender al trabajador, finalizando tiempo de servicio.

Lo anterior debe ser tomando en cuenta al momento de programar, sin importar el lenguaje en uso.

IV.4 Modelaje de Servidores Individualmente: Entidad de Facility

En GPSS/H una *entidad* es una herramienta o un conjunto de herramientas que pueden ser utilizadas para lograr ciertos objetivos en un modelo. Por ejemplo, el lenguaje provee transacciones, para lograr esto el lenguaje utiliza cierto tipo de entidades para modelar unidades de tráfico discretas para ciertos sistemas. Existen varios tipos de entidades, que son utilizadas para construir modelos de simulación, usualmente estos tipos de entidad implican el uso de uno o más tipos de bloques, con ciertas características para lograr algún objetivo.

Este lenguaje provee una entidad llamada facility, que sirve como herramienta para modelar servidores únicos. Esta entidad contiene dos tipos de bloques, el bloque SEIZE y el bloque RELEASE.

Cuando esta entidad es utilizada para modelar un sistema de un solo servidor, las transacciones son empleadas para modelar a los clientes que requieren de un servicio. Un ejemplo claro puede ser de una *facility* que modela el trabajo de SOFIA y las transacciones serán utilizadas para representar a los trabajadores que se interesan en sacar el pasaporte para ir a trabajar a Canadá.

Una transacción en el punto del tiempo uno en la *Figura IV.3.2*, tratará de avanzar de su bloque actual al bloque SEIZE. Si el servidor se encuentra dando servicio a otra transacción, entonces le negará su entrada al bloque SEIZE y permanecerá en el bloque actual. Es decir, la transacción es forzada a esperar su turno para que se le de el servicio. Cuando logra entrar al bloque SEIZE, se ejecutará este y se iniciará el tiempo de recepción; para finalizar el servicio, la transacción ejecuta el bloque RELEASE.

Al ejecutar el bloque SEIZE, es cuando a una transacción se le da un servicio hasta que eventualmente ejecuta el bloque RELEASE. Una transacción simplemente toma *control del servidor* al ejecutar el bloque SEIZE; no necesita permanecer en el bloque para continuar con el control del servidor.

El bloque SEIZE es un tipo bloque que puede negar la entrada a una transacción, forza a una transacción a permanecer en ciertas partes de un modelo, hasta que el estado del sistema le permita avanzar.

Los bloques SEIZE y RELEASE tienen un solo operando A, el cual sirve para identificar a un servidor en particular (facility), una transacción solicita un servicio al tratar de ejecutar al bloque SEIZE y luego deja el servicio al ejecutar el bloque RELEASE.

Un servidor (facility) en particular debe ser identificado en los bloque SEIZE y RELEASE, dado que pueden existir dos o más servidores en un modelo. Esto hace necesario indicar al bloque SEIZE que servidor se requiere de su servicio y al bloque RELEASE que servidor regresa a su estado de *no ocupación*. A cada facility o servidor se le debe describir con un nombre o identificador, este nombre debe estar estructurado de uno a ocho caracteres alfanuméricos, siendo el primero de orden alfabético, en general pueden consistir en un mínimo de tres caracteres alfabéticos, es importante evitar el uso de palabra reservadas en GPSS/H. Los identificadores pueden contener números enteros positivos.

Operando	Significado	Valores por default
A	Identificador del servidor que es capturado en el bloque SEIZE y liberado en bloque RELEASE	No existe valor

Tabla IV.4.1 Resumen de los bloques SEIZE y RELEASE

Ahora, si se utiliza el ejemplo de los trabajadores y SOFIA, en este caso el identificador será SOFIA, como se muestra en la *Figura IV.4.1*, los bloques SEIZE y RELEASE están situados en el modelo utilizando la lógica de la *Figura IV.3.2*, como se indica, una transacción solicita a un servidor al tratar de avanzar al bloque SEIZE donde el operando A identifica al servidor. Si el recurso se encuentra libre (idle) es entonces cuando se ejecuta el bloque sin espera, el servidor en ese instante comienza el servicio. Ahora, si el servidor se encuentra atendiendo a otra transacción, negará la entrada a la transacción. El bloque anterior al SEIZE puede ser cualquier tipo de bloque.

¿Qué ocurre con las transacciones a las que se les niega la entrada al bloque SEIZE y que permanecen en el bloque anterior? Cuando un servidor termina el servicio de una transacción existe un efecto de desbloqueo de las transacciones en espera. El GPSS/H guarda en su memoria un mensaje que le indica que después debe dar la oportunidad de avanzar a estas transacciones bloqueadas. La primera en tratar de avanzar para ser atendida por un servidor, permanecerá en la entidad hasta que termine su servicio, y así las siguientes transacciones intentarán avanzar y permanecerán en el bloque anterior al SEIZE, bloqueadas hasta que esa transacción termine con el servicio.

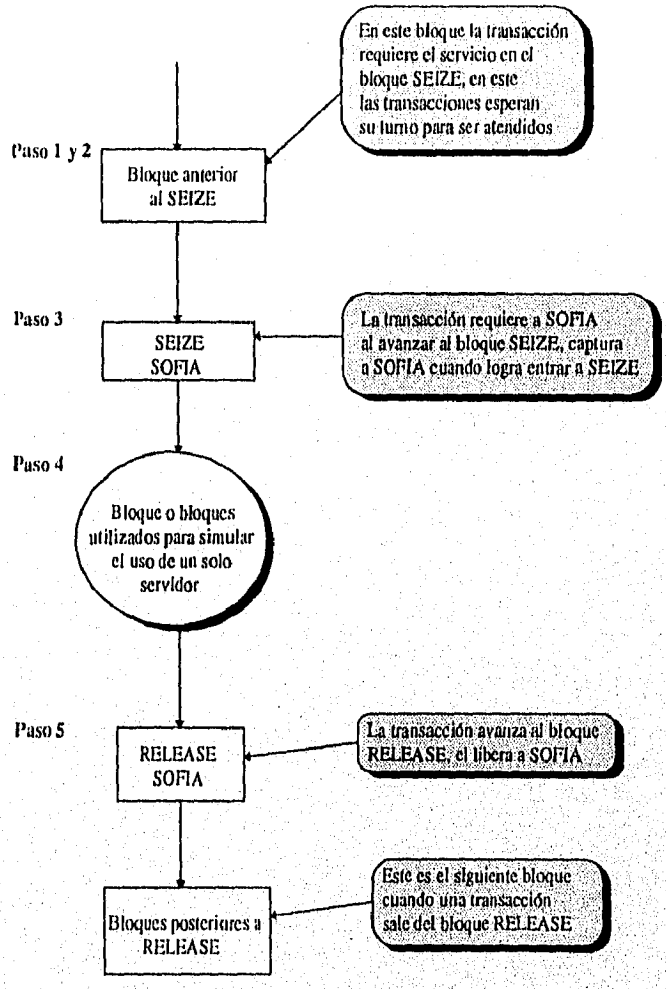


Figura IV.4.1 Localización de los bloques SEIZE y RELEASE

Lo que ocurre después de que se ha terminado con el servicio, es que continúa con los siguientes bloques. Después que una transacción ha terminado el servicio, ejecutará el bloque RELEASE para dejar libre al servidor. Cuando se ejecuta el bloque RELEASE el lenguaje guarda en su memoria el recordatorio de que debe permitir la entrada a otras transacciones desbloqueadas.

Los bloques SEIZE y RELEASE son bloques complementarios entre sí. Cuando el SEIZE es ejecutado el servidor es puesto en un estado de ocupación; en forma complementaria cuando un RELEASE es ejecutado el servidor es puesto en un estado libre.

REPORTES DE SERVIDORES

Durante una simulación, el GPSS/H colecciona información de cada servidor en un modelo y reproduce un reporte de cada servidor, la *Tabla IV.4.2* muestra un ejemplo de un reporte de un servidor llamado SOFIA.

	--AVG-UTIL-DURING--								
FACILITY	TOTAL TIME	AVAIL TIME	UNAVL TIME	ENTRIES	AVERAGE TIME/XACT	CURRENT STATUS	PERCENT AVAIL	SEIZING XACT	PREEMPTING XACT
SOFIA	0.853			20	4.496	AVAIL			

Tabla IV.4.2 Un ejemplo de un reporte de un servidor

- La columna **FACILITY** indica el nombre del servidor.
- **--AVG-UTIL-DURING-- TOTAL TIME**, muestra el tiempo total simulado, en forma de fracción, en la que el servidor estuvo ocupado, es decir dando servicio, en este caso el servidor SOFIA estuvo ocupado en un 80% del tiempo total.
- **--AVG-UTIL-DURING-- AVAIL TIME**, muestra en fracciones el tiempo simulado, la disponibilidad de un servidor que estuvo en estado de servicio. Este tiempo es la cantidad de tiempo simulado que un servidor estuvo disponible para trabajar. En este caso, tanto este valor como el siguiente no son impresos, dado que durante toda la simulación el servidor siempre estuvo disponible a *trabajar*. Por ejemplo, SOFIA siempre estuvo dispuesta a recibir a un trabajador y hacer el trámite correspondiente.
- **--AVG-UTIL-DURING-- UNAVL TIME**, muestra en fracciones, el tiempo simulado, la no disponibilidad de un servidor que estuvo en estado de servicio. Es la cantidad de tiempo simulado que un servidor no estuvo disponible a trabajar, este valor no aparece en este reporte dado que es cero.
- **ENTRIES**, indica el número de veces que un servidor fue puesto en estado de captura, es decir, indica el número de clientes que fueron atendidos por el servidor.
- **AVERAGE TIME/XACT**, muestra el tiempo promedio de retención por el servidor a una transacción (tiempo de atención), es decir, el tiempo promedio que SOFIA se tardo con cada trabajador en promedio.
- **CURRENT STATUS**, indica si en el momento de la elaboración del reporte, el servidor estaba disponible, es decir, si se encuentra disponible a trabajar o no disponible.
- **PERCENT AVAIL**, muestra la fracción del total de tiempo simulado que un servidor estuvo en disponibilidad de trabajo.
- **SEIZING XACT**, muestra el número de identificador de alguna transacción que se encontraba en servicio en el momento de elaborar el reporte. Un servidor puede estar disponible y a la misma vez puede estar en servicio, en el GPSS/H la palabra *(avail)* disponible tiene el significado de estar apto para trabajar y no tiene nada que ver con el concepto en servicio o libre.

- PREEMPTING XACT, muestra el identificador de alguna transacción que mantenía al servidor en estado pre-vaciado *(preemption)* al momento de que el reporte fue hecho. Un servidor es puesto en este estado si una transacción toma al servidor de otra y empieza a utilizarlo.

En modo interactivo estos reportes pueden ser analizados al teclear **display fac.**

Para hacer una corrida de un programa en modo batch se teclea lo siguiente:

```
gpssh nombr_arch.gps
```

Nombr_arch.gps representa el nombre del archivo donde se encuentra el programa, siendo **gps** la extensión del archivo. Al hacer esto el GPSS/H guardará un archivo *nombr_arch*.lis en donde estarán localizados los resultados de la simulación.

CUÁNDO UN BLOQUE QUE SIGUE A GENERATE PUEDE NEGAR LA ENTRADA

Se ha dicho que un bloque es ejecutado cuando una transacción avanza hacia él sin importar si inmediatamente la transacción se mueva del bloque actual al siguiente; aunque el bloque actual ya ha sido ejecutado muchas veces la transacción debe permanecer en ese bloque por cierto tiempo simulado. Sin embargo, el bloque GENERATE es una excepción a lo anterior. Este bloque no se ejecuta cuando una transacción llega a él, sino cuando sale de él al siguiente bloque. En la mayoría de los casos al bloque GENERATE le sigue un bloque que nunca le niega la entrada a las transacciones, cuando esto sucede las transacciones siempre podrán salir al momento que llegan a él, no existirá tiempo de retraso simulado.

Pero si existe un bloque después del GENERATE que puede negar la entrada, como el bloque SEIZE, entonces existirá un tiempo de retraso simulado entre el movimiento de una transacción al bloque GENERATE y la eventual ejecución del bloque GENERATE.

Por ejemplo, se puede considerar el *Programa IV.4.1*:

```
      SIMULATE
*
      GENERATE  10
      SEIZE     RECURSO
      ADVANCE   15
      RELEASE   RECURSO
      TERMINATE 1
*
      START     25
*
      END
```

Programa IV.4.1 Un ejemplo con un GENERATE con retraso

Los tiempos entre llegadas son determinísticos e iguales a 10. Si se considera esto parecería que las transacciones saldrían del bloque GENERATE en los momentos 10.0, 20.0, 30.0, etc. Esto sería el caso si del bloque GENERATE salen y entran al mismo tiempo. Pero dentro del contexto solamente la primera transacción puede entrar y salir de bloque GENERATE sin tiempo de retraso. La segunda transacción entraría al bloque GENERATE pero saldría hasta 25.0. Y la tercera entraría al bloque en el momento 35.0 y saldría hasta el momento 40.0

En el ejemplo del *Programa IV.4.1* se puede analizar que si el siguiente bloque del GENERATE puede negar la entrada, causa que el tiempo de realización entre movimientos de transacciones del bloque GENERATE pueda exceder los tiempos entre llegadas especificados en los operandos A y B del bloque GENERATE. En el ejemplo del *Programa IV.4.1*, después de la primera transacción se generan, cada 15 unidades de tiempo y no cada 10 como fue descrito en el operando A.

Aunque muchas veces se puede tomar ventaja de este comportamiento, es necesario evitarlo pues genera distorsión en los valores de los operandos, se inserta en estos casos un bloque ADVANCE con un valor de operando cero, este tiempo de espera se le llama bloque ADVANCE **dummy**. Como ya se sabe, este bloque nunca niega la entrada y las transacciones siempre podrán salir del bloque GENERATE al ADVANCE. Esto puede ser utilizado para no crear distorsiones en tiempos entre llegadas que pueden afectar la lógica del modelo. Se puede utilizar bloques idénticos varias veces en un programa.

BLOQUES DE TIEMPO CERO

Algunos bloques como el RELEASE y ADVANCE tienen las siguientes propiedades:

1. Nunca niegan la entrada a una transacción.
2. Cuando alguna transacción llega a ellos, inmediatamente intenta salir no las retienen.
3. El único camino para salir es el bloque siguiente.

A estos bloques se les llama bloques de tiempo cero, porque una transacción ejecuta una secuencia de dos o más bloques con tiempo simulado de cero. Algunos de ellos no son de este tipo, por ejemplo el bloque SEIZE; el bloque ADVANCE con un tiempo de retención mayor que cero; TERMINATE, TRANSFER y GENERATE, entre otros. El bloque SEIZE no satisface la propiedad uno, los comandos ADVANCE con un tiempo mayor que cero y TERMINATE no satisfacen la propiedad dos. Los bloques TRANSFER no satisfacen la propiedad tres. Y GENERATE no es de este tipo dado que un ocurriría error si una transacción intenta entrar al bloque GENERATE desde otro bloque.

FACILITIES DUMMY (RECURSOS FICTICIOS)

El que un servidor esté en servicio no implica que sea usado, algunas veces es de interés calcular la fracción de tiempo simulado en que un recurso fue usado productivamente. Es aquí donde una facility **dummy**, puede ser de utilidad. En forma de entidad, el bloque

SEIZE nunca niega la entrada a transacciones en el contexto del modelo. Este tipo de entidad no es utilizado para restringir recursos sino para forzar al GPSS/H a recolectar información durante una simulación, ya que de otra manera no lo haría.

USO DE BLOQUES SEIZE/RELEASE SOBREPOSICIONADOS

Algunas veces una unidad de tráfico debe capturar un recurso antes de que pueda liberar otro. Por ejemplo, en una fabrica de coches para que una estación de pintura reciba una pieza de un automóvil, la parte ya pintada debe ser sacada por un carro automático antes de que la estación pueda trabajar con la siguiente pieza, el *Programa IV.4.2* muestra el uso de este tipo de situación, Una transacción que ha ejecutado "SEIZE ESTAC1" quizá tenga que activar "SEIZE AGV" antes de que pueda aplicarse "RELEASE ESTAC1". Esto da como resultado una situación en donde el bloque RELEASE para ESTAC1 aparece después del SEIZE para AGV, y en este sentido se sobrelapan. No existen ningún problema al hacer esto en el modelo. El *Programa IV.4.2* muestra el uso de este tipo de situación:

```
            SIMULATE
*
            GENERATE  50,30
            ADVANCE   0
            SEIZE     ESTAC1
            ADVANCE   45,10
            SEIZE     AGV
            RELEASE   ESTAC1
            ADVANCE   5,2
            RELEASE   AGV
            TERMINATE 1
*
            START     50
            END
```

Programa IV.4.2 Uso de SEIZE/RELEASE sobreposicionado

En el modelo del *Programa IV.4.2* se puede analizar este tipo de sobreposición; una parte de un coche sale del bloque GENERATE y en un paso debe ser pintado en la estación de trabajo ESTAC1 y luego sacado de esa estación por medio de un carro AGV. La siguiente pieza de trabajo no puede ser pintada en la estación de trabajo hasta que la más reciente pieza de trabajo haya sido liberada por AGV y luego removida de la estación. Y por esto una pieza terminada debe entrar al bloque SEIZE AGV antes de que pueda salir del bloque RELEASE ESTAC1.

IV.5 ENTIDAD QUEUE

La *Figura IV.5.1* se muestra dos puntos arbitrarios, A y B de un camino recorrido por algunas transacciones conforme se mueven en el modelo.

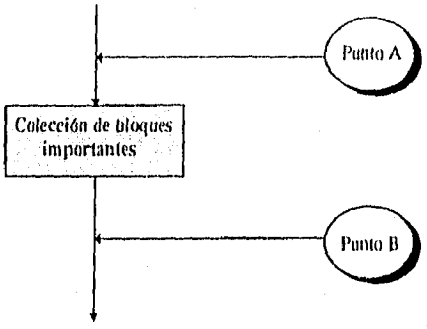


Figura IV.5.1 Lógica de la entidad Queue

Algunas respuestas a las siguiente preguntas pueden servir de utilidad al final de una simulación permitiendo la mejor comprensión de un modelo:

1. ¿Cuántas transacciones pasaron el punto A durante una simulación?
2. ¿Cuál fue el número promedio de transacciones en movimiento entre los puntos A y B?
3. ¿Cual fue el máximo número de transacciones en movimiento entre los puntos A y B?
4. ¿Cuál fue el tiempo promedio requerido para una transacción llegara del punto A al punto B?
5. ¿Cuántas transacciones pasaron un tiempo simulado de cero en movimiento entre los puntos A y B?

El GPSS/H provee una **entidad de línea de espera** *(queue entity)*, para coleccionar las respuestas a las preguntas anteriores, para lograr esto se cuenta con dos bloques QUEUE y DEPART. El uso de esta entidad requiere de escribir el bloque QUEUE en el punto A y DEPART en el punto B. El uso de estos bloques permite que GPSS/H tenga un seguimiento de cierta información de las transacciones que se mueven entre esos dos puntos de referencia y así proporcionar un resumen de esta información, como parte del resultado obtenido, al final de la simulación.

Queue es el nombre de la entidad que tiene la capacidad de coleccionar información. Tanto esta entidad como la de facility, tienen varias características en común. Pueden haber dos o más queues y facilities en un modelo, dando la flexibilidad de trabajar con cualquier número de puntos A y B distintos, con un conjunto de información separado e independiente como resultado de ese análisis. Quizá el nombre de queue puede sugerir una fila, sin embargo el uso de esta entidad es para reunir datos sobre ciertos bloques; el hecho que esta entidad se pueda utilizar como una línea de espera a la que se unen las transacciones después de ser generadas, no significa que sea necesariamente una entidad de línea de espera.

Como en las entidades facilities, dos queues diferentes, pueden distinguirse en base a su identificador o nombre, que consiste en uno o hasta ocho caracteres alfanuméricos, siendo el

primero alfabético, no se deben utilizar palabras reservadas y se pueden utilizar los mismos nombres tanto en facilities como en queues.

La mayor diferencia entre una entidad facility y un queue es que las facilities son utilizadas para modelar recursos en un sistema, es decir, servidores, mientras que las queues no son utilizadas para este propósito, sino para recopilar información. El uso de esta herramienta es opcional, y su presencia o ausencia no cambia la lógica de un modelo.

QUEUE A	DEPART A

Operando	Significado	Valores por default
A	Identificador para la entidad Queue que una transacción comienza vía QUEUE o termina vía DEPART su membresia	Error de compilación

Tabla IV.5.1 Resumen de la entidad QUEUE

Cuando las transacciones han pasado el punto A pero no han llegado al punto B, se dice que son miembros de una queue. Una transacción comienza su membresia cuando se ejecuta el bloque QUEUE y la termina cuando se activa el bloque DEPART. El operando A se utiliza para indicar el identificador de la queue relevante.

Los bloques QUEUE y DEPART son complementarios. La ejecución del segundo bloque en conjunto deshace lo hecho por el primer bloque. Tanto QUEUE como DEPART nunca niegan la entrada a una transacción, son bloques de tiempo cero.

REPORTES QUEUE

La información de los reportes emitidos por el GPSS/H se encuentra en columnas y se discutirá a continuación, la *Tabla IV.5.2* muestra dichos datos:

QUEUE	MAXIMUM CONTENTS	AVERAGE CONTENTS	TOTAL ENTRIES	ZERO ENTRIES	PERCENT ZEROS	AVERAGE TIME/UNIT	$AVERAGE TIME/UNIT	QTABLE NUMBER	CURRENT CONTENTS
COLA	9	5.739	47	1	2.1	29.304	29.941	9	

Tabla IV.5.2 Un reporte queue de un ejemplo

- La columna **QUEUE** identifica a la entidad queue para la que la información es reportada.
- **MAXIMUM CONTENTS**, muestra el máximo tamaño de la membresia de queue, es el valor mayor que se obtuvo en CURRENT CONTENTS.
- **AVERAGE CONTENTS**, muestra cuántas transacciones fueron miembros de queue, en promedio
- **TOTAL ENTRIES**, es el total de transacciones que fueron miembros.
- **ZERO ENTRIES**, es el número de miembros que duraron como tales 0.0 tiempo simulado. Una entrada cero es el resultado de una transacción que ejecuta los bloques QUEUE y DEPART a la vez y al mismo tiempo simulado.

- **PERCENT ZEROS**, es el porcentaje de transacciones que pasaron una queue con tiempo simulado cero. Es el porcentaje de TOTAL ENTRIES que fueron ZERO ENTRIES.
- **AVERAGE TIME/UNIT**, indica cuanto tiempo simulado las transacciones en promedio fueron miembros de un queue.
- **$AVERAGE TIME/UNIT**, es el promedio de duración de los miembros de queue, excluyendo los casos donde las transacciones pasaron una queue en tiempo cero.
- **QTABLE NUMBER**, muestra el identificador para Qtable que es utilizado en conjunción con queue.
- **CURRENT CONTENTS**, muestra el número de transacciones que son miembros de una queue al momento que el reporte queue es producido.

Al ejecutar el modelo en forma interactiva, esta información se puede obtener mediante el tecleo de **display que**.

MUESTRA DE LOS REPORTES DE UNA SIMULACIÓN

Los reportes de una simulación, cuando un programa es ejecutado en forma batch son guardados en un archivo. Estos reportes son guardados en un archivo de tipo texto con el mismo nombre de nuestro archivo de programa pero con la extensión lis. Estos reportes pueden ser vistos tecleado lo siguiente:

```
type nomarch.lis
```

Los reportes se archivan y pueden ser modificados en un procesador de palabras, como: Word, Word Perfect, entre otros.

IV.6 EJEMPLOS

En este apartado no se intenta dar una solución óptima al problema introductorio, únicamente se verán algunos ejemplos en base al problema y se describirá brevemente lo que se representa en el modelo con respecto al sistema.

EJEMPLO 1: SIMULACIÓN DE VEINTE TRABAJADORES

En este ejemplo se pretende simular a veinte trabajadores que llegan a ser atendidos por SOFIA, quien les recibe el acta de nacimiento, las fotos, la cartilla liberada y que además les toma la huella digital, el esquema del sistema puede ser visualizado en la *Figura IV.6.1*.

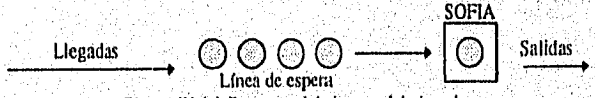


Figura IV.6.1 Esquema del sistema del ejemplo uno

Se supone que los tiempos entre llegadas son uniformes, en un intervalo de 5±4 minutos. Y también se supone que SOFIA atiende a los trabajadores en forma uniforme en un intervalo de 4±3 minutos. El programa se verá planteado en *Programa IV.6.1*.

```
*EJEMPLO I 20 TRABAJADORES
          SIMULATE
          GENERATE  5,4
          ADVANCE   0
          SEIZE     SOFIA
          ADVANCE   4,3
          RELEASE   SOFIA
          TERMINATE 1
          START     20
          END
```

Programa IV.6.1 Primer ejemplo con el uso de la entidad facility

En el *Programa IV.6.1*, existe un GENERATE, con operandos 5 y 4. En este bloque se crea a los trabajadores. Después aparece una proposición ADVANCE con operando 0, esta entidad nunca niega la entrada y sirve como un especie de fila en la que los trabajadores se forman. Después se verá que esta estructura puede ser reemplazada con la entidad queue. Posteriormente aparece el bloque SEIZE, aquí las personas piden el servicio a SOFIA. El siguiente ADVANCE sirve para simular el tiempo que SOFIA tarda en atender a los trabajadores. El bloque RELEASE significa que SOFIA ha terminado de atender a las personas, el bloque TERMINATE hace que los trabajadores salgan del sistema. El operando con valor 1 del bloque TERMINATE significa que el contador disminuye en uno. Los comandos de control START con el operando 20, significa que en la simulación SOFIA recibirá a 20 trabajadores.

Al correr la simulación en forma interactiva, en un archivo con extensión lis. se obtiene información sobre SOFIA, los requerimientos en bytes del programa, el tiempo de simulación relativo y absoluto, en este caso la simulación se llevó a cabo en aproximadamente una hora y 45 minutos, se puede analizar también cuantas transacciones pasaron por los distintos bloques al terminar la simulación y cuantas había en ellas, entre otras cosas. También aparece el reporte de la entidad facility, el cual se muestra en la *Tabla IV.6.1*.

	--AVG-UTIL-DURING--								
FACILITY	TOTAL TIME	AVAIL TIME	UNAVL TIME	ENTRIES	AVERAGE TIME/XACT	CURRENT STATUS	PERCENT AVAIL	SEIZING XACT	PREEMPTING XACT
SOFIA	0.853			20	4.496	AVAIL			

Tabla IV.6.1 La información recolectada de la entidad facility del ejemplo uno

SOFIA estuvo trabajando un 85.3% del total del tiempo que duro la simulación. Atendió a 20 trabajadores y dedicó a cada solicitud un promedio de 4.496 minutos.

EJEMPLO 2: SIMULACIÓN DE CUATRO HORAS DE TRABAJO

En este ejemplo se pretende simular cuatro horas de trabajo continuas. Se plantea que los tiempos entre llegadas son los mismos que en el ejemplo anterior, también el mismo tiempo de servicio de SOFIA. La estructura del programa se muestra en el *Programa IV.6.2*.

```
*EJEMPLO 2 DE CUATRO HORAS DE TRABAJO
        SIMULATE
        GENERATE  5,4
        ADVANCE   0
        SEIZE     SOFIA
        ADVANCE   4,3
        RELEASE   SOFIA
        TERMINATE
        GENERATE  240
        TERMINATE 1
        START     1
        END
```

Programa IV.6.2 Ejemplo dos hecho en GPSSH

Este programa es muy parecido al del ejemplo anterior. Las únicas diferencias son la inclusión de un bloque GENERATE al final del programa, existen dos bloques TERMINATE, uno sin el operando y otro con el operando con valor uno y el bloque START tiene un operando con valor de uno. El bloque GENERATE 240, genera una transacción al momento 240, el primer TERMINATE, tiene por default el valor cero, es decir, al momento que el trabajador se aleja del sistema, el contador no se decrementa, el segundo TERMINATE tiene el operando con valor uno, pues en el momento en que se genere una transacción en el bloque GENERATE 240, la simulación habrá terminado y el contador se decrementará en uno, y esto dará fin a la simulación, pues en el comando START se puso como operando el valor uno.

Al solicitar información se obtienen prácticamente los mismos resultados que en el ejemplo anterior, la única variante fue el tiempo de simulación que en este caso fueron cuatro horas, el reporte de la entidad facility, aparece a continuación *Tabla IV.6.2*.

	--AVG UTIL DURING--								
FACILITY	TOTAL TIME	AVAIL TIME	UNAVL TIME	ENTRIES	AVERAGE TIME/XACT	CURRENT STATUS	PERCENT AVAIL	SEIZING XACT	PREEMPTING XACT
SOFÍA	0.808			48	4.038	AVAIL.			

Tabla IV.6.2 Los resultados del ejemplo dos

El tiempo total de trabajo para SOFIA fue del 80.8%, es decir que estuvo desocupada un 19.2% del total del tiempo. Atendió a 48 trabajadores y dedicó a cada trabajador un promedio de 4.038 minutos.

EJEMPLO 3: SIMULACIÓN DE DOS SERVIDORES EN SERIE, CON CUATRO HORAS

En este ejemplo se pretende simular cuatro horas de trabajo continuas suponiendo que después de ser atendidos por SOFIA quien recibe los papeles necesarios y les toma la huella, los trabajadores deben pasar con MARI, quien elabora un recibo, el cual debe ser llevado a pagar a la caja. El problema puede ser analizado en base al esquema de la *Figura IV.6.3*.

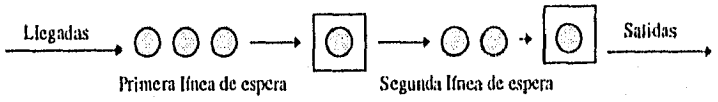


Figura IV.6.3 Esquema del sistema del ejemplo tres

Se supone que los tiempos entre llegadas son los mismos que en el ejemplo anterior. Se supondrán los mismos tiempos de atención de SOFIA, en cuanto al tiempo de servicio de MARI se supone que se tarda aproximadamente 4±2 minutos. El programa planteado se muestra en *Programa IV.6.3*.

```
*EJEMPLO 3 DOS SERVIDORES EN SERIE
        SIMULATE
        GENERATE 5,4
        ADVANCE  0
        SEIZE    SOFIA
        ADVANCE  4,3
        RELEASE  SOFIA
        ADVANCE  0
        SEIZE    MARI
        ADVANCE  4,2
        RELEASE  MARI
        TERMINATE
        GENERATE 240
        TERMINATE 1
        START    1
        END
```

Programa IV.6.3 Programa del ejemplo tres

Este programa es muy parecido al del ejemplo anterior. El bloque ADVANCE 0 aparece después del RELEASE SOFIA, en este bloque el trabajador se vuelve a formar después de haber sido atendido por SOFIA para ser atendido por MARI. Se usa la misma lógica que para el servidor SOFIA.

Al ejecutar la simulación, se obtiene los datos de las dos recepcionistas SOFIA y MARI, en el reporte de facilities, *Tabla IV.6.3*.

FACILITY	--AVG-UTIL-DURING-- TOTAL TIME	AVAIL TIME	UNAVL TIME	ENTRIES	AVERAGE TIME/XACT	CURRENT STATUS	PERCENT AVAIL	SEIZING XACT	PREEMPTING XACT
SOFIA	0.938			50	4.501	AVAIL		51	
MARI	0.830			48	4.150	AVAIL		49	

Tabla IV.6.3 La información obtenida de las dos recepcionistas para el ejemplo tres

El tiempo total de trabajo para SOFIA fue de 93.8% es decir que estuvo desocupada un 6.2% del total del tiempo. Atendió a 50 trabajadores y le dedicó a cada trabajador un promedio de 4.501 minutos, y cuando la simulación finalizó se encontraba atendiendo al trabajador 51. En cuanto a MARI, trabajó del total de tiempo un 83%, y estuvo desocupada un 17%. Atendió a 48 trabajadores y les dedicó en promedio 4.15 minutos a cada trabajador, al momento terminar con la simulación se encontraba atendiendo al trabajador 49.

EJEMPLO 4: SIMULACIÓN DE DOS SERVIDORES, PARA QUE EL CLIENTE TERMINE DE SER ATENDIDO DEBE SOLICITAR DOS DIFERENTES SERVICIOS

En este ejemplo se pretende simular cuatro horas de trabajo continuas y se supone que antes de terminar el ser atendidos por SOFIA quien recibe y revisa la solicitud, el acta de nacimiento y la cartilla liberada, deben ser atendidos por MARI quien les toma una muestra de la huella digital. El esquema del sistema puede ser visto en la *Figura IV.6.4*.

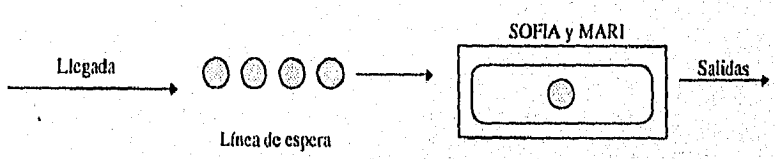


Figura IV.6.4 Esquema del sistema del ejemplo uno

Los tiempos entre llegadas son los mismos que los del ejemplo anterior, así como el tiempo de servicio de SOFIA y MARI. El programa del modelo anterior se muestra en el *Programa IV.6.4*.

```
*EJEMPLO 4 DOS SERVIDORES TRABAJAN
*CONJUNTAMENTE
        SIMULATE
        GENERATE 5,4
        ADVANCE  0
        SEIZE    SOFIA
        ADVANCE  4,3
        SEIZE    MARI
        RELEASE  SOFIA
        ADVANCE  4,2
        RELEASE  MARI
        TERMINATE
        GENERATE 240
        TERMINATE 1
        START    1
        END
```

Programa IV.6.4 Programa del ejemplo cuatro

Este ejemplo es muy parecido al anterior excepto por el hecho de que antes de terminar de ser atendido un trabajador por SOFIA debe ser atendido por MARI, esto se hace al poner el bloque SEIZE MARI después del bloque ADVANCE 4,3 dentro de la entidad facility de SOFIA. La cantidad de trabajadores que SOFIA atiende se basa en el número de los que atiende MARI. Se puede suponer que existirá una distorsión en cuanto al tiempo que se tarda en atender SOFIA a un trabajador.

Al llevar a cabo la simulación se obtuvo la información de SOFIA y de MARI, la cual es resumida en el reporte de facility en la *Tabla IV.6.4*.

	--AVG-UTIL-DURING--								
FACILITY	TOTAL TIME	AVAIL TIME	UNAVL TIME	ENTRIES	AVERAGE TIME/XACT	CURRENT STATUS	PERCENT AVAIL	SEIZING XACT	PREEMPTING XACT
SOFIA	0.987			43	5.508	AVAIL		44	
MARI	0.710			42	4.056	AVAIL		49	

Tabla IV.6.4 La información recabada durante la simulación de SOFIA y MARI

El tiempo total de trabajo para SOFIA fue de 98.7% es decir que estuvo desocupada un 1.3% del total del tiempo. Atendió a 43 trabajadores y dedicó a cada trabajador un promedio de 5.508 minutos, en este valor se puede analizar la distorsión en cuanto al tiempo que tarda SOFIA por trabajador, cuando la simulación terminó se encontraba atendiendo al trabajador 44. El tiempo que dedicó a los trabajadores SOFIA aumentó, pues tenían que pasar con MARI antes. En cuanto a MARI trabajo del total de tiempo un 71%, y estuvo desocupada un 19%. Atendió a 42 trabajadores y les dedicó en promedio 4.056 minutos.

EJEMPLO 5: SIMULACIÓN CON USO DE LA ENTIDAD QUEUE

Este modelo es el mismo del ejemplo dos, pero con el uso de la entidad queue, como se recuerda esta entidad sirve para analizar el número de transacciones que pasan por allí y para recolectar información que sin esta entidad no sería posible. El programa se muestra en el *Programa IV.6.5*.

```
*EJEMPLO 5 USO DE QUEUE
           SIMULATE
           GENERATE  5,4
           QUEUE     COLA
           SEIZE     SOFIA
           DEPART    COLA
           ADVANCE   4,3
           RELEASE   SOFIA
           TERMINATE
           GENERATE  240
           TERMINATE 1
           START     1
           END
```

Programa IV.6.5 Programa del ejemplo cinco

Los bloques que se desean analizar se ponen entre el bloque QUEUE y DEPART. En este caso fue SEIZE SOFIA, al ponerlos aquí se desea analizar la línea de espera para ser atendidos por SOFIA. El bloque QUEUE sirvió como el bloque ADVANCE 0 que anteriormente se utilizó para recibir a las transacciones generadas, recordando que esta entidad jamás niega la entrada.

Al llevar a cabo la simulación, se obtuvo los datos de la entidad queue, en la *Tabla IV.6.5*.

QUEUE	MAXIMUM CONTENTS	AVERAGE CONTENTS	TOTAL ENTRIES	ZERO ENTRIES	PERCENT ZEROS	AVERAGE TIME/UNIT	$AVERAGE TIME/UNIT	QTABLE NUMBER	CURRENT CONTENTS
COLA	2	0.221	48	18	37.5	1.105	1.768	0	

Tabla IV.6.5 Resultados de la entidad queue

El máximo de personas que esperaron a SOFIA fue de dos. 18 personas no tuvieron que esperar para ser atendidos, es decir, el 37.5% del total de trabajadores. Sin tomar en cuenta las personas que fueron atendidas inmediatamente, en promedio los trabajadores esperaron casi dos minutos, en general, tomando en cuenta a los 48 trabajadores atendidos, el promedio de tiempo de espera de cada persona en la fila fue de un minuto.

CAPÍTULO V
MODELOS CON GRUPOS IDÉNTICOS DE SERVIDORES

V.1 UN SOLO SERVIDOR Y GRUPOS DE SERVIDORES

Algunos sistemas de servicio sólo incluyen un mecanismo de atención como por ejemplo, el caso de Sofía, la recepcionista de pasaportes. Sin embargo, puede suceder que existan dos recepcionistas que trabajen de la misma manera. La entidad facility, como se ha visto, puede ser usada para modelar un solo servidor pero existe una entidad denominada *storage* que permite modelar grupos de servidores idénticos.

Por ejemplo, si se supone que en la recepción de pasaportes existen dos personas brindando el servicio: Sofía y Mari, quienes son distintas como personas pero como recepcionistas operan de la misma manera, haciendo el mismo tipo de trabajo, al mismo ritmo, desde el punto de vista de un trabajador que va a Canadá no existe diferencia entre si Sofía o Mari lo atienden, es decir, estas dos personas son servidoras idénticas. Ahora, si se supone que Mari y Sofía no trabajan de la misma manera, quizá Sofía tenga más experiencia que Mari en el trato con los trabajadores, dando como resultado una mayor rapidez al atender a los trabajadores. En este caso al cliente si le importará quien lo atiende, esperando ser atendido con mayor rapidez.

Con la entidad de almacenaje (*storage*), el GPSS/H podrá modelar dos o más servidores idénticos, cuando estos no lo son, esta entidad no podrá ser utilizada para modelarlos, esto sucede dado que una entidad storage no contiene identidades individuales. Cuando una transacción entra en servicio con alguno de estos servidores, la transacción no sabrá si es atendida por el servidor uno o por el servidor dos. Para modelar servidores no idénticos se puede realizar una aproximación a la realidad utilizando la entidad facility.

V.2 UTILIZACIÓN DE SERVIDORES EN UN GRUPO DE SERVIDORES IDÉNTICOS: BLOQUES ENTER Y LEAVE

La forma de representar un servidor estructurado por un grupo de recepcionistas con la entidad de almacenaje (*storage*), es similar a la usada con servidores únicos de la entidad facility; la lógica: si una transacción captura a un servidor, recibe el servicio y luego sale de ahí, es aplicada de la misma manera para servidores idénticos (*Figura V.2.1*), es decir:

1. Una transacción llega a un punto donde un grupo de servidores es modelado y pide ser atendida.
2. Una transacción espera su turno, si es necesario, para ser atendida.
3. Cuando llega su turno, la transacción *captura* a un servidor.
4. La transacción mantiene al servidor en estado de captura durante algún intervalo de tiempo mientras el servicio es llevado a cabo.
5. Finalmente la transacción libera al servidor y continua con el sistema.

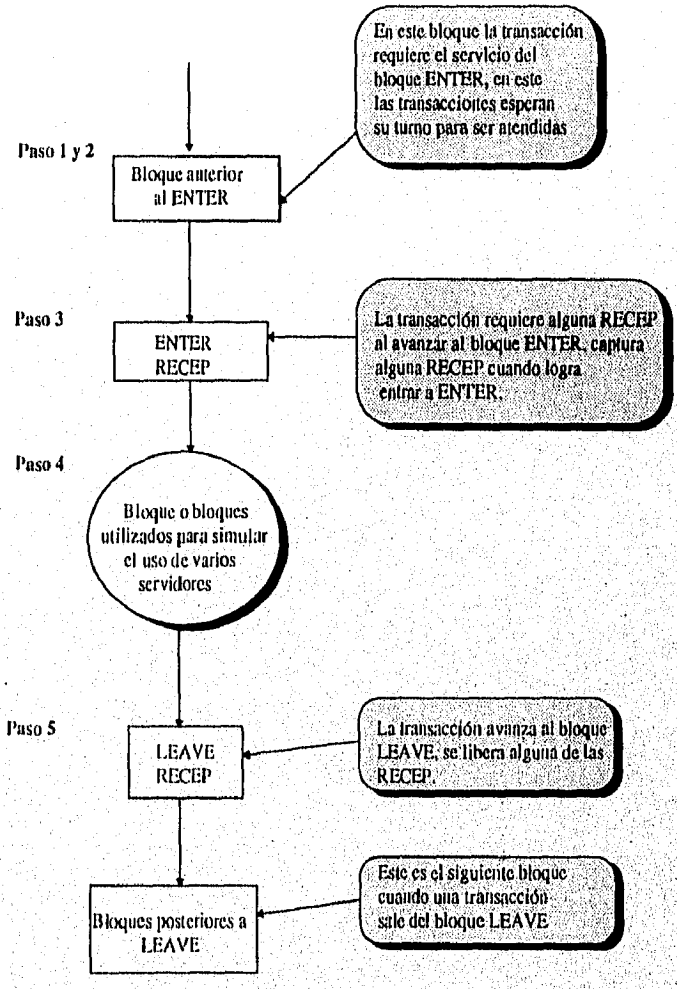


Figura V.2.1 Localización de los bloques ENTER/LEAVE

Su uso se puede visualizar en la *Figura V.2.1*, la entidad storage contiene los bloques ENTER y LEAVE. La diferencia entre la entidad storage y facility es que en vez de utilizar SEIZE se utiliza ENTER, y en el caso de RELEASE se usa LEAVE. El bloque ENTER puede negar la entrada a cierta transacción. Se puede notar que la palabra RECEP se asocia con los bloques ENTER y LEAVE para identificar a la entidad. Cada bloque ENTER y LEAVE contienen un operando A. El papel del operando A es para establecer la identidad de un grupo en particular de servidores (*storage*), donde una transacción pide el control al tratar de ejecutar el bloque ENTER y deja el control al ejecutar el bloque LEAVE. Como el operando A para SEIZE, RELEASE, QUEUE y DEPART, el operando A para los bloques ENTER y LEAVE indica una entidad storage específico que modela el grupo de servidores de interés que las transacciones ejecutan en los bloques ENTER y LEAVE.

Al igual que facilities y queues, algunas distinciones deben ser hechas entre storages dando a cada grupo de servidores idénticos su propio identificador. Estos identificadores pueden consistir de uno a ocho caracteres alfanuméricos siendo el primero alfabético. El nombre utilizado para identificar a facilities, queues o storages o etiquetas de bloques, puede consistir de al menos tres caracteres alfabéticos y se debe evitar el uso de palabras reservadas para GPSS/H, también se pueden utilizar números enteros positivos para identificar a estos. Es factible utilizar el mismo identificador para storages, facilities y queues, pero no se puede usar el mismo para la etiquetación de bloques. Es importante visualizar que un identificador para un storage es el descriptor de un grupo específico de servidores en un modelo y no para cada servidor en grupo de ellos.

Operando	Significado	Valores por default o resultado
A	Identificador para la entidad storage que es capturado en el bloque ENTER y liberado en bloque LEAVE	Un error de compilación

Tabla V.2.1 Resumen de la entidad storage, con operando A

CAPTURANDO Y LIBERANDO MÁS DE UN SERVIDOR EN UN GRUPO A LA VEZ

Los bloques ENTER y LEAVE tienen el operando opcional B, éste indica cuántos servidores son capturados y cuántos servidores son liberados cuando una transacción pide un servicio, su valor por default es uno.

ENTER A,B	LEAVE A,B

Operando	Significado	Valores por default o resultado
A	Identificador para la entidad storage que es capturado en el bloque ENTER y liberado en bloque LEAVE	Un error de compilación
B	Número de servidores que son capturados o requeridos en ENTER y liberados en LEAVE	1

Tabla V.2.2 Resumen de la entidad storage, con operando A y B

V.3 Especificación del Tamaño de Grupo: Comando de Control STORAGE

El número de servidores a ser modelados en cierta entidad storage es indicado al incluir en el programa un comando de control STORAGE, la forma como se debe incluir en el programa este comando se muestra en la *Tabla V.3.1*. El identificador del *storage* se escribe en el campo de etiquetación, la palabra STORAGE es tecleada en el campo de operación. El operando A, usualmente es un entero constante que indica la capacidad de la entidad, es decir, el número de servidores para esa entidad.

1	Etiqueta (2-9)	Operación (11-20)	Operando (22-72)
	Ver abajo	STORAGE	A

Operando o Etiqueta	Significado	Valores por default o resultado
Etiqueta	Identificador para la entidad storage donde la capacidad será definida	Causa un error de compilación
A	Capacidad de la entidad storage (número de servidores modelados)	Causa un error de compilación

Tabla V.3.1 Resumen del uso de STORAGE y su localización en un programa

En la *Tabla V.3.2* se muestran algunos ejemplos del uso de storage, el primero contiene a cinco servidores idénticos y el segundo a 10,

```
RECEP    STORAGE 5 La entidad identificada con RECEP contiene 5 recepcionistas
DOCTOR   STORAGE 10 La entidad identificada con DOCTOR contiene 10 doctores
```

Tabla V.3.2 Algunos ejemplos del uso de STORAGE

Como todos los comandos de control, el comando STORAGE es ejecutado por el GPSS/H después que el archivo del modelo haya sido compilado. Este comando se escribe antes que la fase de movimiento de transacciones de inicio, después de compilar, GPSS/H ejecuta todos los comandos de control que aparecen antes del comando START, obviamente el

comando STORAGE debe ser puesto antes del comando START, una forma de ilustrarlo puede ser la *Figura V.3.1*:

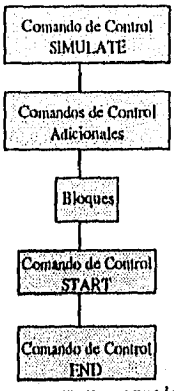


Figura V.3.1 Muestra como un programa puede ser escrito en GPSSIH

Si el usuario no define la capacidad de la entidad de almacenaje, entonces por default el valor de esta será 2,147,483,647, infinito para este caso.

FACILITIES CONTRA STORAGES DE CAPACIDAD UNO

Una facility modela un solo servidor, al igual que un *storage* con capacidad uno, permitiendo así tener dos alternativas para modelar servidores con una sola entidad de atención. La diferencia en el uso de una y de otra, es que las facilities son más sofisticadas que las entidades *storages*, un ejemplo es el caso cuando una transacción llega y le quita el servicio a otra, esto sólo puede ser modelado con la entidad facility. Otro caso es el concepto de disponibilidad y no disponibilidad, que se encuentra más elaborado en la entidad facility. Una tercer diferencia es el concepto de posesión que se aplica a un solo servidor y no a un grupo de servidores, cuando una transacción se convierte en el propietario de una facility se graba el identificador de la transacción que fue capturada. En contraste, cuando una transacción llega a una entidad storage no se graba el identificador de la transacción que entra a ella.

EL VOCABULARIO UTILIZADO CON LA ENTIDAD STORAGES

Se utiliza *storage* como nombre de la entidad de almacenaje, y el término *capacidad* para indicar cuantos servidores se encuentran en el grupo modelado. Existe también otro vocabulario asociado:

1. Una storage se dice que está llena **(full)** si todos los servidores se encuentran capturados, es decir, en servicio.
2. Una storage se dice que esta vacía **(empty)** si todos los servidores se encuentran libres.

3. El contenido actual (current contenls) de una storage es el número de servidores que se encuentran en servicio.
4. La capacidad restante (remaining capacity) de un storage es el número de servidores que se encuentran libres.

TIPO DE SISTEMA DE LÍNEA QUE STORAGE MODELA

En forma general, las transacciones que esperan a un servidor en un grupo de estos pueden aguardar en una o en dos o más líneas de espera. La *Figura V.3.2* muestra la geometría de una línea con múltiples servidores, en contraste se puede visualizar la forma de líneas múltiples con servidores múltiples (*Figura V.3.3*). Generalmente el uso de una línea con múltiples servidores es utilizada en bancos, aeropuertos, salas de belleza, oficinas de pasaportes, etc. Los sistemas de líneas múltiples con servidores múltiples son utilizados frecuentemente en tiendas de autoservicio como Aurrerá, Comercial Mexicana, Gigante, entre otras.

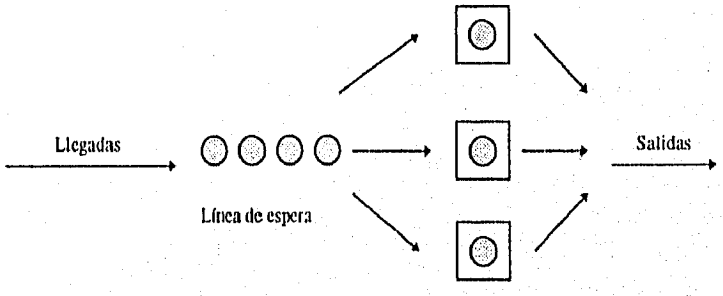


Figura V.3.2 Una línea con múltiples servidores

La ventaja de una línea con múltiples servidores es que el servicio se realiza con la regla de que el primero en llegar será el primero en ser atendido, esto no sucede en el caso para líneas múltiples con servidores múltiples. Si líneas múltiples son utilizadas en un banco, un cliente puede quedarse atorado detrás de otros que desean abrir una cuenta o que depositan dinero y que tardan más tiempo del *normal*. Es por esto que una línea evita un exceso de tiempo de espera para algunos clientes.

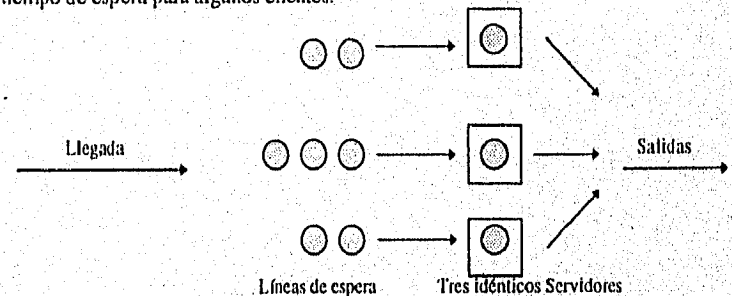


Figura V.3.3 Sistema de múltiples líneas con múltiples servidores

Un sistema de una línea con múltiples servidores puede ser simulado con el uso de una entidad de almacenaje. El modelar múltiples líneas con múltiples servidores, requiere que las transacciones tengan alguna forma de identificar a un servidor en particular, pero si se recuerda, con el uso de la entidad de almacenaje, no existe control para identificar a los servidores, es por ésto que puede simular modelos de líneas múltiples con servidores múltiples; este tipo de sistemas es simulado con la entidad facility.

REPORTES DE LA ENTIDAD DE ALMACENAJE

Durante una simulación, el lenguaje almacena cierta información de cada entidad storage en un modelo y produce un reporte, esta información se encuentra clasificada como se ve en la *Tabla V.3.3*:

STORAGE	--AVG-UTIL-DURING-- TOTAL TIME	AVAIL TIME	UNAVL TIME	ENTRIES	AVERAGE TIME/UNIT	CURRENT STATUS	PERCENT AVAIL	CAPACITY	AVERAGE CONTENTS	CURRENT CONTENTS	MAXIMUM CONTENTS
RECEP	0.407			48	4.070	AVAIL	100.0	2	0.814	1	2

Tabla V.3.3 Reporte de un ejemplo de la entidad storage

- La columna **STORAGE** muestra al identificador de la entidad storage.
- **--AVG-UTIL-DURING-- TOTAL TIME**, muestra la fracción del total de tiempo simulado que la entidad *storage* estuvo en estado de servicio.
- **--AVG-UTIL-DURING-- AVAIL TIME**, muestra la fracción del total de tiempo de disponibilidad en tiempo simulado que storage estuvo en estado de captura, es decir, en servicio. Recordando que la disponibilidad significa que la entidad storage estuvo dispuesta a trabajar.
- **--AVG-UTIL-DURING-- UNAVL TIME**, muestra la fracción del total de tiempo de no disponibilidad en tiempo simulado.
- **ENTRIES**, indica el número de transacciones que el grupo de servidores atendió durante la simulación.
- **AVERAGE TIME/UNIT**, muestra el tiempo promedio de retención por servidor en cada atención a una transacción.
- **CURRENT STATUS**, indica si la entidad storage quedó finalmente disponible a trabajar, es decir, AVAIL si estuvo disponible y UNAVAIL si no estuvo disponible.
- **PERCENT AVAIL**, muestra la fracción del tiempo total simulado que storage estuvo disponible para dar servicio.
- **CAPACITY** marca la capacidad de cada entidad storage en el modelo, es decir, el número de servidores idénticos.
- **AVERAGE CONTENTS** muestra cuántos servidores estuvieron en servicio en promedio durante el curso de la simulación.
- **CURRENT CONTENTS** muestra cuántos servidores se encuentran en servicio al momento de finalizar la simulación.
- **MAXIMUM CONTENTS** rotula el valor máximo tomado por CURRENT CONTENTS en el transcurso de la simulación.

Para ver esta información en modo test se teclea lo siguiente: display sto.

ESTIMACIÓN DE MÍNIMO Y MÁXIMO NÚMERO DE SERVIDORES

La entidad storage puede ser usada para estimar el mínimo número de servidores requeridos para mantener el equilibrio entre la demanda y la oferta y poder encontrar un número óptimo de servidores como se muestra en la *Figura IV.1.1*. También se puede estimar el máximo número de servidores sin restringir al sistema, en este caso los costos serían altos y los costos de espera mínimos. La técnica es modelar un sistema utilizando una entidad *storage* con su valor por default de capacidad. Al finalizar la simulación, el reporte de storage mostrará el promedio (AVERAGE CONTENTS) y el máximo (MAXIMUM CONTENTS) número de servidores capturados durante la simulación. El siguiente entero del promedio de AVERAGE CONTENTS sirve para determinar el valor mínimo de servidores para mantener el equilibrio.

Si se supone que se desea saber cuántas recepcionistas se necesitan para la oficina de pasaportes, tomando en cuenta que las personas que desean trabajar en Canadá llegan cada 2±1 minutos, y que las recepcionistas se tardan en cada trabajador 10±5 minutos, el modelo de este sistema sería el que se muestra a continuación se simularán cuatro horas continuas de trabajo, *Programa V.3.1*:

```
* SIMULACION DE DOS SERVIDORES IDENTICOS
        SIMULATE
RECEP   STORAGE   2147483647
        GENERATE  2,1
        QUEUE     COLA
        ENTER     RECEP
        DEPART    COLA
        ADVANCE   10,5
        LEAVE     RECEP
        TERMINATE
        GENERATE  240
        TERMINATE 1
        START     1
        END
```

Programa V.3.1 Programa que estima el mínimo y máximo de número de servidores

Los resultados obtenidos del reporte de la entidad storage sería el siguiente (*Tabla V.3.4*):

STORAGE	-AVG UTIL DURING- TOTAL TIME	AVAIL TIME	UNAVL TIME	ENTRIES	AVERAGE TIME/UNIT	CURRENT STATUS	PERCENT AVAIL	CAPACITY	AVERAGE CONTENTS	CURRENT CONTENTS	MAXIMUM CONTENTS
RECEP	0.000			121	9.980	AVAIL	100.0	2147483647	5.032	7	9

Tabla V.3.4 Reporte de información de una entidad storage

Para mantener el equilibrio de costos el valor mínimo de servidores es 6 servidores, así como el máximo número de servidores sin restringir al sistema es 9. En el *Programa V.3.1* el

operando del comando STORAGE tiene el valor de 2147483647, pero se puede utilizar otro número menor.

DUMMY STORAGES

La idea de *dummy facility* ya fue introducida anteriormente, cuando se explicó que un dummy facility es una entidad donde el bloque SEIZE nunca niega la entrada a las transacciones. Estas entidades no son utilizadas para restringir los recursos, sino para recolectar información de la simulación. El valor de este dummy está en el reporte que se emite al final de la simulación.

La entidad dummy storage es análoga a la *dummy facility*. Un *dummy storage* es aquel donde el bloque ENTER nunca niega la entrada a transacciones de la misma forma en que se utiliza *storage*. Al igual que el dummy facility, no restringe recursos, sino que forza al lenguaje a recolectar más información durante una simulación.

ANIDADO DE LOS BLOQUES ENTER Y LEAVE

La acción de anidar los bloques SEIZE y RELEASE ya fue mencionada anteriormente, la idea básica de este tipo de acción es que una transacción quizá tenga que haber pasado por cierto servicio como requisito para ser liberada por otro servicio. Como resultado, el bloque RELEASE se coloca en el programa en alguna parte después del bloque SEIZE de otro servidor; esta idea puede ser extendida a un grupo de servidores idénticos con los bloques ENTER y LEAVE.

ADMITIENDO TRANSACCIONES EN UN MODELO SÓLO CUANDO SON NECESARIAS

Algunas veces, al momento de construir un modelo, las transacciones deben ser admitidas en ciertos instantes de una simulación. Esta circunstancia sobresale cuando las transacciones deben ser aceptadas sólo cuando una condición o varias condiciones se cumplen. Si estas condiciones se realizan con tiempos probabilísticos, entonces es muy difícil o imposible estructurar un bloque GENERATE que introduzca transacciones al modelo en el momento preciso.

Este tipo de situaciones pueden ser resueltas con el uso de un bloque GENERATE en blanco, seguido de un bloque que condicionalmente puede negar la entrada, tales como el bloque ENTER o SEIZE. Un bloque GENERATE en blanco es aquel donde el operando A tiene el valor de cero. Este bloque produce transacciones durante toda la simulación, pero esto no ocurre cuando eventualmente estas transacciones llegan a un punto donde el siguiente bloque les niega la entrada, el bloque GENERATE deja de proporcionarlas. Si esto ocurre, la transacción que trata de salir del bloque GENERATE se encontrará bloqueada, deberá permanecer en el bloque GENERATE. Cuando eventualmente pueda salir del bloque GENERATE en el momento en que otra transacción sea necesitada por el modelo, se

reemplaza a sí misma al ejecutar el bloque, lo que significará que otra transacción será creada y llevada al bloque GENERATE.

Si supone que en la oficina de pasaportes, un policía deja pasar a las personas sólo cuando la recepcionista deja de atender a la que estaba atendiendo, el *Programa V.3.2* muestra la situación anterior.

```
* SIMULACION DE GENERATE EN BLANCO
        SIMULATE
        GENERATE
        SEIZE       RECEP
        ADVANCE     3,2
        RELEASE     RECEP
        TERMINATE
        GENERATE    240
        TERMINATE   1
        START       1
        END
```

Programa V.3.2 Programa en donde se utiliza el GENERATE en blanco

Los resultados de la entidad facility fue la siguiente (*Tabla V.3.5*):

	--AVG-UTIL-DURING--								
FACILITY	TOTAL TIME	AVAIL TIME	UNAVL TIME	ENTRIES	AVERAGE TIME/XACT	CURRENT STATUS	PERCENT AVAIL	SEIZING XACT	PREEMPTING XACT
RECEP	1.000			80	3.000	AVAIL		81	

Tabla V.3.5 Reporte de información de una entidad facility

Cada tres minutos una transacción fue creada para entrar al modelo y así durante el periodo de la simulación llegaron 81 personas y fueron atendidas 80.

V.4 USOS DEL BLOQUE TRANSFER

MODO CONDICIONAL DEL BLOQUE TRANSFER

Es posible que el siguiente bloque de una transacción sea seleccionado probabilísticamente en base a dos alternativas, cuando existen más alternativas se debe definir una función, esto se verá más adelante en el capítulo VI. Por ejemplo, las transacciones pueden escoger a uno de dos bloques con un 25 % de posibilidades para avanzar a un bloque B y con un 75% para irse a un bloque C, durante todo el tiempo de la simulación, puede ser llevado a cabo con el uso del bloque en modo condicional TRANSFER, que probabilísticamente separa al tráfico de transacciones en dos partes.

Las características de este TRANSFER se mencionan en la *Tabla V.4.1*. Como se puede ver antes del operando A existe un punto decimal, el resto del operando es un valor que se interpreta como partes por millar. Los operandos B y C son utilizados para identificar a los

dos bloques alternos del modelo. Esta identificación se lleva a cabo al etiquetar cada uno de los dos bloques alternos y repitiendo estas etiquetas en los operandos B y C. El operando B señala al *bloque B* y el C al *bloque C*. El operando A indica la fracción de tiempo que una transacción, al llegar al bloque TRANSFER, tomará al bloque C como su siguiente bloque, como se muestra en la *Figura V.4.1*.

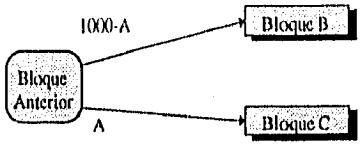


Figura V.4.1 Función de un bloque TRANSFER condicional

Cuando el valor A es constante, se recomienda utilizar el valor con tres dígitos. Cuando una transacción ejecuta este bloque saca un valor aleatorio de una población de enteros distribuidos uniformemente con el intervalo cerrado de 0 a 999, si el valor sacado es menor que el operando A, el siguiente bloque será C y si es mayor será B.

TRANSFER ,A,B,C

Operando	Significado	Valores por default o resultado
A	Fracción del tiempo que el bloque C será el siguiente bloque de una transacción (ejemplo: .352, .456)	Un error de compilación
B	Señala a un bloque, debe ser la etiqueta correspondiente a ese bloque	El bloque será el siguiente en el programa
C	Señala a un bloque, debe ser la etiqueta correspondiente a ese bloque	El bloque será el siguiente en el programa

Tabla V.4.1 Resumen del bloque TRANSFER condicional

Un ejemplo puede ser el siguiente:

TRANSFER ,250, TRABA, JUEGO

Cuando una transacción llega a este bloque, una de cuatro (250 de 1000) veces el siguiente bloque será JUEGO, y tres de cuatro veces su siguiente bloque será TRABA. La determinación del siguiente bloque es independiente de las opciones anteriores, es decir, que es posible que cinco veces consecutivas las transacciones que llegan al bloque TRANSFER ,250, TRABA, JUEGO tomen como su siguiente bloque etiquetado con JUEGO, el valor de A es tan solo un promedio de que un 25 % de las transacciones decidirá a JUEGO. Si el valor del operando A es cero, entonces la transacción siempre avanzará al bloque B, también si el operando es mayor que 1000, la transacción será mandada al bloque C, en ambos casos se consideran como un TRANSFER incondicional. Se puede omitir el operando B si el

bloque B es el siguiente en el programa después del bloque TRANSFER. Este bloque nunca niega la entrada a las transacciones, instantáneamente tratará la transacción de avanzar al siguiente bloque, cuando el siguiente bloque puede negar su entrada, entonces permanece ahí, hasta volver a intentar salir de él. Más adelante se verá un ejemplo del uso de este tipo de bloque.

EL MODO BOTH DEL BLOQUE TRANSFER

Cuando se utiliza un TRANSFER condicional, se determina probabilísticamente el siguiente bloque. La transacción que llega al TRANSFER condicional toma la decisión de su siguiente bloque, aunque inicialmente le niegue la entrada, seguirá intendando entrar a ese mismo bloque. En contraste con este bloque, el modo BOTH manda a una transacción al primer bloque de las dos opciones que acepte n la transacción. En la sección V.6 se verá un ejemplo de este tipo de bloque.

La forma general del modo BOTH del bloque TRANSFER se muestra en la *Tabla V.4.2*, el operando A es la palabra BOTH. El operando B y C son utilizados para señalar a los otros dos bloques en el modelo. El señalamiento se lleva a cabo etiquetando cada una de las dos opciones de bloques siguientes y utilizar esas etiquetas como los operandos B y C. La transacción que llega al bloque BOTH TRANSFER inmediatamente tratará de entrar al bloque B, si este le niega la entrada, entonces intentará entrar al bloque C si también este le niega el acceso, entonces dejará de moverse por un momento y permanecerá allí, hasta volver a intentar avanzar.

TRANSFER A,B,C

Operando	Significado	Valores por default o resultado
A	Utilizar la palabra BOTH	Un error de compilación
B	Señala a un bloque, debe ser la etiqueta correspondiente a ese bloque	El bloque será el siguiente en el programa
C	Señala a un bloque, debe ser la etiqueta correspondiente a ese bloque	El bloque será el siguiente en el programa

Tabla V.4.2 Resumen del bloque TRANSFER BOTH

EL MODO ALL DEL BLOQUE TRANSFER

Este bloque es una forma extendida del bloque BOTH TRANSFER, en el bloque BOTH la transacción espera hasta ser aceptada por alguno de dos bloques alternos. En el bloque ALL TRANSFER una transacción espera hasta que sea aceptada por dos o más bloques siguientes alternos. El operando A es la palabra ALL, el operando B señala al bloque B, el cual es el primero en el conjunto de bloques siguientes. El operando C señala al bloque C, el cual es el último del conjunto del bloques alternos siguientes. No todos los bloques entre B y C serán intentados, cada *n-ésimo* bloques entre B y C inclusive será intentado. El operando

D identifica al valor *n*. En términos de la numeración de la localización de los bloques, el bloque C deberá ser mayor que el bloque B, y el bloque C menos el bloque B debe ser divisible por *n* en forma par. En la parte final de este capítulo se analizará un ejemplo con el uso de este bloque.

En este bloque el número de los siguientes bloques alternos deberá ser igual a [(C-B)/D]+1, donde B y C son la localización de los bloques B y C; y D es el valor del operando. Estos bloques son el bloque B, el bloque *n*, el bloque *2n*, el bloque *3n*, ... bloque C. Si todos los bloques siguientes alternos niegan la entrada a una transacción, entonces esta se detiene por un instante, y volverá a intentar avanzar un tiempo después.

TRANSFER A,B,C

Operando	Significado	Valores por default o resultado
A	Utilizar la palabra ALL	Un error de compilación
B	Señala a un bloque, debe ser la etiqueta correspondiente a ese bloque	El bloque será el siguiente en el programa
C	Señala a un bloque, debe ser la etiqueta correspondiente a ese bloque	Un error de compilación
D	Un entero que indica la separación entre bloques que son rutas alternas	Un error de compilación

Tabla V.4.3 Resumen del bloque TRANSFER ALL

V.5 Modificación de Nivel de Prioridad: Bloque PRIORITY

Tanto el bloque condicional TRANSFER como el BOTH TRANSFER pueden ser usados para determinar hacia donde continúa una transacción. Pero también puede ser importante establecer cuándo una transacción debe avanzar con respecto a las otras que lo intentan en el mismo instante.

La prioridad inicial de una transacción está determinada en el momento de su creación y por el valor del operando E del bloque GENERATE y generalmente tiene el valor de cero. Con lo visto hasta este momento, la prioridad de una transacción permanece idéntica durante toda la simulación, sin embargo, es posible modificarla varias veces durante todo el ciclo de vida de una transacción. Para lograr esto se utiliza el bloque PRIORITY, el cual nunca niega la entrada a una transacción, cuando una transacción ejecuta este bloque, su prioridad toma el valor dado por el operando A de este bloque. Este valor se utiliza de la misma manera que el operando E del bloque GENERATE que ya se vio anteriormente en el Capítulo III, generalmente este nivel de prioridad se utiliza cuando dos transacciones tienen la misma

posibilidad de intentar avanzar a un siguiente bloque, es aquí donde el analista puede darle una prioridad a cierta transacción, mientras más grande sea este valor mayor prioridad tendrá.

PRIORITY A

Operando	Significado	Valores por default o resultado
A	El valor asignado como nivel de prioridad de las transacciones que ejecutan este bloque	Un error de compilación

Tabla V.5.1 Resumen del bloque PRIORITY

USO DE NIVELES DE PRIORIDAD PARA CONTROLAR EL ORDEN DE LA ACTUALIZACIÓN DEL MODELO

El uso de niveles de prioridad tiene los siguientes usos en GPSS/H:

1. Ayudar estructurar el orden del servicio.
2. Controlar el orden cronológico en donde las actualizaciones están aplicadas al estado de un modelo en un tiempo simulado.

V.6 EJEMPLOS

En este apartado se analizarán algunos ejemplos tomando como base el modelo visto en el Capítulo II. No se pretende dar una solución óptima en este momento, ya que sólo se busca mostrar los diferentes tipos de modelos que se pueden representar con el lenguaje GPSS/H.

EJEMPLO 1: SIMULACIÓN DE DOS SERVIDORES IDÉNTICOS

El siguiente ejemplo simula a dos recepcionistas de pasaportes (Sofía y Mari) con la mismas características, las dos reciben las solicitudes de pasaportes, toman la huella digital de los trabajadores, revisan las actas de nacimiento y las cartillas, las dos en promedio al realizar esta tarea se tardan 4±3 minutos uniformemente por trabajador. Antes de llegar a la fila de pasaportes, los trabajadores deben ser atendidos por un vigilante que es nuevo y que no les permite el acceso inmediatamente, es por eso que en promedio llegan a la fila de pasaportes entre 5±4 minutos. Se simularán cuatro horas de trabajo continuas. El sistema se ve en la *Figura V.6.1*:

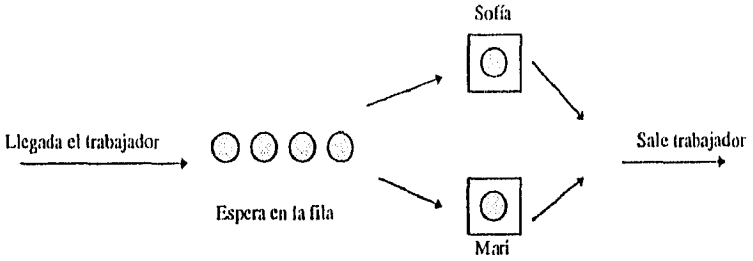


Figura V.6.1 Sistema del ejemplo uno

El problema planteado puede ser programado de la siguiente forma (*Programa V.6.1*):

```
*EJEMPLO 1 SIMULACIÓN DE DOS
*SERVIDORES IDENTICOS
          SIMULATE
RECEP     STORAGE   2
          GENERATE  5,4
          QUEUE     COLA
          ENTER     RECEP
          DEPART    COLA
          ADVANCE   4,3
          LEAVE     RECEP
          TERMINATE
          GENERATE  240
          TERMINATE 1
          START     1
          END
```

Programa V.6.1 Programa del ejemplo uno

Este ejemplo, en cuanto a su estructura es muy parecido a los ya vistos con la entidad facility, la diferencia principal es el comando STORAGE, que aparece al principio del programa, su operando con valor de dos, significa que dos recepcionistas con características parecidas estarán disponibles, y su etiqueta es RECEP. Este modelo simula dos recepcionistas con las mismas características de forma de trabajo.

Al ejecutar el modelo se obtuvieron los reportes de queue y de storage de la siguiente manera (*Tabla V.6.1*):

QUEUE	MAXIMUM CONTENTS	AVERAGE CONTENTS	TOTAL ENTRIES	ZERO ENTRIES	PERCENT ZEROS	AVERAGE TIME/UNIT	$AVERAGE TIME/UNIT	QTABLE NUMBER	CURRENT CONTENTS
COLA	1	0.002	48	47	97.9	0.008	0.388		0

Tabla V.6.1 Información de la entidad queue del ejemplo uno

En total se atendieron 48 personas, de las cuales tan solo una tuvo que esperar menos de medio minuto para ser atendida.

STORAGE	-AVG UTIL DURING- TOTAL TIME	AVAIL TIME	UNAVL TIME	ENTRIES	AVERAGE TIME/UNIT	CURRENT STATUS	PERCENT AVAIL.	CAPACITY	AVERAGE CONTENTS	CURRENT CONTENTS	MAXIMUM CONTENTS
RECEP	0.407			48	4.070	AVAIL.	100.0	2	0.814	1	2

Tabla V.6.1a Información de la entidad storage del ejemplo uno

En la *Tabla V.6.1a* se puede ver que más de la mitad del tiempo las recepcionistas estuvieron desocupadas, en promedio se tardaron en atender a cada trabajador cuatro minutos. De este ejemplo se puede suponer que, con una sola recepcionista es suficiente para llevar a cabo el trabajo, es importante recordar que la corrida de este modelo es tan solo una corrida y para llegar a alguna conclusión sería necesario hacer varias corridas con el modelo y un análisis profundo. Los trabajadores estuvieron menos tiempo en el sistema, en este ejemplo sería óptimo que el vigilante dejara pasar a los trabajadores con mayor rapidez, permitiendo así que a una mayor cantidad de trabajadores les sea permitido obtener su pasaporte.

EJEMPLO 2: SIMULACIÓN DE DOS LÍNEAS DE ESPERA EN SERIE CON DOS SERVIDORES, CADA UNA IDÉNTICAS

En este ejemplo se supondrá que los trabajadores primero tienen que pasar con una de dos recepcionistas a que les revisen todos sus papeles, como certificado médico, acta de nacimiento, fotos, entre otros, por lo que el tiempo de recepción es en promedio 6±2 minutos. Después de esta revisión, ellos deben integrarse a una segunda fila, en ella esperarán a ser atendidos por una de dos recepcionistas, éstas les tomarán sus huellas digitales y llenarán un cuestionario sobre el nivel económico de su familia. El tiempo de servicio para este proceso uniforme. En este caso el vigilante se encuentra enfermo por lo que no pudo asistir a trabajar, es por ello que los trabajadores tendrán en promedio 2±1 minutos entre llegadas. El esquema del modelo puede ser planteado de la siguiente manera (*Figura V.6.2*):

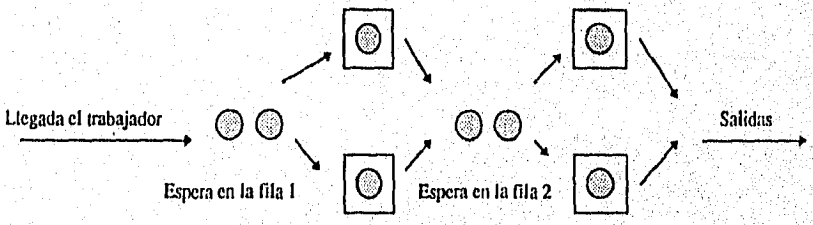


Figura V.6.2 Sistema del ejemplo dos

Se llevó a cabo una simulación de tres horas continuas, a continuación se planteará el programa generado (*Programa V.6.2*):

```
* EJEMPLO 2 DOS LÍNEAS DE ESPERA CON
*DOS SERVIDORAS
          SIMULATE
RECEP1    STORAGE    2
RECEP2    STORAGE    2
          GENERATE   2,1
          QUEUE      COLA
          ENTER      RECEP1
          DEPART     COLA
          ADVANCE    6,2
          LEAVE      RECEP1
          QUEUE      COLA1
          ENTER      RECEP2
          DEPART     COLA1
          ADVANCE    5,3
          LEAVE      RECEP2
          TERMINATE
          GENERATE   240
          TERMINATE  1
          START      1
          END
```

Programa V.6.2 Programa del ejemplo dos

Este programa es una variación del anterior, donde el trabajador espera a ser atendido por una de dos trabajadoras, después pasa a otra línea donde también espera a ser atendido. Se lleva la misma metodología del primer servicio como del segundo. En este ejemplo el trabajador permanece más tiempo en el sistema.

Los resultados obtenidos se ven reflejados en los reportes de queue (*Tabla V.6.2*) y storage (*Tabla V.6.2a*) de la siguiente manera.

QUEUE	MAXIMUM CONTENTS	AVERAGE CONTENTS	TOTAL ENTRIES	ZERO ENTRIES	PERCENT ZEROS	AVERAGE TIME/UNIT	$AVERAGE TIME/UNIT	QTABLE NUMBER	CURRENT CONTENTS
COLA	37	16.664	116	2	1.7	34.476	35.081		37
COLA1	2	0.113	77	54	70.1	0.352	1.178		0

Tabla V.6.2 Reportes de la entidad queue del ejemplo dos

En la primera fila, el número maximo de personas fue de 37, además 116 personas solicitaron el servicio y tan solo dos no tuvieron que esperar para ser atendidas. El tiempo promedio de espera fue de aproximadamente media hora y, al momento de finalizar la simulación, 37 personas esperaban a ser atendidas. En cuanto a la siguiente fila, el máximo de personas que hubo fue de dos y 54 personas no tuvieron que esperar para ser atendidas.

STORAGE	--AVG UTIL DURING-- TOTAL TIME	AVAIL TIME	UNAVL TIME	ENTRIES	AVERAGE TIME/UNIT	CURRENT STATUS	PERCENT AVAIL	CAPACITY	AVERAGE CONTENTS	CURRENT CONTENTS	MAXIMUM CONTENTS
RECEP1	0.989			79	6.012	AVAIL	100.0	2	1.979	2	2
RECEP2	0.769			77	4.795	AVAIL	100.0	2	1.538	2	2

Tabla V.6.2a Reportes de la entidad storage del ejemplo dos

Las recepcionistas de la primera fila casi todo el tiempo estuvieron ocupadas atendiendo a trabajadores, en total se atendió a 79 personas, y las recepcionistas se tardaron con cada trabajador en promedio un poco más de seis minutos, casi siempre las dos recepcionistas estuvieron ocupadas. En cuanto a la segunda fila, las recepcionistas estuvieron atareadas más del 75% del tiempo y atendieron a 77 personas, en promedio tardaron con cada trabajador algo más de cuatro minutos y medio.

EJEMPLO 3: SIMULACIÓN CON EL USO TRANSFER CONDICIONAL

Un día antes de llevar sus papeles para la tramitación del pasaporte, a los trabajadores se les hace un estudio médico, ahí se encuentra una enfermera que después de hacerles el estudio, les comunica cual es el siguiente trámite. Como es demasiado comunicativa, les comenta que Sofía es más eficiente y más amable que Mari. Con esta sugerencia, un 65% prefieren a Sofía. El vigilante se reportó enfermo, por lo cual no pudo asistir a trabajar, entonces los trabajadores llegan en promedio 2±1 minutos a las filas. Los trabajadores se presentan a la oficina y se encuentran con dos filas, una para cada una de las recepcionistas. Sofía y Mari reciben el acta de nacimiento, la cartilla liberada, las fotos y toman las huellas, además llenan un comprobante de pago. Mari se tarda más tiempo en recibir los papeles que Sofía, pues apenas tiene medio año trabajando en esta oficina, es por eso que su tiempo de servicio es de 6±2 minutos. Sofía que ya tiene trabajando dos años, en promedio se tarda por cada trabajador 5±3 minutos. El esquema del sistema planteado puede quedar formulado de la siguiente manera (*Figura V.6.3*):

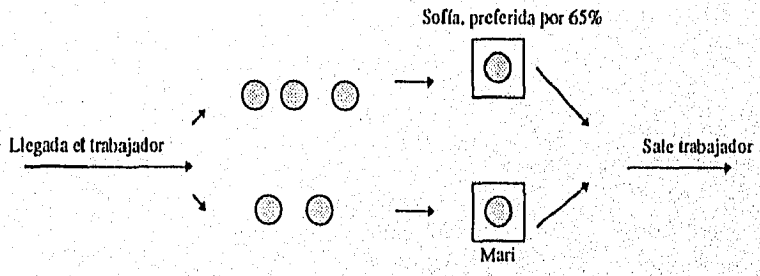


Figura V.6.3 Esquema del ejemplo tres

Se simulará cuatro horas continuas de trabajo, el programa planteado queda estructurado de la siguiente forma *Programa V.6.3*.

```
*USO TRANSFER CONDICIONAL
          SIMULATE
          GENERATE      2,1
          TRANSFER      .650,FILA1
          QUEUE         COLA
          SEIZE         MARI
          DEPART        COLA
          ADVANCE       6,2
          RELEASE       MARI
          TERMINATE
FILA1     QUEUE         COLA1
          SEIZE         SOFIA
          DEPART        COLA1
          ADVANCE       5,3
          RELEASE       SOFIA
          TERMINATE
          GENERATE      240
          TERMINATE     1
          START         1
          END
```

Programa V.6.3 Programa del ejemplo tres

En este ejemplo existen dos opciones, una es ir a hacer el servicio con Mari y otra con Sofía. Se utiliza el bloque TRANSFER .650 ,FILA1, para ponderar la decisión del trabajador con respecto a que fila formarse, tomando en cuenta una preferencia por SOFIA. FILA1 es la etiqueta de la segunda fila. En este modelo los trabajadores pueden seguir uno de dos caminos, estos caminos están identificados como un conjunto de bloques, que terminan con la instrucción TERMINATE, con ella los trabajadores salen del sistema.

Al ejecutar el programa se obtiene le siguiente información sobre la simulación.

QUEUE	MAXIMUM CONTENTS	AVERAGE CONTENTS	TOTAL ENTRIES	ZERO ENTRIES	PERCENT ZEROS	AVERAGE TIME/UNIT	$AVERAGE TIME/UNIT	QTABLE NUMBER	CURRENT CONTENTS
COLA	4	1.476	39	5	12.8	9.085	10.421		1
COLA1	35	17.697	80	1	1.3	53.091	53.763		35

Tabla V.6.3 Reporte de las entidad de queue del ejemplo tres

Como se puede analizar en la *Tabla V.6.3*, la fila de Mari, la no preferida, fue de cuatro personas como máximo, hubo 39 personas que trataron buscar el servicio. Cinco personas no tuvieron que esperar a ser atendidas. En general tuvieron que esperar en promedio más de nueve minutos. Al momento de finalizar la simulación, una persona se encontraba en espera para ser atendida. En cuanto a la fila de Sofía, 35 personas como máximo esperaron en la fila, 80 personas esperaron a ser atendidas y tan solo el primero no tuvo que esperar a ser atendido. En general, las personas tuvieron que aguardar casi una hora para ser atendidos. En promedio, en la fila hubo casi 18 personas esperando. Al momento de finalizar la simulación, 35 personas esperaban a ser atendidas, si esto ocurriera en la realidad el costo de espera sería muy alto y quizá muchos trabajadores que desean ir al Canadá se no se animarían a ir. Se analizará el reporte de la entidad facility.

	--AVG UTIL DURING--							
FACILITY	TOTAL TIME	AVAIL. TIME	UNAVL TIME	ENTRIES	AVERAGE TIME/XACT	CURRENT STATUS	PERCENT AVAIL	SEIZING XACT	PREEMPTING XACT
MARI	0.936			38	5.912	AVAIL.		113	
SOFIA	0.972			45	5.185	AVAIL.		69	

Tabla V.6.3a Reportes de las entidades facilities para el ejemplo tres

Como se puede ver en la *Tabla V.6.3a* la mayor parte la de simulación, las recepcionistas estuvieron trabajando, la primera recepcionista atendió a 38 personas con un tiempo promedio de casi 6 minutos y la segunda recepcionista recibió a 45 personas en un promedio de 5 minutos.

EJEMPLO 4: SIMULACIÓN CON EL USO DE TRANSFER BOTH

En esta simulación, los trabajadores llegan a una sola fila, donde pueden ser atendidos por Sofía o por Mari. Ahora sólo tienen que recibir los papeles sin llenar el formato de pago. El tiempo de servicio de Sofía es de 4±2 minutos en promedio y el de Mari es de 5±3 minutos en promedio. Como el vigilante se reportó enfermo otra vez, se decidió poner a otro vigilante, pero como ya tiene experiencia, los deja entrar al edificio en promedio 4±2 minutos entre trabajadores. La recepcionista que se libere primero será con la que el trabajador intente recibir el servicio. Un esquema del sistema es el siguiente (*Figura V.6.4*):

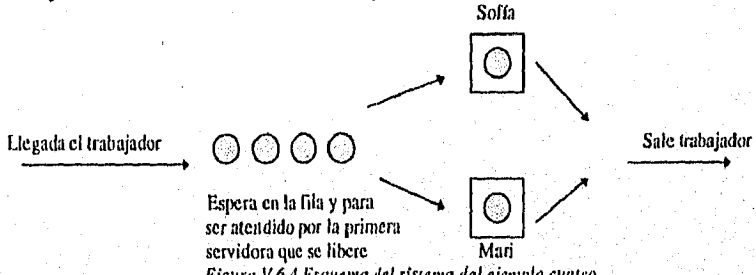


Figura V.6.4 Esquema del sistema del ejemplo cuatro

A continuación se muestra el programa planteado (*Programa V.6.4*), se simularán cuatro horas continuas de trabajo.

```
*EJEMPLO 4 USO DE TRANSFER CON BOTH
          SIMULATE
          GENERATE   4,2
          QUEUE      COLA
          TRANSFER   BOTH,FILA1,FILA2
FILA1      SEIZE      SOFIA
          DEPART     COLA
          ADVANCE    4,2
          RELEASE    SOFIA
          TERMINATE
FILA2      SEIZE      MARI
          DEPART     COLA
          ADVANCE    5,3
          RELEASE    MARI
          TERMINATE
          GENERATE   240
          TERMINATE  1
          START      1
          END
```

Programa V.6.4 Programa del ejemplo cuatro

Este ejemplo es muy parecido al primer ejemplo de este capítulo, la diferencia mayor es que en este caso se está simulando a dos recepcionistas diferentes, con diferentes tiempos de servicios, para llevar a cabo esto se utilizó la palabra BOTH en el bloque TRANSFER. Como es tan sólo una sola fila se usa un solo bloque QUEUE y como son dos opciones de recepción se utilizan dos bloques DEPART.

Al llevar a cabo la simulación se obtienen los resultados que se muestran a continuación:

QUEUE	MAXIMUM CONTENTS	AVERAGE CONTENTS	TOTAL ENTRIES	ZERO ENTRIES	PERCENT ZEROS	AVERAGE TIME/UNIT	$AVERAGE TIME/UNIT	QTABLE NUMBER	CURRENT CONTENTS
COLA	1	0.003	58	57	98.3	0.012	0.713		0

Tabla V.6.4 Información del cuarto ejemplo del reporte queue

En la *Tabla V.6.4* se puede observar que solo una persona tuvo que esperar para ser atendida, 58 trabajadores pidieron el servicio y casi el 99% de las personas no tuvieron que esperar para ser atendidas.

	--AVG-UTIL-DURING--								
FACILITY	TOTAL TIME	AVAIL TIME	UNAVL TIME	ENTRIES	AVERAGE TIME/XACT	CURRENT STATUS	PERCENT AVAIL	SEIZING XACT	PREEMPTING XACT
SOFIA	0.649			38	4.098	AVAIL		59	
MARI	0.434			20	5.211	AVAIL			

Tabla V.6.4a Muestra la información de la entidad facility

En la *Tabla V.6.4a* se puede ver que Sofía trabajó casi un 65% del tiempo total atendiendo así a casi 40 personas y Mari estuvo desocupada un 57% del tiempo total, y atendió a 20 personas. De este ejemplo se puede analizar que Sofía está trabajando más que Mari, y que se le tiene que exigir más a Mari para que logre atender a más personas o pagarle más a Sofía por su eficiencia.

EJEMPLO 5: SIMULACIÓN DONDE EXISTE LA OPCIÓN DE NO ESPERAR A SER ATENDIDO (DESERCIÓN)

En este ejemplo Sofía atenderá a los trabajadores, recibiendo el acta de nacimiento, la cartilla liberada, las fotos y les tomará las huellas. Para llevar a cabo esto se tardará en promedio por trabajador 5±3 minutos. Ahora, para que los trabajadores no tengan que esperar parados, se han instalado dos sillas y como el lugar es muy pequeño, sólo se permite que dos personas permanezcan en la sala. El vigilante les está permitiendo la entrada en un promedio de 4±3 minutos entre trabajador. El esquema del sistema es el siguiente (*Figura V.6.5*):

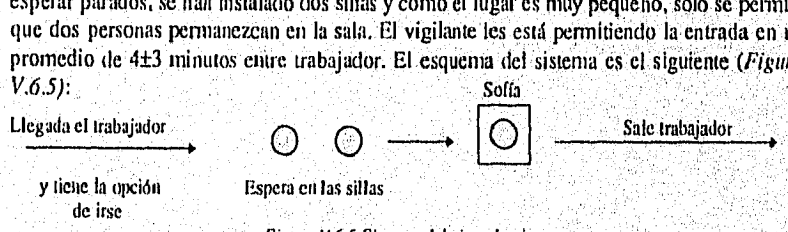


Figura V.6.5 Sistema del ejemplo cinco

En este ejemplo se simulará la llegada de 35 trabajadores que fueron atendidos. El programa queda planteado de la siguiente manera (*Programa V.6.5*):

```
*EJEMPLO 5 EXISTE LA OPCIÓN DE NO
*ESPERAR A SER ATENDIDO
          SIMULATE
LUGAR     STORAGE    2
          GENERATE   4,3
          TRANSFER   BOTH,,IRSE
          ENTER      LUGAR
          SEIZE      SOFIA
          LEAVE      LUGAR
          ADVANCE    5,3
          RELEASE    SOFIA
          TERMINATE  1
IRSE      TERMINATE
          START      35
          END
```

Programa V.6.5 Programa en GPSSIII del ejemplo cinco

En vez de utilizar la entidad queue, en este ejemplo se utilizó la entidad storage, en ella, se simula la espera de dos trabajadores en las sillas. Se usa el bloque TRANSFER para determinar si el trabajador espera a ser atendido o se va y regresa al día siguiente. En el caso que la persona decida no esperar, el bloque donde se encuentra la etiqueta IRSE, la persona sale del sistema.

Los resultados obtenidos son los siguientes, primero se analizará el reporte de la información sobre SOFIA, luego la información sobre las sillas, seguido de la información sobre los bloques y finalmente los tiempos marcados por los relojes.

	--AVG-UTIL-DURING--									
FACILITY	TOTAL TIME	AVAIL TIME	UNAVL TIME	ENTRIES	AVERAGE TIME/XACT	CURRENT STATUS	PERCENT AVAIL	SEIZING XACT	PREEMPTING XACT	
SOFIA	0.969			35	4.972	AVAIL				

Tabla V.6.5 Información de la entidad facility del ejemplo cinco

Como se puede ver en la *Tabla V.6.5*, Sofía estuvo bastante ocupada, casi el 97% del tiempo total, atendió a 35 personas y en promedio con cada trabajador se tardo casi cinco minutos de servicio.

	--AVG-UTIL-DURING--										
STORAGE	TOTAL TIME	AVAIL TIME	UNAVL TIME	ENTRIES	AVERAGE TIME/UNIT	CURRENT STATUS	PERCENT AVAIL	CAPACITY	AVERAGE CONTENTS	CURRENT CONTENTS	MAXIMUM CONTENTS
LUGAR	0.541			37	5.249	AVAIL	100.0	2	1.082	2	2

Tabla V.6.5a Información de la entidad storage del quinto ejemplo

Las sillas estuvieron ocupadas más de la mitad del tiempo (*Tabla V.6.5a*), 37 personas intentaron sentarse y en promedio más de cinco minutos tuvieron que esperar para ser atendidos.

BLOCK	CURRENT	TOTAL
1		44
2		44
3	2	37
4		35
5		35
6		35
7		35
8		35
IRSE		7

Tabla V.6.5b Información sobre los bloques del programa cinco

De la *Tabla V.6.5b* se puede obtener el número de personas que decidieron irse sin esperar a ser atendidos, en este caso fue de 7.

RELATIVE CLOCK: 179.5311	ABSOLUTE CLOCK: 179.5311

Tabla V.6.5c Información sobre los relojes

Para atender a 35 trabajadores, se necesitaron casi tres horas (*Tabla V.6.5c*).

EJEMPLO 6: SIMULACIÓN CON EL USO ALL DEL BLOQUE TRANSFER

En este ejemplo se supondrá que hay tres recepcionistas de pasaportes (Sofía, Marí y Concha), recibiendo la solicitud, el acta de nacimiento, la cartilla liberada, etc. Sofía, como ya lleva bastante tiempo haciendo este tipo de trámites, se tarda 5±3 minutos en promedio por trabajador, Mari se tarda 5±4 y Concha por ser la que apenas está aprendiendo a recibir ese trámite se tarda 6±2 minutos por persona. En esta ocasión el vigilante no pudo asistir a trabajar y las personas tienen el acceso inmediato, llegan en forma uniforme con un promedio 2±1 minutos entre personas a la fila a una sola cola y esperan hasta que alguna de las recepcionistas los puedan atender. El esquema del sistema puede ser el siguiente (*Figura V.6.6*):

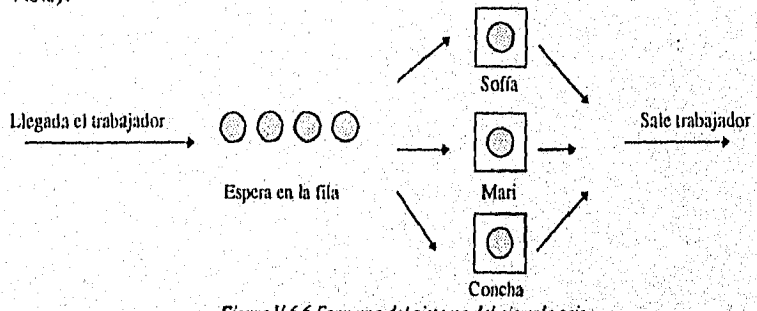


Figura V.6.6 Esquema del sistema del ejemplo seis

Se simularán a 100 trabajadores, el programa puede ser codificado de la siguiente manera (*Programa V.6.6*):

```
*EJEMPLO 6 USO ALL DEL TRANSFER
           SIMULATE
*
           GENERATE    2,1
           QUEUE       COLA
           TRANSFER    ALL,TRA1,TRA3,5
*********************************************
TRA1       SEIZE       MARI
           DEPART      COLA
           ADVANCE     5,4
           RELEASE     MARI
           TERMINATE   1
*********************************************
           SEIZE       CONCHA
           DEPART      COLA
           ADVANCE     6,2
           RELEASE     CONCHA
           TERMINATE   1
*********************************************
TRA3       SEIZE       SOFIA
           DEPART      COLA
           ADVANCE     5,3
           RELEASE     SOFIA
           TERMINATE   1
*********************************************
           START       100
           END
```

Programa V.6.6 Programa del ejemplo seis

Este modelo es muy parecido al ejemplo cuatro de esta sección, la única diferencia es que son tres recepcionistas y no dos, tanto Sofía como Mari y Concha, tienen diferente comportamiento y es la diferencia entre al usar la entidad storage y este tipo de modelo.

Los resultados son los siguientes:

	--AVG-UTIL-DURING--								
FACILITY	TOTAL TIME	AVAIL TIME	UNAVL TIME	ENTRIES	AVERAGE TIME/XACT	CURRENT STATUS	PERCENT AVAIL	SEIZING XACT	PREEMPTING XACT
MARI	0.917			40	4.690	AVAIL		101	
CONCHA	0.884			30	6.031	AVAIL		102	
SOFIA	0.819			32	5.237	AVAIL			

Tabla V.6.6 Información de las recepcionistas del ejemplo seis

Como se muestra en la *Tabla V.6.6*, las tres recepcionistas trabajaron más del 80% del tiempo, Mari atendió a 40 personas, Concha recibió a 30 trabajadores y 32 personas fueron atendidas por Soffa. Mari se tardó en promedio 4.6 minutos por trabajador, Sofía 5.23 minutos y Concha 6 minutos. Aunque Sofía sea la que tarda menos en atender a los trabajadores, Mari atendió a más trabajadores por el tipo de bloque TRANSFER utilizado en

el problema, dado que primero intenta ser atendido por Mari o Concha y al final cuando ninguna de las dos puede es entonces cuando intenta ser atendido por Sofía.

RELATIVE CLOCK: 204.6455 ABSOLUTE CLOCK: 204.6455

Tabla V.6.6a Relojes al terminar la simulación del ejemplo seis

Para atender a 100 trabajadores se requirieron más de tres horas de trabajo continuas (*Tabla V.6.6a).*

QUEUE	MAXIMUM CONTENTS	AVERAGE CONTENTS	TOTAL ENTRIES	ZERO ENTRIES	PERCENT ZEROS	AVERAGE TIME/UNIT	$AVERAGE TIME/UNIT	QTABLE NUMBER	CURRENT CONTENTS
COLA	3	0.248	104	60	57.7	0.488	1.153		2

Tabla V.6.6b Información de la entidad queue del ejemplo seis

En la *Tabla V.6.6b*, el máximo de personas en la fila fue de tres, 60 no tuvieron que esperar, es decir, más de la mitad y a los que tuvieron que esperar, les tomo más de un minuto en recibir el servicio.

CAPÍTULO VI
DISEÑO DE EXPERIMENTOS PARA COMPARAR ALTERNATIVAS

VI.1 VISUALIZACIÓN DE UNA MUESTRA SIMULADA

En la *Figura VI.1.1*, se ilustra el proceso de obtener una muestra de una población que consiste de valores provenientes de una variable aleatoria continua. Como ya se mencionó, una variable continua es una variable que puede tomar valores de un intervalo, estos pueden ser los tiempos entre llegadas, los tiempos de servicio, etc. A la izquierda se ilustra la función de distribución de probabilidad para una población uniforme continua distribuida en un intervalo de 0 a 1.

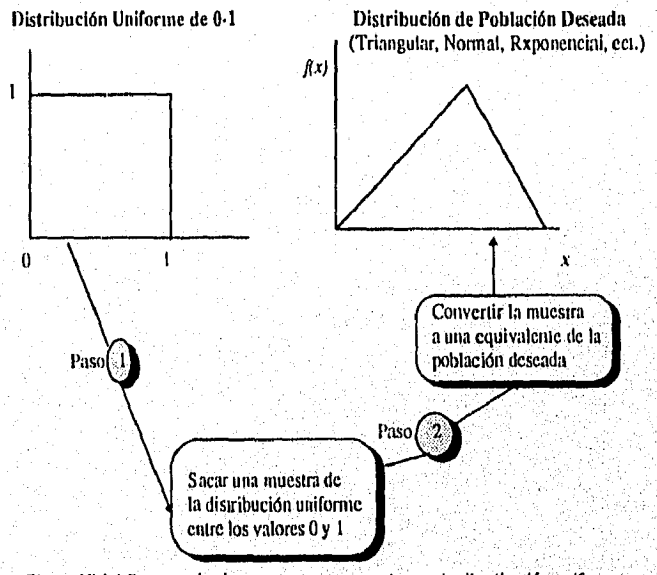


Figura VI.1.1 Proceso de obtener una muestra en base a la distribución uniforme

A la derecha de la *Figura VI.1.1* se obtiene una muestra con base en la función de densidad de probabilidad *f(x)* deseada. Este proceso se lleva a cabo de la siguiente forma:

1. Sacar un valor de una población uniforme entre 0 y 1.

2. Convertir este valor a un valor equivalente para la población deseada. Para transformar estos valores se usan los siguientes métodos:

- Método de la transformación inversa
- Método de rechazo
- Método de composición
- Método de simulación directa

Para poder lograr el proceso es necesario tener una fuente de números aleatorios uniformemente distribuidos en un intervalo de 0 a 1.

GENERADORES DE VALORES ALEATORIOS UNIFORMES DEL GPSS/H

El lenguaje GPSS/H contiene una fuente casi ilimitada de números aleatorios uniformes, en un intervalo de 0 a 1. A esta fuente usualmente se le llama: generadora de números aleatorios o RNGs (Random Number Generators). Estos generadores obtienen muestras aleatorias uniformemente distribuidas en el intervalo abierto de 0 a 1. Cada uno de estos generadores no tiene un punto de comienzo idéntico por default y utiliza un procedimiento matemático para producir valores aleatorios uniformes llamado Lehmer. Este algoritmo es determinístico, es decir, que los valores producidos no son exactamente aleatorios, son pseudoaleatorios (reproducibles). Existen más de dos billones de valores en el intervalo abierto 0-1 de donde GPSS/H puede generar valores. Para identificar a los generadores en el lenguaje GPSS/H se utilizan números, es decir RN1, RN2, etc.

VI.2 MUESTRAS DIFERENTES A LA UNIFORME EN EL GPSS/H

En el lenguaje GPSS/H existe la opción de simular no sólo valores uniformes, sino diferentes distribuciones: la distribución exponencial, erlang, normal y triangular, además de que también puede generar muestras para distribuciones empíricas. A continuación se tratará el tema de las diferente opciones, bien valdría la pena agregar que los ejemplos vistos hasta aquí pueden ser simulados utilizando alguna de las otras distribuciones.

MUESTRAS DE UNA DISTRIBUCIÓN EXPONENCIAL: RVEXPO

GPSS/H contiene generadores de funciones de distribución como la exponencial, normal y la triangular. La variable aleatoria exponencial, que es continua, puede tomar valores mayores a cero y está muy relacionada con la distribución de Poisson. La *Figura VI.2.1* muestra su función de densidad para ciertos valores. Generalmente esta variable es utilizada para modelar tiempos entre llegadas sucesivas cuando no hay control sobre ellas, corresponde a la probabilidad de no ocurrencia en el intervalo de longitud t. En el lenguaje GPSS/H se utiliza la función RVEXPO (Random Variate EXPOnential) para generar valores de este tipo, tomando la siguiente forma:

RVEXPO (j, λ)

Donde la *j* es el número del generador de valores uniformes y λ (ocurrencias del evento por unidad de tiempo) puede ser especificada con o sin punto decimal. Si se supone, a partir del ejemplo visto a lo largo del trabajo, que la media del tiempo entre llegadas es 18.5, distribuida exponencialmente, y que se desea utilizar el generador 4, por citar alguno, entonces se usaría el siguiente bloque para introducir trabajadores al modelo:

GENERATE RVEXPO (4,18.5)

RVEXPO (4,18.5) fue especificada como el operando A y el operando B tuvo el valor por default 0. Cuando el bloque GENERATE es ejecutado, la función RVEXPO puede ser evaluada para determinar el valor actual del operando A. Esta distribución contiene un alto grado de variabilidad, como se puede ver en la *Figura VI.2.1*, su varianza es la media al cuadrado. Para analizar si las observaciones de tiempos entre llegadas siguen esta distribución, existen métodos que verifican si efectivamente los datos corresponden a una variable exponencial, algunos de éstos se pueden encontrar en el paquete *Statgraphics*. La información de esta distribución se encuentra en la *Tabla VI.2.1*. La distribución exponencial en el operando del bloque ADVANCE, se puede usar de la siguiente manera:

ADVANCE RVEXPO (4,18.5)

Función de Densidad	$f(x) = \lambda e^{-\lambda x}$
Parámetro	λ : número de ocurrencias por unidad de tiempo
Rango	$x \in [0,\infty)$
Media	$\lambda : \alpha = 1/\lambda$ donde α es el tiempo promedio entre ocurrencias
Varianza	$1/\lambda^2$

Tabla VI.2.1 Información de la distribución exponencial

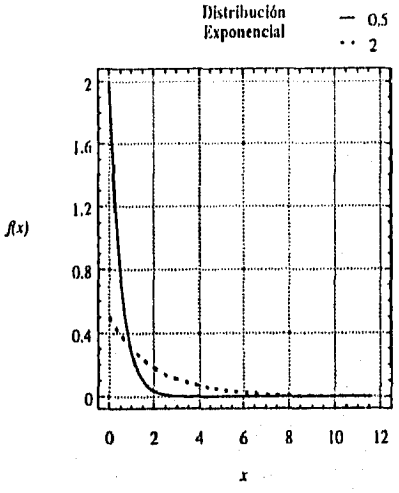


Figura VI.2.1 Gráficas de la distribución exponencial, con $\lambda=0.5$ *y* $\lambda=2$

MUESTRAS DE LA DISTRIBUCIÓN ERLANG

Para producir una muestra de la distribución Erlang se requiere de dos o más muestras de una distribución exponencial. Una variable aleatoria exponencial es un caso especial de la variable aleatoria Erlang. Si el número de eventos aleatorios independientes ocurren en un lapso específico de una variable de Poisson con una frecuencia constante de ocurrencia de λ, entonces el tiempo de espera hasta que ocurre el *k-ésimo* evento de Poisson sigue una distribución Erlang. Esta distribución se caracteriza por dos parámetros, conocidos como el orden y escala, donde el parámetro orden puede tomar valores desde uno a mayores cantidades. La distribución exponencial igualmente es un caso especial de la distribución Erlang donde la gráfica tiene valor de 1 *(Figura VI.2.2)*. Una forma para sacar una muestra de una distribución Erlang de orden k con un valor λ es el formar la suma de k muestras de una distribución exponencial con un valor esperado de λ/k, si se supone que se desea una muestra de la distribución Erlang de orden 2 con un valor esperado 0.9, se puede simular esta distribución utilizando la secuencia de los siguientes bloques:

```
ADVANCE RVEXPO (3,0.45)
ADVANCE RVEXPO (3,0.45)
```

En el GPSS/H las expresiones aritméticas pueden ser utilizadas en casi todos los contextos, incluyendo los operandos. Es por eso que el ejemplo visto anteriormente se puede expresar como un solo bloque, donde el operando A puede ser expresado como la suma de dos RVEXPO, se muestra a continuación:

```
ADVANCE RVEXPO (3,0.45) +RVEXPO (3,0.45)
```

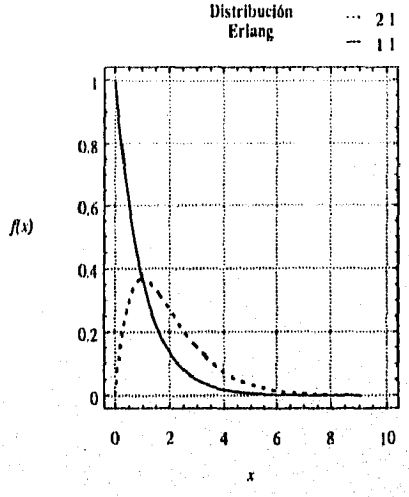


Figura VI.2.2 Distribución Erlang de orden 2 y 1, y escala 1 y 1 respectivamente

Se pueden usar expresiones aritméticas en GPSS/H: la suma, la resta, la división convencional, la división modular, la raíz cuadrada y funciones trigonométricas. Los valores obtenidos de estas expresiones pueden ser almacenadas en ampervariables con el comando de control LET o con el comando BLET, también pueden ser almacenadas como elementos de una lista de los parámetros de las transacciones con el bloque ASSIGN y con los bloques SAVEVALUE y MSAVEVALUE.

El GPSS/H permite el uso de expresiones booleanas, es decir tener valores verdaderos o falsos, formadas al utilizar operadores lógicos para unir referencias a valores ciertos o falsos.

MUESTRA DE UNA DISTRIBUCIÓN NORMAL: RVNORM

Las variables aleatorias distribuidas normalmente son continuas y simétricas con respecto a su media, tomando valores entre - ∞ a + ∞. La variable aleatoria toma con mayor frecuencia valores *cercanos* (±2σ) a la media y con muy poca continuidad valores lejanos. La distribución es simétrica con respecto a la media y generalmente ocurre en muestras *grandes (mayores a 50)*. La distribución normal puede ser caracterizada completamente dando su media (μ) y desviación estándar (σ) o varianza (σ^2), cierta información de la distribución puede ser analizada en la *Tabla VI.2.2*. En la *Figura VI.2.3*, se muestran algunas curvas de la distribución normal. GPSS/H contiene un función para sacar la muestra de una distribución normal, esta es RVNORM (Random Variate NORMal), tomando la siguiente forma:

```
RVNORM (j, μ,σ)
```

Donde la *j* indica el número del generador a usar; μ, la media y σ, la desviación estándar pueden ser expresadas sin punto decimal. En el GPSS/H, esta función regresa valores ±43 desviaciones standars de la media.

Si se considera el caso para tiempos entre llegadas que se modela de una distribución normal, estos tiempos no pueden ser negativos, así que se debe utilizar a la función con su debida precaución, para evitar que la muestra sea negativa se puede usar la función ABS (ABSolute), de la siguiente manera:

```
ABS (xpr)
```

Donde *xpr* es la forma general para una expresión aritmética. ABS convierte el resultado de una expresión aritmética a valor positivo. Si se supone ahora que los tiempos entre llegadas se comportan en forma normal, y que se desea simular esto, entonces se debe utilizar un bloque GENERATE, de la siguiente manera:

```
GENERATE ABS(RVNORM (5,5.5,1.0))
```

El bloque ADVANCE puede ser expresado de la siguiente manera:

```
ADVANCE ABS(RVNORM (6,250,45))
```

Función de Densidad	$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$
Parámetros	$\mu \in R$ $\sigma^2 > 0$
Rango	$x \in R$
Media	μ
Varianza	σ^2

Tabla VI.2.2 Información de la distribución normal

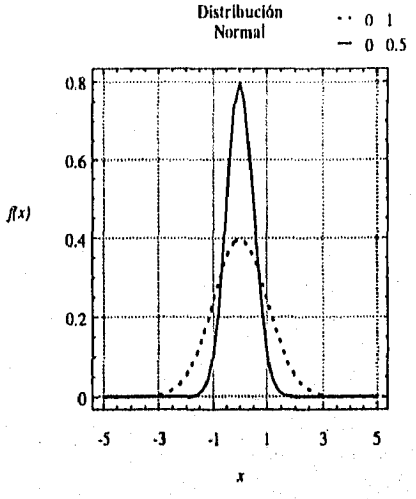


Figura VI.2.3 Distribución Normal

MUESTRA DE UNA DISTRIBUCIÓN TRIANGULAR: RVTRI

Las variables aleatorias triangulares son continuas, caracterizadas por sus valores: *o*ptimista, probab*l*e y *p*esimista, puede representar el tiempo de duración de una actividad. La *Figura VI.2.4* muestra algunas distribuciones de probabilidad triangulares, puede ser asimétrica hacia la izquierda o la derecha, existen otras distribuciones con esta característica como la gama y la lognormal, cierta información de esta distribución puede ser vista en la *Tabla VI.2.3*. En GPSS/H existe una función que genera valores que se distribuyen triangularmente, RVTRI, se escribe de la siguiente manera:

```
RVTRI(j, pesimista, posible, optimista)
```

Donde la *j* indica el número del generador de números aleatorios y los valores de pesimista, posible y optimista *(o, L, P)* pueden llevar punto decimal o no. Si se supone que se desea generar transacciones, entonces se utiliza a la función el bloque GENERATE de la siguiente manera:

```
GENERATE RVTRI (1,15,30,60)
```

Función de Densidad	$f(x) = \frac{2(x-o)}{(L-o)(P-o)}$ $o < x < L$ $= \frac{2(P-x)}{(P-L)(P-o)}$ $L < x < P$
Parámetro	o,L,P con o<L<P
Rango	x ε [o,P]
Media	(o+L+P)/3
Varianza	$(o^2+L^2+P^2-oL-oP-PL)/18$

Tabla VI.2.3 Información de la distribución Triangular

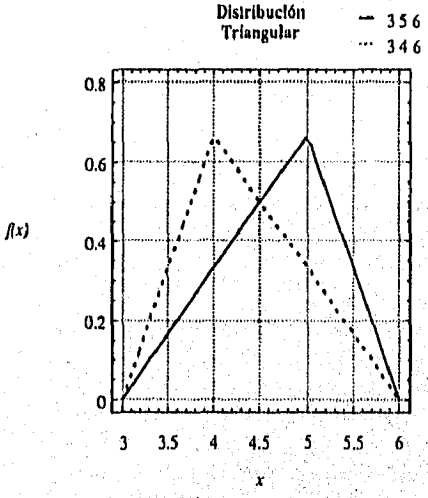


Figura VI.2.4 Distribución triangular

MUESTRA DE DISTRIBUCIONES DISCRETAS: FUNCIONES D

El GPSS/H no contiene funciones hechas para muestras de distribuciones empíricas, sin embargo incluye las herramientas necesarias para definir funciones discretas, así como de cualquier distribución empírica. Para definir estas funciones se debe especificar un generador de números aleatorios para ser usado como punto de partida en la obtención de la muestra deseada. Para definir a las funciones también se requiere indicar tanto los valores que ocurren en cierta población discreta (datos reales) como sus frecuencias de ocurrencia, no se necesitan las frecuencias relativas sino las frecuencias acumuladas.

El analista, para definir funciones, utiliza un comando de control llamado FUNCTION, seguido de uno o más comandos, para crear una función siempre se debe contemplar lo siguiente:

1. Una etiqueta para identificar a la función, se hace de la misma manera que se etiqueta a entidades facilities, queues, storages y a cualquier bloque.
2. En el campo de operación la palabra FUNCTION.
3. El operando A, dado por el analista, siendo éste el argumento. Una función mapea el valor de una variable (argumento) hacia el valor de otra variable (valor producido por la función). Si una función esta definida para obtener una muestra de una distribución empírica, entonces su argumento será un generador de valores uniformes entre 0 y 1, por ejemplo, RN5 o RN1.
4. El primer carácter del operando B es una letra que indica el tipo de función a ser definida, por ejemplo la letra D indica que es una función discreta. Existen otros cinco tipos de funciones, las funciones de lista (L), es un caso especial de las funciones discretas, las funciones que identifican un símbolo (S), las funciones que tienen atributos numéricos estandares (E) y toman forma de una expresión aritmética y las funciones (M), las cuales son un caso especial de las funciones E.
5. El resto del operando B es un entero que indica el *número* de parejas ordenadas de puntos dados para la función.

La función se define de la siguiente manera:

```
Etiqueta FUNCTION argumento, tipo_número
x1,y1/x2,y2/x3,y3/.....lxnúmero,ynúmero
```

La siguiente línea después del comando FUNCTION consta de parejas ordenadas de puntos que definen a la función, puede existir más de una línea de este tipo. El primer miembro de la pareja ordenada (*x1,x2,x3, etc.*) es el valor del argumento de la función y el segundo miembro es el valor que produce la función. Si una función se utiliza para definir una distribución empírica, el primer miembro es la frecuencia acumulada, y el segundo es el valor de la variable aleatoria. Esta línea comienza desde la primera columna, un slash (/) separa a las parejas ordenadas, y no se permite el uso de espacios, pues después del primer espacio el resto se toma como comentario. Si se supone el ejemplo de la *Tabla VI.2.4*:

	Tiempo	Frecuencia Relativa	Frecuencia Acumulada
1	4.0	0.20	0.20
2	7.5	0.35	0.55
3	10.5	0.15	0.70
4	13.5	0.10	0.80
5	16.5	0.20	1

Tabla VI.2.4 Ejemplo para definir una distribución empírica

Su función se definirfa de la siguiente manera:

```
EMPI FUNCTION RN6, D5
.2,4/.55,7.5/.7,20.5/.8,13.5/1,16.5
```

Hay que notar que en el argumento se escribió un RN6, que especifica que generador se utiliza, en este caso el número seis. Los pares ordenados no pueden exceder de la columna 72, pero se puede utilizar más de una línea para estos. Una función debe ser tecleada antes del comando de control START, puede ser escrita entre el comando SIMULATE y los bloques.

UTILIZACIÓN DE LAS FUNCIONES DISCRETAS: BLOQUES GENERATE Y ADVANCE

El patrón usado para referirse a las funciones toma la siguiente forma:

```
FN(Identificador)
```

Por ejemplo, para llamar a la función EMPI, se utilizaría lo siguiente:

```
FN(EMPI)
```

Estos identificadores también pueden ser números enteros positivos. Si se supone que la función EMPI describe una distribución de retención de tiempo, entonces el bloque ADVANCE sería utilizado de la siguiente manera:

```
ADVANCE FN(EMPI)
```

Este mismo procedimiento se utiliza para un bloque GENERATE.

USO DE FUNCIONES PARA DETERMINAR EL SIGUIENTE BLOQUE DE UNA TRANSACCIÓN EN FORMA ALEATORIA

Anteriormente se analizaron algunos usos del bloque TRANSFER, aquí se introduce el método de hacer que las transacciones seleccionen su siguiente bloque en forma aleatoria de un conjunto de dos o más bloques posibles, para ello se utiliza el TRANSFER incondicional. El operando B señala la localización ocupada por el siguiente bloque, este señalamiento se lleva a cabo al etiquetar el bloque siguiente y utilizando esa etiqueta como el operando B en TRANSFER. El operando B puede ser una función con el valor producido por la función, siendo la etiquetación del bloque, la transacción en un TRANSFER incondicional entonces tomará al bloque etiquetado como su siguiente bloque. Esta función es como un bloque TRANFER condicional con más de dos opciones. Un ejemplo se muestra en la *Tabla VI.2.5*:

```
TRO FUNCTION RN1, D3
.2,CAM1/.8,CAM2/1,CAM3
....
      TRANSFER ,FN(TRO)
.....
```

Tabla VI.2.5 Ejemplo del uso de una función en un bloque TRANSFER

En el ejemplo anterior, una función discreta llamada TRO tiene un generador de números aleatorios como su argumento, los utilizados para definir a la función no son números sino etiquetas de bloques. Esta función produce una etiqueta de un bloque aleatoriamente. Supongamos que se ejecuta lo anterior, y que la función da como resultado CAM2, entonces la transacción en el bloque TRANSFER tomaría como su siguiente bloque al etiquetado con CAM2.

MUESTRAS DE DISTRIBUCIONES CONTINUAS: FUNCIONES C

La única diferencia para definir las distribuciones discretas y las continuas es que en vez de utilizar la letra D para discretas, se utiliza la letra C en el primer carácter en el operando B del comando de control FUNCTION en el *tipo*.

En la *Tabla VI.2.3* muestra un ejemplo de como hacer una función continua:

```
BANC FUNCTION RN7, C6
0,60/.07,120/.23,180/.73,240
.92,300/1,360
```

Tabla VI.2.3 Uso de una función para definir una distribución continua

MUESTRA DE DISTRIBUCIONES UNIFORMES CONTINUAS CON LA FUNCIONES C

Es posible obtener una muestra de distribuciones uniformes utilizando funciones continuas, que están definidas con dos pares de puntos; si se supone que en un bloque GENERATE se desea simular tiempos entre llegadas uniformemente y continuamente distribuidas en el intervalo abierto 360±240, y se desea usar el generador de números aleatorios 3. Esto puede ser obtenido al utilizar la función de dos puntos continuos de la siguiente manera:

```
TYPE2IAT FUNCTION RN3, C2
0,120/1,600
```

La función llamada TYPE2IAT tiene por argumento RN3, los dos puntos de los pares ordenados para definir la función x_1 y x_2 son 0 y 1, respectivamente, siempre se utilizará C2. El segundo miembro del primer par ordenado y_1 es 120=360-240, y el segundo miembro y_2 es 600=360+240. Para las distribuciones continuas se utiliza en el GPSS/H el método de la inversa.

VALORES DIFERENTES A CERO DEL OPERANDO B EN LOS BLOQUES GENERATE Y ADVANCE CUANDO EL OPERANDO A ES UNA FUNCIÓN

Cuando se revisó el uso de funciones en los bloques GENERATE y ADVANCE se tomó al operando B como cero, sin embargo es posible que este operando tenga un valor diferente a cero, si se supone el siguiente caso:

```
ADVANCE FN(TIEMPO), 2.5
```

Cuando este ADVANCE sea ejecutado, el valor del operando A será el producido por la función TIEMPO, entonces será utilizada como la media de la distribución uniforme, en otras palabras, el tiempo de retención será FN(TIEMPO)±2.5.

También el operando B de los bloques ADVANCE y GENERATE puede ser una función, en este caso se tomará al valor producido por el operando B y el valor del operando A, será multiplicado a él.

VI.3 Experimentos para Diseño de Alternativas

La simulación como ya se sabe es una técnica que sirve para analizar que tan bien operará un sistema si es creado o cambiado en cierta forma. Después que un diseño propuesto de un sistema ha sido modelado, es necesario realizar algunos experimentos con el modelo para analizar su comportamiento con algunos cambios. El objetivo es observar el comportamiento en algunos casos específicos para que se hagan inferencias sobre el futuro del sistema. Estas inferencias utilizan métodos de la estadística en el área de inferencia.

VI.3.1 Importancia de Llevar a Cabo Réplicas de un Modelo de Simulación

Llevar a cabo una corrida de una simulación es como lanzar una moneda una vez. El resultado es una observación de un valor de una variable de salida. Si se supone que se lleva una corrida por segunda ocasión utilizando valores diferentes para muestra de una variable de entrada, entonces un valor diferente saldrá de la variable de salida, y así ocurriría si se lleva una tercera simulación, una cuarta, ect. El resultado en promedio al llevar a cabo varias simulaciones puede ser usado para estimar el valor promedio de la variable de salida.

Una *réplica* es una corrida de una simulación, llevada a cabo utilizando una serie de números aleatorios. Estos números aleatorios se rigen por una distribución y esta distribución no debe variar de réplica a réplica, lo que varía son los valores obtenidos por esta distribución.

Comando de control CLEAR

Este comando sirve como ayuda para llevar a cabo dos o más réplicas consecutivas de un modelo cuando un programa es ejecutado. Debe ser ejecutado entre réplicas, y provee una transición de una réplica a otra.

Consiste de la palabra CLEAR, su campo de operación es como se muestra en la tabla. Este comando nunca debe ser etiquetado, pues ocurriría un error de ejecución, no tiene operando. En la *Tabla VI.3.1.1* se muestra un breve resumen del comando CLEAR.

1	Etiqueta (2-9)	Operación (11-20)	Operando (22-72)
		CLEAR	

Operando	Significado	Valores por default o resultado
Etiqueta	No existe aplicación	Error de ejecución

Tabla VI.3.1.1 Resumen del comando CLEAR

Cuando se ejecuta el comando START, las transacciones inician su movimiento y el resultado es una réplica. Si dos o más réplicas deben ser hechas cuando un archivo es procesado, entonces dos o más ejecuciones del comandos START deberán ser ejecutadas, hasta ahora sólo se ha visto cuando un archivo contiene un comando START y cuando el comando END le sigue. La *Figura VI.3.1.1* muestra el orden de los comandos en un archivo (programa), el rectangulo cinco corresponde a comandos de control adicionales que pueden ser escritos entre el comando START y END. Para obtener varias réplicas es necesario incluir los comandos alternos CLEAR y START, un ejemplo de ello es lo siguiente, *Tabla VI.3.1.2*:

```
.....
        START 100
        CLEAR
        START 100
.....
        END
```

Tabla VI.3.1.2 Ejemplo del uso CLEAR

Con la ejecución del primer START, se lleva a cabo la primera réplica y produce un reporte de ella, y después aparece un CLEAR con un START, entonces comienza una nueva réplica.

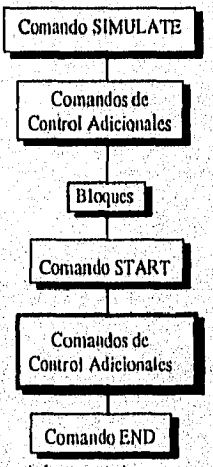


Figura VI.3.1.1 Orden de los comandos en un programa en GPSSIII

Este comando de cierta manera regresa al modelo al comienzo de una simulación; cuando este comando es ejecutado se llevan a cabo algunos pasos:

1. El reloj relativo y absoluto se reinicializan en cero.
2. Todas las transacciones que estaban al final del modelo con la simulación anterior son destruidas.
3. Todas las entidades Facility o Storage, son liberadas.
4. El modelo es reinicializado estadísticamente, es decir, los conteos totales y actuales son asignados con el valor de cero, los conteos de entradas para las facilities, storages y queues son asignadas con el valor de cero, entre otros.

Existen algunos aspectos en los cuales el modelo no es reinicializado, entre ellos se encuentran los siguientes:

1. Los generadores de números aleatorios no son reinicializados, es decir, la siguiente réplica tendrá diferentes valores aleatorios.
2. El contador utilizado para asignar identificadores a las transacciones tampoco es reinicializado.

RÉPLICAS CON EL COMANDO DE CONTROL DO/ENDDO

Si se supone que para un análisis se requieren de 10 simulaciones, esto significará el escribir 10 veces el comando START y nueve el comando CLEAR. Una forma más práctica para realizar lo anterior es el uso de un control anidado, con una secuencia de un START/CLEAR, esto es logrado con el comando DO/ENDDO.

Los comando DO y ENDDO estructuran un control anidado como se analiza en la *Tabla VI.3.1.3*, siendo este control una colección de comandos de control ejecutados en repetidas ocasiones hasta que una condición de terminación ha sido satisfecha, un ejemplo puede ser el uso de comando START y CLEAR en varias ocasiones. Las palabras de operación para estos comando son DO y ENDDO, una etiqueta opcional puede agregarse a uno o a los dos comandos.

```
etiqueta DO  &index=valor_comienzo, valor_limite, incremento
.....
Comandos a ser ejecutados varias veces
.....
etiqueta ENDDO
```

Tabla VI.3.1.3 Uso del comando anidado DO/ENDDO

Los operandos del comando DO especifican un *index*, un *valor_comienzo*, un *valor_limite* y un opcional *incremento* para el index. El index deberá ser una ampervariable, sólo se podrán usar valores enteros en *valor_comienzo*, *valor_limite* y *incremento*. Por default *incremento* será uno, y deberá ser uno o mayor que uno.

Un ejemplo del uso del comando DO es el siguiente:

```
DO  &I=1,10,1
```

Cuando este comando es ejecutado, el índice I tomará 10 valores desde 1 hasta 10. El siguiente ejemplo muestra el uso del comando DO/ENDDO:

```
DO &I=1,10
START I
CLEAR
ENDDO
```

Deberán tomarse en cuenta:

1. Los comandos DO/ENDDO se utilizan para estructurar comandos de control anidados y no bloques anidados
2. Los operandos deben ser escritos consecutivamente, es decir, sin dejar espacio.

Estos comandos son elementos de un lenguaje llamado **Control Statemant Language (CLS)**, existen otros elementos como LET, GETLIST , PUTPIC, IF, ELSEIF, ELSE, ENDIF CALL y GOTO.

AMPERVARIABLES

Estas son utilizadas de la misma manera como se utilizan las variables en un lenguaje de programación, como Pascal o FORTRAN. Son nombres simbólicos para localizaciones en memoria, en donde ciertos valores son almacenados. Estas variables son identificadas con un & es por eso el término *ampervariables*. Excluyendo al &, pueden ser compuestas de uno a ocho caracteres alfanuméricos, siendo la primera alfabética. Existen cuatro tipos de variables: enteras, reales, carácter y externa. Las variables enteras, almacenas valores enteros; las variables reales, números decimales; las de carácter, una cadena de caracteres; y las variables externas identifican funciones y subrutinas externas de librerías externas escritas en FORTRAN o C. Con excepción de las variables externas, los nombres de las variables pueden ser el nombre de una sola localización de memoria o de una lista consecutiva de localizaciones en memoria. Cuando una ampervariable es utilizada como una lista consecutiva, entonces se le agrega un subíndice, por ejemplo &LISTA(10), se refiere a la décima localización de la lista LISTA.

Estas variables juegan un papel integral para un conjunto de capacidades computacionales del GPSS/H. Si estas variables son utilizadas, se deben declarar incluyendo sus directivas de compilador de ampervariables. Se pueden declarar variables enteras, de la siguiente manera *Tabla VI.3.1.4*:

1	Etiqueta (2-9)	Operación (11-20)	Operando (22-72)
	Nunca será utilizado	INTEGER	A,B,C,...

Operando	Significado	Valores por default o resultado
Etiqueta	No existe aplicación	Error de ejecución
A	Nombre de la variable (&I)	Error de ejecución
B,C,...	Nombres adicionales de variables	Fin de los operandos

Tabla VI.3.1.4 Forma de definir valores enteros

Un ejemplo sería el siguiente:

```
INTEGER  &I,&J, &H
INTEGER &UNO, &DOS
```

Estos tipos de comandos pueden ser tecleados después del comando SIMULATE como se muestra en la *Figura VI.3.1.2.*

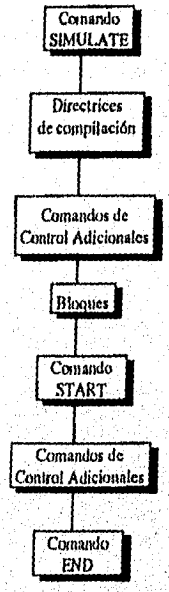


Figura VI.3.1.2 Orden de los comandos en un programa en GPSSIII

Existen cuatro hechos que deberán ser tomados en cuenta en relación con estas variables:

1. Las variables enteras siempre al principio de la ejecución del programa tendrán un valor cero.
2. Los valores de estas variables serán desplegados como parte del reporte de una réplica.

3. La ejecución de comandos CLEAR no cambian los valores a estas variables.
4. Cuando se ejecuta el forma interactiva, el comando **display amp** puede ser usado para desplegar los valores de estas variables.

COMANDO DE CONTROL RESET

Este comando auxilia para llevar a cabo algunos experimentos estadísticos en el GPSS/H, al igual que el comando CLEAR. Al aplicar RESET en un modelo, las transacciones no serán destruidas, simplemente reinicializa los aspectos estadísticos de un modelo y el reloj relativo se reinicializa, sin embargo el reloj absoluto se mantiene. Este comando tiene el efecto de eliminar observaciones estadísticas.

VI.4 COMANDO DE CONTROL PUTPIC

Los reportes de 10 réplicas ejecutadas en un modelo no son identificadas individualmente. La única forma para encontrar el reporte de la quinta réplica es revisar desde el comienzo de los resultados y contar el número de reportes hasta llegar al quinto. Esta situación puede ser remediada con el uso del comando de control PUTPIC, el cual sirve para imprimir una etiqueta al final de cada reporte, su comando equivalente BPUTPIC, es un comando de salida. Existen otros comandos de entrada, como GETLIST o BGETLIST, los cuales se utilizan para leer registros directamente de recursos externos como algún archivo, estos valores pueden ser almacenados en ampervariables, parámetros de transacciones, matrices o lista de valores almacenados.

El comando PUTPIC (PUT out a PICture) se utiliza como el WRITE de FORTRAN, o PRINT de BASIC. Este comando va acompañado de uno o varios comandos más, que presentan información de salida, proveen texto o valores de variables utilizadas en el modelo que deben ser incluidas en la salida e indican en que parte de la salida el texto debe ser escrito.

etiqueta putpic *LINES=número_linea,FILE=nombre_arch,lista_salida* *c... texto o campos editados a ser puestos en el reporte en la línea 1* *c... texto o campos editados a ser puestos en el reporte en la línea2* ..... *c... texto o campos editados a ser puestos en el reporte en la número_linea*

Tabla VI.4.1 Resumen del uso del comando PUTPIC

A continuación se analizará la estructura anterior de la *Tabla VI.4.1*, la *etiqueta* es opcional, tanto *LINES=número_linea,* como *FILE=nombre_arch* y *lista_salida* son opcionales e indican:

- *LINES=número_linea* provee el número de líneas a ser utilizadas por texto o campos editados, el valor por default es uno.

- *FILE=nombre_arch* indica el destino de la salida, producida cuando se ejecuta PUTPIC. Si se escribe FILE=SYSPRINT la salida de comando tendrá el mismo destino que el reporte de la réplica, por default la salida será direccionada a la terminal en uso.

- *lista_salida* es opcional, es la lista de las variables que los valores serán impresos en los campos editados en la salida PUTPIC. Si existen más de dos variables, deberán ser separadas por comas.

La información que le continua al comando PUTPIC corresponde a un renglón de salida producido por el comando. En la columna de cada renglón aparece un carácter (c en la *Tabla VI.4.1*) que controla el espacio vertical usado para cada renglón de texto de salida. Un espacio en esta posición resulta un solo espacio entre renglones; un cero da como resultado doble espacio y un menos (-) triple espacio. A excepción de la primera columna, el resto puede ser el texto o los campos editados. El texto consiste de una lista de caracteres copiados exactamente como aparecerán en la salida. Los campos editados representados por un asterisco (*) indican las posiciones antes, entre o siguiente al texto donde los valores de las variables en la lista de salida son editadas.

Un ejemplo de esto puede ser el siguiente:

```
PUTPIC  LINES=1,FILE=SYSPRINT, (I&)
La información anterior es para la réplica *.
```

Se analizará este ejemplo con mayor profundidad, primero el renglón de PUTPIC:

1. El comando PUTPIC se escribe primero y en este caso no fue etiquetado.
2. LINES=1 indica que sólo hay una línea de texto.
3. FILE=SYSPRINT direcciona la salida de PUTPIC al archivo SYSPRINT.
4. &I es la variable de salida, sus valores deberán ser copiados a la salida de PUTPIC

El siguiente renglón se analiza de la siguiente manera:

1. El primer carácter en la columna uno controla el espacio vertical en la salida, en este caso el carácter fue cero lo que significa que habrá doble espacio.
2. Después continua el texto ("La información anterior es para la réplica") será copiada tal como se escribe aquí.
3. Luego aparece el asterisco (*) que indica la columna de comienzo en el campo de salida en donde el valor de la variable de salida deberá ser copiada.
4. Finalmente más texto continua (".").

El resultado al ser ejecutado el comando PUTPIC será para el caso cuando &I es 5:

```
La información anterior es para la réplica 5.
```

Ahora se estudiará el siguiente ejemplo:

```
      PUTPIC  LINES=3,FILE=SYSPRINT,(I&)
0-----------------------------------------
 La información anterior es para la réplica *.
 ----------------------------------------
```

En este ejemplo se utilizaron tres renglones para el texto de salida, existen algunas diferencias entre los dos modelos:

1. La línea del comando PUTPIC LINES=3 indica que son tres renglones de texto.
2. El primer carácter para controlar los espacios verticales es cero y en los siguientes es un espacio en blanco.

Si este comando es tecleado entre START y CLEAR (o entre los comandos CLEAR y ENDDO) el modelo producirá una etiquetación al final de cada reporte. Si el comando es puesto entre DO y START la etiquetación aparecerá al principio de cada reporte de réplica.

VI.5 Problemas al Analizar los Resultados de un Modelo de Simulación

Siempre que se analizan los datos estadísticos de una simulación, se deben tomar en cuenta ciertas suposiciones sobre la información colectada. Si éstas se hacen sobre un sistema y son incorrectas, se tiene el riesgo de llegar a conclusiones incorrectas sobre el comportamiento del sistema. Cuando se estudian los resultados usualmente se asume lo siguiente:

- Las observaciones son independientes.
- El proceso es de tiempo invariable.
- Cuando el número de observaciones es mayor que 30, la media de la muestra se distribuye como una normal.

Si estas suposiciones se cumplen se pueden usar los siguientes estimadores de la media y la varianza del tiempo de espera en un fila.

$$\overline{X} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{X})^2$$

donde x_i es el tiempo promedio de espera para la réplica i y n es el número de réplicas.

Si n es mayor que 30 con un 95% de confianza el intervalo de la media del tiempo de espera puede usar lo siguiente:

$$P(E(X) = \overline{X} \pm 1.96S / \sqrt{n}) = .95$$

Desafortunadamente en las simulaciones de eventos discretos las suposiciones no se cumplen y para lograr que se cumplan se requiere más tiempo de uso de cómputo. Una solución para los problemas de datos correlacionados y la no estacionaridad, es hacer múltiples simulaciones, independientemente.

TIPOS DE SIMULACIONES

Un sistema de eventos discretos puede ser categorizado como de tipo terminal y no terminal. Se supone que es terminal si los eventos que se manejan en el sistema ocurren en ciertos puntos del tiempo, mientras en uno no terminal los eventos discretos son recurrentes indefinidamente. Cuando estos eventos reocurren, se les llama puntos de regeneración. En un sistema de terminal discreto, el evento T_E señala el final de un jornada.

Es importante distinguir las simulaciones terminales y no terminales, dado que existen diferentes métodos de análisis de resultados para cada caso. Los términos: *terminal, no terminal y de estado estable*, describen a sistemas dinámicos y simulaciones de estos sistemas. Se debe hacer una distinción entre los sistemas y las simulaciones de los mismos: toda simulación es de proceso terminal, pero no todo sistema lo es. Generalmente las simulaciones se distinguen entre terminales y de estado estable, pero es importante distinguir primero a los sistemas entre terminales y no terminales y luego considerar cuándo un sistema llega a un estado estable o si tan solo contiene estados transitorios. La forma de analizar los resultados de una simulación dependen de la naturaleza del mismo (terminal o no terminal) y en sus características de comportamiento (estado constante o estado transitorio), ejemplos de sistemas terminales pueden ser los siguientes:

- Una oficina de pasaportes, abre cada día a las 9:00 a.m. sin trabajadores y cierra a la 1:00 p.m. Cada jornada será diferente, pero al comienzo del día siempre se comenzará sin personas a quien atender y terminará sin clientes a quien atender. El evento terminal será cuando al último trabajador entregue sus papeles; en una simulación de este estilo, una medida de comportamiento sería el tiempo de espera de los trabajadores.

- Un sistema computarizado: un sistema comienza en la mañana cuando el primer usuario se conecta al sistema (este no sería el caso de World Wide Web), y cuando el último usuario se desconecta del sistema, el sistema se apaga. Quizá aquí el punto de interés puede ser en el momento cuando mucha gente se encuentra conectada a él, en general esto ocurre a medio día.

El caso de sistemas no terminales puede ser:

- Un sistema de inventario: un vendedor guarda la mercancía y reordena cuando el nivel del inventario baja. Aunque el vendedor sólo hace negocios por ocho horas, cinco días a la semana, el final de un día de inventario, es el comienzo de otro al siguiente día. Los eventos que se manejan en este sistema continúan indefinidamente.

- Un aeropuerto: en veinticuatro horas, los aviones llegan y se van. Aunque existen periodos de mucha y poca actividad, el estado del aeropuerto en la mañana depende de como se manejo el tráfico la tarde anterior.

Cuando se analizan los resultados de una simulación, es importante diferenciar si los datos obtenidos fueron en fase transitoria o en fase estable. Un ejemplo de sistema que llega a un estado constante puede ser el siguiente:

- Un banco de sangre colecciona y guarda la sangre, dotándoselas a los hospitales cuando sea necesaria. La demanda de sangre es uniforme durante todo el año, existirá una distribución de estado constante de la sangre inventariada.

Existen algunos sistemas que no alcanzan estados estables, como el siguiente:

- Usuarios de un sistema computarizado de tiempo compartido, conectado al sistema entre las 8:00 a.m. y las 5 p.m. La cantidad de usuarios que se conectan a él, varía conforme el día avanza.

Los métodos utilizados para coleccionar y analizar la información de una simulación de un sistema de eventos discretos depende de si el sistema es terminal o no terminal y si la distribución de los estados del sistema se comporta de forma estable o transitoria o ambas.

Con respecto al análisis estadístico de los resultados de simulaciones, es apropiado distinguir entre las simulaciones con terminación y con estado estable. Una simulación con terminación es aquella donde las medidas de comportamiento de un sistema están definidas relativamente con un intervalo de tiempo simulado $[0,T_E]$, donde la T_E es el instante en el tiempo simulado donde el evento E ocurre. Este evento se especifica antes del comienzo de una simulación. Las simulaciones de estado constante son aquellas donde las medidas de comportamiento están definidas como límites conforme su tamaño se va al infinito. Dado que no existe un evento E que pueda terminar con la simulación, el tamaño de la misma se hace lo suficientemente grande para obtener *buenas* estimaciones de las cantidades de interés.

Una simulación de estado estable implica una situación donde cierto sistema ha alcanzado un condición estable de operación y continua operando indefinidamente dentro de estas condiciones. Con esta idea se puede suponer que las distribuciones de las variables de entrada no cambian con el tiempo y que las condiciones del comportamiento del sistema actual no dependen de las condiciones con las que el sistema dio inicio.

INTERVALO S DE ESTIMACIÓN PARA VALORES ESPERADOS

El objetivo fundamental al observar valores de una variable de salida es estimar su valor esperado, si se crea cierto modelo de un sistema y se lleva a cabo una réplica, el resultado obtenido de esa réplica es un punto estimado del valor esperado; es un número utilizado

como una estimación del valor de un parámetro de una población no conocido. Si se llevan a cabo dos o más réplicas, entonces cada réplica produce un valor estimado independiente.

Un intervalo estimado de un valor esperado es un par de números y entre ellos, con cierto grado de confianza, se encuentra el valor. El coeficiente de certeza tal como el 90% o el 95% es el nivel de confianza o grado de confianza.

Para estimar intervalos de confianza con los valores obtenidos se usan técnicas estadísticas, para muestras pequeñas menores a 30, se asume que los puntos estimados son independientes entre ellos y que son de distribuciones idénticas, también se asume que para el caso de una muestra pequeña, que los puntos estimados están distribuidos normalmente. El intervalo de confianza para una muestra pequeña es el siguiente:

$$\bar{x} \pm t_{\alpha/2}(s/\sqrt{n})$$

donde $\bar{x}$ es la media de los puntos estimados en la muestra de tamaño n, s es la desviación estándar de la muestra y $t_{\alpha/2}$ es el estadístico t con un nivel de confianza $100(1-\alpha)\%$ basada con $n-1$ grados de libertad.

Si se desean obtener los niveles de confianza de una muestra grande, es decir mayor que 30, se usa la siguiente formula.

$$\bar{x} \pm Z_{\alpha/2}(s/\sqrt{n})$$

en este caso en vez de utilizar el estadístico t se utiliza el estadístico Z.

ANÁLISIS DE RESULTADOS PARA SISTEMAS TERMINALES

El método más usual para estimar las características de un sistema, utilizando un modelo de simulación, es coleccionar muestras de las características de interés, conforme el modelo es ejecutado. Una vez hecho esto, la información obtenida puede ser usada para establecer el punto y el intervalo estimado de esa característica. Pero antes de considerar como se deben analizar los datos, es importante considerar las propiedades de los dos estimadores más usados, para la muestra x_i, i=1,2,...n:

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{X})^2$$

Se puede demostrar que $E(x_i) = \mu$ para toda i entonces $E(X) = \mu$, no es necesario que toda x_i sea una observación independiente, pero sólo que todas tengan un mismo valor esperado. Y para que $E(S^2) = \sigma^2$, se cumpla es necesario que cada x_i tenga un valor esperado común y que sean independientes.

Para una simulación terminal, el método más usado para asegurar que las observaciones x_i sean independientes y tengan un valor común esperado, es el uso de réplicas, es decir, la simulación se repite varias veces, siendo cada réplica independiente de la otra. Durante el curso de cada simulación, ciertas observaciones se hacen en ciertos puntos en el tiempo o con la ocurrencia de ciertos eventos. Usualmente las características y comportamiento del sistema son diferentes durante las distintas fases del sistema.

Si una simulación es réplicada R veces, con K observaciones intermedias, entonces la simulación lleva a:

$$\overline{X}_j = \sum_{i=1}^{R} x_{ij} / R \qquad j = 1,2,...K$$

$$\overline{Y} = \sum_{i=1}^{R} y_i / R$$

$$S_j^2 = \sum_{i=1}^{R} (x_{ij} - \overline{X}_j)^2 / (R-1) \qquad j = 1,2,...K$$

$$S_Y^2 = \sum_{i=1}^{R} (y_j - \overline{Y})^2 / (R-1)$$

x_{ij} = la *j-ésima* observación de la *i-ésima* réplica, donde i=1,2,...R y j=1,2,....K, y sea y_i el comportamiento general de la medida de comportamiento durante la *i-ésima* réplica.

Una vez que la media y los puntos de los estimadores se han establecido, utilizando las ecuaciones anteriores, se puede aproximar un intervalo de confianza del $100(1-\alpha)\%$ para $E(x_{ij})$ y $E(y_i)$ utilizando los estimadores siguientes:

$$P(\mu_j = \overline{X}_j \pm t_{\alpha/2,(R-1)} S_j / \sqrt{R}) = 1-\alpha$$

$$P(\mu_Y = \overline{Y} \pm t_{\alpha/2,(R-1)} S_y / \sqrt{R}) = 1-\alpha$$

La mitad del intervalo es:

$$I = t_{\alpha/2,(R-1)} S / \sqrt{R}$$

Se supone que los x_{ij} y y_i están distribuidos normalmente, si esto no se cumple, entonces el intervalo anterior no será confiable, pero con base en el teorema del límite central, se puede saber que, mientras mayor sea el número de réplicas, la media de la muestra tiende a estar distribuida normalmente.

ANÁLISIS DE RESULTADOS DE SISTEMAS NO TERMINALES

Cuando se analizan simulaciones de sistemas no terminales, se deben analizar varios problemas que no ocurren con simulaciones terminales.

Estos problemas son:

- Condiciones Iniciales
- Covarianza entre muestras
- Tamaño de la corrida

Existen varios métodos para analizar este tipo de sistemas, entre ellos se encuentran:

- ***Réplicación.*** Los resultados pueden ser analizados utilizando métodos similares a los usados con simulaciones terminales. Sin embargo, para este tipo de simulaciones se debe minimizar el efecto de las condiciones iniciales en los resultados. Una forma de evitar el efecto de estas condiciones, es desechar las observaciones obtenidas durante la primera fase de la simulación y utilizar la información recaudada cuando el sistema llega a un estado estable. Al hacer esto se puede causar algún conflicto, como sería: ¿cómo saber qué información desechar y en qué momento empezar analizar la información? Se tiene que hacer una primera simulación y analizar cuándo el sistema ha llegado a un estado estable, y muchas veces se utiliza el método de ensayo y error.

- ***Batching.*** Estos métodos se basan en una simulación en el cual periódicamente se registran las medidas estadísticas, y se aplica un RESET o CLEAR estas medidas, como un nuevo comienzo.

- ***Autocorrelación.*** A diferencia de los métodos de batching, intentan eliminar o reducir la correlación entre muestras. Los métodos que utilizan la autocorrelación incluyen el efecto de la correlación entre observaciones para estimar la varianza del proceso en simulación.

- ***Puntos de Regeneración (Simulación Regenerativa).*** Este método evita el uso de condiciones iniciales, al usar puntos de regeneración, siendo estos puntos como un estado del sistema en donde el comportamiento futuro es independiente de la historia pasada.

VI.6 EXPERIMENTOS ESTADÍSTICOS PARA COMPARAR ALTERNATIVAS

El siguiente procedimiento estima si existe una diferencia entre los valores esperados de dos variables de salida, una por cada alternativa, permite determinar estadísticamente si una alternativa es realmente mejor que otra:

1. Hacer n réplicas con cada una de las alternativas.
2. Poner en pareja la réplica i de la primera alternativa con la réplica i de la segunda alternativa, $i=1,2,3,...n$
3. Calcular la diferencia en cada pareja, la media y la desviación estándar de esta diferencia.
4. Obtener el intervalo de confianza para las diferencias, con un cierto nivel de significancia. Si el intervalo de confianza excluye al cero, entonces se puede creer que las variables de

salida difieren a su valor esperado; si los intervalos de confianza incluyen al cero, existen posibilidades de las variables de salida no difieran a sus valores esperados.

Las réplicas en el paso uno, son independientes entre las réplicas mismas y también con respecto a las réplicas de la otra alternativa. Esta aproximación de comparar dos poblaciones y reducirlas a una sola facilita la labor de análisis.

Si se supone que en el paso cuatro el intervalo de confianza excluye al cero, a cierto nivel de confianza, usualmente un 95% de confianza que se puede rechazar con un 5% de desconfianza, la hipótesis de que no existe diferencia entre los dos valores esperados de las variables de salida, es decir, existe diferencia entre las dos variables. Ahora, si en intervalo de confianza se encuentra el cero con el 95% de confianza, entonces esto es equivalente a aceptar la hipótesis con un 5% de que no existe diferencia entre los valores esperados de las variables de salida.

Para llevar a cabo este procedimiento se requiere que las diferencias pares sean independientes entre ellas e idénticamente distribuidas. En el caso de que las muestras sean menos de 30 deberán ser distribuidas normalmente. Se utiliza el estadístico *t* para resolver en el paso cuatro los intervalos de confianza cuando los pares son no correlacionados. Si los resultados de las réplicas son de un proceso de obtención de promedios, entonces por la generalización del teorema del límite central tienden a una distribución normal esto tan solo para ciertos tipos de correlación, permitiendo que las diferencias satisfagan el requerimiento de la distribución normal en forma asintótica.

En el caso de una muestra mayor de 30, ni los resultados de la simulación ni las diferencias necesitan estar distribuidas normalmente. Por el teorema del límite central, el promedio de las diferencias en pares tenderá a una distribución normal conforme la muestra sea mayor, lo cual justifica el uso del estadístico Z.

Esta aproximación no requiere que las varianzas de las dos variables de salida sean iguales. Esto es importante pues la igualdad entre ellas no ocurre muy a menudo. Se supone que las dos réplicas que hace la pareja son independientes. Hay que hacer notar que la aproximación de usar pares-*t* no correlacionados y comparar las dos poblaciones, reduce el problema a uno de una sola población, siendo ésta la diferencia.

Para ilustrar lo anterior se verán dos alternativas del problema visto en el Capítulo II. Estas dos alternativas serán:

- Un solo servidor (Sofía)
- Dos recepcionistas idénticas(Sofía y Mari)

Se supondrá que son muestras independientes, en este caso la variable de salida de interés será el tiempo promedio de espera de los trabajadores y el objetivo será identificar al servicio que tiene el valor esperado menor. Para estos modelos se llevaron a cabo 20 réplicas de cada modelo. Los resultados se muestran en la *Tabla VI.6.1*:

Réplica	Promedio de espera de un solo servidor	Promedio de espera de dos servidores idénticos	Diferencia
1	8.283	3.822	4.461
2	8.570	6.687	1.883
3	6.452	6.106	0.346
4	8.268	5.565	2.703
5	9.533	5.375	4.158
6	8.135	5.672	2.463
7	7.436	3.376	4.06
8	7.543	3.584	3.959
9	7.071	4.571	2.5
10	8.306	4.471	3.835
11	6.802	5.120	1.682
12	7.799	4.738	3.061
13	6.850	5.666	1.184
14	9.346	6.661	2.685
15	6.418	5.455	0.963
16	6.541	4.865	1.676
17	8.989	3.771	5.218
18	5.446	6.019	-0.573
19	7.186	4.422	2.764
20	6.413	4.775	1.638
	Media de la muestra		2.5333
	Desviación estándar de la muestra		2.8431112
	Intervalo de confianza 90%		[3.378,1.68906]
	Intervalo de confianza 95%		[3.6324,1.4349]
	Intervalo de confianza 99%		[4.1474,0.91915]

Tabla VI.6.1 Réplicas para comparar dos alternativas del ejemplo del Capítulo II

Utilizando la información y determinando los estimadores con un estadístico t que se muestran en la *Tabla VI.6.1*, se puede concluir que existe una diferencia estadística entre usar un servidor y dos servidores. El usar dos servidores es la mejor opción para los trabajadores, sin tomar en cuenta los costos por traer a una nueva recepcionista. Es importante que en el ejemplo también se analicen los costos que puede acarrear traer a una recepcionista. Aunque estadísticamente la solución sea traer a una persona, no se puede concluir que esa sea la mejor opción, es importante tomar en cuenta otros factores, como el espacio en la oficina y el sueldo, entre otros.

OTROS TIPOS DE INTERVALOS DE CONFIANZA PARA LA DIFERENCIAS DE DOS VALORES ESPERADOS DE DOS POBLACIONES, EN BASE A MUESTRAS INDEPENDIENTES.

Existen varias alternativas para crear intervalos de confianza para las diferencias de los valores esperados de dos poblaciones basados en su independencia (no correlación). Los métodos varían de acuerdo a sus suposiciones, entonces sucede que un método puede ser mas conveniente y útil que otro en casos específicos. Se comentarán tres casos:

- El estadístico *t*
- El estadístico Z
- Intervalos de confianza Welch

El criterio para distinguir entre ellos es el siguiente:

- ¿Los tamaños de muestras de las dos poblaciones deben ser iguales?
- ¿Las varianzas en las dos poblaciones deben ser iguales?

El método visto anteriormente requiere que los tamaños de muestras sean los mismos, pero no varianzas iguales. La igualdad en el tamaño de muestra será fácil obtener si se están manejando dos poblaciones simuladas.

Existe otra forma de obtener el intervalo además de la vista la cual utiliza el estadístico *t* cuando los tamaños de muestras son menores de 30. No requiere que los tamaños de muestras sean los mismos pero requiere que las variables de salida estén distribuidas normalmente y que su varianzas sean las mismas. La necesidad que las varianzas sean las mismas limita el uso de este estadístico. Para un mayor análisis sobre lo anterior se recomienda del uso de libros en la materia.

Si el tamaño de muestra es mayor que 30 entonces se utiliza el estadístico Z, en este caso las variables de salida no necesariamente deben estar distribuidas normalmente y las varianzas no deben ser iguales para las dos poblaciones.

Los intervalos de confianza Welch son una variación del antes mencionado, asumiendo que las varianzas en las dos poblaciones no son iguales y que los tamaños de muestra son menores que 30, y tampoco son iguales. En esta aproximación se debe trabajar con las varianzas para estimar los grados de libertad usados en el estadístico *t*.

COMPARAR ALTERNATIVAS DE DISEÑOS

Cuando se comparan dos alternativas de diseños, los resultados de las simulaciones son probabilísticos, entonces es importante hacer algunas pruebas estadísticas para llegar a conclusiones válidas. Muchas veces las diferencias en el comportamiento del sistema pueden ser significativas estadísticamente y no materialmente. Si las pruebas estadísticas son aceptables, ciertas suposiciones sobre los datos obtenidos de las simulaciones deben ser válidos, siendo el más importante su independencia estadística. Si se asume que existen dos alternativas y dos conjuntos de simulaciones, se pueden tener varios casos:

Caso 1: las dos muestras son independientes y sacadas de dos poblaciones con varianzas iguales.
Caso 2: las muestras son independientes con varianzas no iguales.
Caso 3: las muestras son correlacionadas.

*Caso 1 y caso 2, **muestras independientes.*** Las medidas del comportamiento de interés de la simulación se obtienen de dos muestras independientes: Y_{11} , Y_{21} ,..., Y_{n1} para el diseño uno y Y_{12}, Y_{22},..., Y_{n2} para el diseño dos. Cada Y_{ij} representa el valor individual *i* para la media de comportamiento de una sucesión independiente de observaciones para el diseño *j*. Se asume que las muestras son del mismo tamaño, para probar la hipótesis de que si existe una diferencia significativa entre los dos diseños, se forma un intervalo de confianza de las diferencia en las medias de las medidas de comportamiento de interés $\mu_1 - \mu_2$. El intervalo de confianza se hace en base a las medias de las dos muestras. Dependiendo a la posición de este intervalo de confianza con respecto al cero, se puede llegar a una de las tres posibles conclusiones:

1. La media de la población uno es menor que la dos.
2. No existe una diferencia significativa entre las medias.
3. La media de la población uno es mayor que la de la población dos.

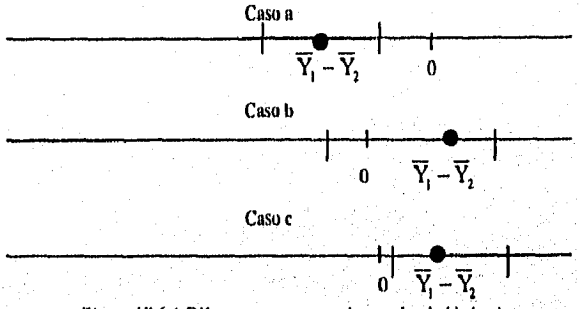


Figura VI.6.1 Diferentes casos para la prueba de hipótesis

Como se puede ver en la *Figura VI.6.1*, en el caso a) el intervalo se encuentra a la izquierda del cero, se puede concluir que $\mu_1 - \mu_2 < 0$ equivalente a $\mu_1 < \mu_2$. b) en el intervalo se encuentra el cero, se concluye que no existe una diferencia estadística entre μ_1 y μ_2. c) el intervalo se encuentra a la derecha del cero, se concluye que $\mu_1 - \mu_2 > 0$, es decir que $\mu_1 > \mu_2$. La hipótesis sería:

$$H_0: \mu_1 - \mu_2 = 0$$
$$H_1: \mu_1 - \mu_2 \neq 0$$

El intervalo de confianza es:

$$\overline{Y}_1 - \overline{Y}_2 \pm t_{\alpha/2,\nu} \text{s.d.}(\overline{Y}_1 - \overline{Y}_2)$$

donde $\overline{Y}_1$ y $\overline{Y}_2$ son:

$$\overline{Y}_1 = 1/n \sum_{i=1}^{n} Y_{i1}$$

$$\overline{Y}_2 = 1/n \sum_{i=1}^{n} Y_{i2}$$

donde:

s.d. $(\overline{Y}_1 - \overline{Y}_2)$: es la desviación estándar del estimador de las diferencias con los parámetros $\mu_1 - \mu_2$.

α: el nivel de confianza para la prueba.

v: son los grados de libertad asociados con la distribución *t*.

n: tamaño de muestra.

Para obtener la desviación estándar de la diferencia de las medias de la muestra, se calcula lo siguiente:

$$\begin{aligned} Var(\overline{Y}_1 - \overline{Y}_2) &= Var(\overline{Y}_1) + Var(\overline{Y}_2) \\ &= \sigma_1^2 / n + \sigma_2^2 / n \end{aligned}$$

cada σ_j^2 puede ser obtenida de la siguiente manera:

$\hat{\sigma}_j^2 = s_j^2 = \sum (Y_{ij} - \overline{Y}_j)^2 / (n-1)$, donde s_j^2 es la varianza de la muestra.

Caso 1: ***muestras con varianzas iguales.*** Puede ser razonable asumir que $\sigma_1^2 = \sigma_2^2$ o si una prueba F confirma lo anterior, se puede estimar la varianza de la siguiente manera:

$$\hat{\sigma}^2 = s_p^2 = \frac{(n-1)s_1^2 + (n-1)s_2^2}{2n-2}$$

tiene 2n-2 grados de libertad. Y la varianza de la diferencia es:

$$Var(\overline{Y}_1 - \overline{Y}_2) = 2s_p^2 / n$$

*Caso 2: **muestras con diferentes varianzas.*** Si lo anterior no se cumple, entonces se ocupan las siguientes fórmulas:

$$s.d.(\overline{Y}_1 - \overline{Y}_2) = \sqrt{(s_1^2 + s_2^2)/n}$$

los grados de libertad se pueden obtener aproximadamente como:

$$v = \frac{((s_1^2 + s_2^2)/n)^2}{((s_1^2/n)^2 + (s_2^2/n)^2/(n+1)} - 2$$

*Caso 3: **muestras correlacionadas.*** Un efecto deseable al usar muestras positivamente correlacionadas, es el de reducir la varianza de las diferencias, $Y_1 - Y_2$. Si esto se logra entonces el intervalo asociado puede ser reducido, el coeficiente de correlación r_{12} indica el grado en que las dos muestras se comportan similarmente.

La varianza en general puede ser escrita de la siguiente manera:

$$\begin{aligned} \mathrm{Var}(\overline{Y}_1 - \overline{Y}_2) &= \mathrm{Var}(\overline{Y}_1) + \mathrm{Var}(\overline{Y}_2) - 2\mathrm{Cov}(\overline{Y}_1, \overline{Y}_2) \\ &= \sigma_1^2 / n + \sigma_2^2 / n - 2r_{12}\sigma_1\sigma_2 / n \end{aligned}$$

donde $r_{12} = \mathrm{Cov}(Y_{i1}, Y_{i,2}) / \sigma_1\sigma_2$

y $\mathrm{Cov}(\overline{Y}_1, \overline{Y}_2) = \mathrm{Cov}(Y_{i1}, Y_{i,2}) / n$

Sea V_i la varianza de la diferencia de muestras independientes y sea V_c la varianza de las muestras correlacionadas. Entonces

$V_c = V_i - 2r_{12}\sigma_1\sigma_2 / n$

Y dado que $r_{12} > 0$, entonces $V_c < V_i$

Para encontrar el intervalo de confianza necesario para la prueba de hipótesis entre los diseños se debe definir una nueva variable aleatoria:

$$D_i = Y_{i1} - Y_{i2}$$

Se hace el mismo análisis que el del caso 1.

Es importante también analizar los *costos* de las diferentes alternativas, y no basar la decisión en cual opción es la mejor exclusivamente por pruebas estadísticas, para determinar de una manera más apropiada la elección, con base en los resultados estadísticos y los costos. Esto puede ser analizado desde la perspectiva que plantean la teoría de juegos, análisis financiero, análisis de decisiones, ect.

CONCLUSIONES

Como se ha citado desde el inicio del presente trabajo, la simulación representa una herramienta de gran utilidad para realizar de manera más eficaz la toma de decisiones. Conforme la competencia en los mercados internacionales se ha venido incrementado, el uso de la simulación se hace cada vez más común en las empresas originarias de los países altamente industrializados y dada la situación actual de nuestra nación, tanto los empresarios como el gobierno ya no pueden seguir ignorando o desechando la posibilidad de emplear las metodologías de optimización que presentan las matemáticas aplicadas, pues se estaría en enorme desventaja competitiva ante las industrias que sí hacen uso de ellas.

Paralelo a esto, es necesario que se trate de promover en mayor medida la vinculación de las instituciones de educación superior con el sector productivo del país, propiciando así una retroalimentación muy significativa para el desarrollo académico y económico de la nación, ya que al efectuarse este fenómeno, es indiscutible que se lograría obtener una mayor visión de lo que se requiere, plantear de mejor manera los planes de estudio, las investigaciones y crear, quizá, nuevas licenciaturas o posgrados que permitan alcanzar un nivel muy superior al que ahora se tiene.

Pero antes de esperar a que esto suceda es importante que nosotros, como egresados de la licenciatura de Matemáticas Aplicadas y Computación, difundamos el uso de la simulación y de las técnicas vistas a lo largo de la carrera en nuestra área de trabajo, ya que al no ser conocidas la única forma de que se despierte el interés en éstas es mostrando los enormes beneficios que genera su empleo adecuado.

Aunque, por supuesto, hay que manejar este procedimiento con cierta cautela, ya que al primer error o un mal resultado puede ser posible que se desconfíe de estas herramientas y de nosotros mismos, al ser algo novedoso para muchos empresarios, éstos no respaldan fácilmente un proyecto de estas características o lo que es peor, dada la ideología de los administradores en México se recurre a contratar empresas de consultoría extranjera para auxiliarlos en su labor, pensando en que por el simple hecho de no pertenecer al país son mejores o más capaces, cuando en nuestras instituciones educativas existen verdaderos expertos en estas áreas.

Para llevar a cabo adecuadamente una simulación es necesario efectuar una correcta planeación, así como un análisis económico sobre las posibilidades de un cambio de sistema o creación del mismo, además de un buen examen para la determinación de qué tipo de

estructura implementar como simulador del sistema real, lo que hace necesario un estudio detallado para lograr la mejor representación con base en los objetivos propuestos.

Muchos de estos modelos pueden ser manipulados con la computadora, a través del uso de lenguajes de propósito general y específico. Como se pudo analizar, existen grandes cualidades en los lenguajes de simulación, además de que hay una gran variedad de ellos, con diferentes capacidades. Este trabajo se centró en el lenguaje de simulación GPSS/H, el cual ya tiene varias décadas de existencia con distintas versiones que con el paso del tiempo han ido mejorando. Este lenguaje puede ser aplicado en una gran variedad de modelos de eventos discretos, tanto modelos con un servidor como varios servidores, así como con diferentes patrones de llegadas. Es importante recalcar que el uso de este lenguaje da la posibilidad de realizar simulaciones de muy alta calidad, sumamente eficientes, además de permitir una comunicación amigable del analista del sistema con el administrador, o con personas ajenas a la toma de decisiones pero importantes en el modelo, dadas las capacidades gráficas que posee el GPSS/H con uso de un paquete de animación. Pero, para lograr todo lo anterior, a riesgo de ser reiterativo; es necesario plantear una buena estructura del modelo ya que sin ella ni el más perfecto de los lenguajes generaría los resultados adecuados.

La simulación permite que los estudiantes practiquen con modelos de sistemas reales utilizando la computadora, logrando así un apoyo pedagógico importante, ya que no tendrán que trabajar con el fenómeno físicamente y podran obtener conomiento basico y experiencia del sistema, que quiza sin el uso de la simulación sería imposible. En muchas ocasiones puede ser indispensable que los alumnos tengan que practicar en modelos pues hacerlo con el sistema real podría significar grandes perdidas al minimo error, ya que con el uso de la simulación el estudiante puede equivocarse sin causar algun daño y logrando así un conocimiento necesario.

Finalmente, es importante mencionar la necesidad de una correcta interpretación de los resultados. No se puede tomar una decisión con tan solo un análisis, es necesario hacer ciertas pruebas estadísticas, lo que implica hacer varias réplicas del modelo, para lograr conclusiones aceptables, además de contemplar la realización de estimaciones económicas que permitan visualizar la factibilidad de llevar a cabo el modelo.

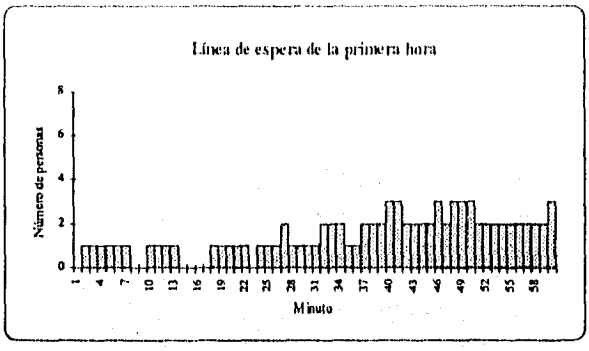
Como conclusión final hay que mencionar que la mejor manera de optimizar tiempo, recursos materiales y humanos, es la planeación de nuestros proyectos, no es posible a estas alturas seguir dejando a la deriva el futuro de nuestras estructuras educativas, de las empresas o el desarrollo del país, ya que si no se cambia de actitud, si no se implanta como doctrina el trabajo, la necesidad de buscar más conocimientos y adecuarlos a los retos que plantea nuestra vida diaria, ni la mejor metodología o herramienta pueden lograr que resurjamos como una nación próspera a inicios del siguiente milenio.

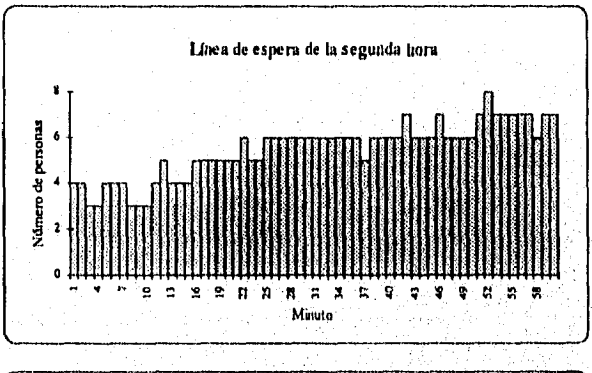
ANEXO
CAPÍTULO II

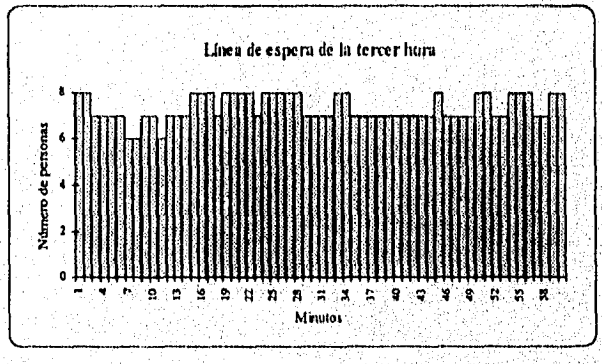
Muestra de observaciones durante tres horas en la oficina de pasaportes en la Secretaría de Trabajo y Previsión Social, obtenidos el 14 de julio de 1995.

Llegadas	Tiempo de Llegada	Siguiente Llegada	Tiempo de Servicio
1	1	8	6
2	9	4	4
3	13	4	5
4	17	3	6
5	20	4	4
6	24	3	4
7	27	5	7
8	32	5	7
9	37	3	5
10	40	6	4
11	46	2	6
12	48	9	6
13	57	3	5
14	60	1	5
15	61	4	6
16	65	6	4
17	71	1	7
18	72	4	7
19	76	3	6
20	79	3	4
21	82	3	6
22	85	5	6
23	90	8	5
24	98	4	4
25	102	4	4
26	106	5	7
27	111	1	5
28	112	7	6
29	119	2	6
30	121	8	5
31	129	3	6
32	132	3	6
33	135	4	5
34	139	5	5
35	144	9	6
36	153	7	5
37	160	5	6
38	165	5	7
39	170	0	6
40	174	4	4
41	179	5	4

Gráficas de la variable de estado *n(t)*, es decir, el número de personas, contra el tiempo (minuto t), durante tres horas seguidas.







ANEXO
EJERCICIOS PROPUESTOS

A continuación se presentan una serie de ejercicios recopilados de varios libros, esperando sean de utilidad, los ejercicios están de forma que conforme se vaya avanzando con la lectura del trabajo, y el lector pueda resolverlos.

1. Describir la forma más adecuada para estudiar cada uno de los siguientes sistemas, en base a la *Figura 1.6.1* y ¿porqué?
   - Un elevador de un edificio con cinco pisos
   - Una sala de emergencia de un hospital
   - Una caja en un supermercado
   - Un sitio de taxis
   - Recepción de mensajes en una computadora de un solo servidor

2. Si se supone que se usa la simulación en los sistemas anteriores, determinar si la simulación de cada uno de los sistemas debe ser estática o dinámica, determinística o estocástica y continua o discreta.

3. ¿Qué aplicación puede generar la combinación de un método de optimización y la simulación y cómo se relacionan estas dos técnicas?

4. ¿Cómo se pueden simular manualmente las siguientes actividades:
   - Lanzamiento de una moneda cargada,
   - Tiempo de llegada de un enfermo a un hospital y
   - Tiempo de servicio de un doctor?

5. Indicar cuales son las entradas, componentes, funciones y salidas de los siguientes sistemas:
   - Fábrica de coches
   - Oficina de pasaportes
   - Hospital
   - Supermercado

6. Discutir la sección de simulación del libro *Introducción a la Investigación de Operaciones*, de Frederick S. Hiller y Gerald J. Lieberman.

7. Buscar y analizar en Internet, algún artículo sobre una aplicación de la simulación, se muestran algunas sugerencias.
   - http://iml.iw.uni-hannover.de/paper/chap5.html
   - http://www.acl.lanl.gov/sunrise/TRANSIMS/intro.html

- http://www.chem.ethz.ch/D-CHEM-Prof/vangunsteren/vangunsterenres.html
- http://pat.mdc.com/ASDS/ASDS_Home.html
- http://meteor.anu.edu.au/dave/od/cosmo.html

8. Buscar algún paquete estadístico y analizar cómo se generan valores aleatorios en el paquete (sugerencia: *Statgraphics).*

9. Investiga en tu comunidad dónde se usa la simulación y qué tipo de simulaciones se hacen, hacer una breve descripción.

10. Investiga en tu comunidad dónde se usa la investigación de operaciones y hacer una breve descripción.

11. Asistir a un museo (Museo del Niño) o algún parque de diversiones (Reino Aventura) donde exista una simulador y describir su experiencia.

12. El clima se puede considerar un sistema estocástico, porque evoluciona de una manera probabilística de un día a otro. Suponer que para cierto lugar este comportamiento es aleatorio y satisface la siguiente descripción:

    - La probabilidad de lluvia para mañana es 0.6 si hoy llueve
    - La probabilidad de un día despejado (sin lluvia) para mañana es 0.8 si hoy esta despejado

    Simular el comportamiento del clima durante cinco días, comenzando con un día que sigue a uno despejado.

13. Un juego de dados requiere que el jugador lance dos dados una o más veces hasta que llegue una decisión: perder o ganar. Gana si la primera tirada suma 7 u 11, o si la primera es 4, 5, 6, 8, 9 ó 10 y sale la misma suma antes de que aparezca una suma de 7. Por el contrario, pierde si el resultando de la primera tirada suma 2, 3 ó 12, o si la primera suma es 4, 5, 6, 8, 9 ó 10 y aparece una suma de 7 antes de que la primera suma vuelva a salir.

    a) Simular cinco jugadas para iniciar el proceso de estimación de la probabilidad de ganar.
    b) Para un número grande de jugadas, la proporción de veces que se gana tiene una distribución *normal* con media = 0.493 y desviación estándar $=0.5 / \sqrt{n}$. Utilizar esta información para calcular el número de jugadas simuladas que se requieren para obtener una probabilidad de al menos 0.95 de que la proporción de veces que gana sea menor que 0.5.

14. En algún libro buscar como se plantea la planeación de la simulación y determinar sus diferencias entre lo propuesto y lo planteado en este trabajo.

15.Tomar un ejemplo de un sistema real, con un servidor y una línea de espera, obtener los datos reales durante un intervalo de tiempo razonable y analizarlo como el ejemplo visto en el trabajo.

16.Determinar en base a los datos anteriores cómo es que se distribuye la población teórica, usando un paquete estadístico.

17.Del sistema real anterior, determinar cuál es el mecanismo de servicio, la fuente de entrada y disciplina de servicio.

18.Investigar las diferencias entre validación y verificación.

19.Investigar los distintos lenguajes de simulación existentes en el mercado y sus características principales, se sugiere utilizar revistas, visitas a los centros de cómputo de la UNAM, IIMAS, y FI(Posgrado), entre otros y buscar en Internet.

20.Investigar y analizar las diferentes técnicas de reducción de varianza.

21.Diseñar el siguiente programa en GPSS/H utilizando este modelo:

```
GENERATE 100
TERMINATE 1
```

Utilizar el operando A con valor de tres en el comando de control START, ¿en qué tiempo simulado la transacción saldrá del bloque GENERATE? y ¿en qué momento la simulación terminará? Correr el programa en forma interactiva.

22.Diseñar el siguiente programa en GPSS/H utilizando este modelo:

```
GENERATE 50
TERMINATE 1

GENERATE 35
TERMINATE 0
```

Utilizar un valor de tres en el operando A del comando de control START, ¿en qué momento las transacciones saldrán de los dos bloques GENERATE?; ¿cuál será su identificador? y ¿en qué momento la simulación finalizará?

23.Las transacciones salen del bloque GENERATE cada 25±10 minutos, y luego son destruidas inmediatamente.

- Diseñar el modelo para que exactamente 20 transacciones sean creadas.
- Diseñar el modelo para que la simulación dure tres horas seguidas.

24.Crear el siguiente programa:

```
        GENERATE 60,30
        TERMINATE 1
*
        GENERATE 50,25
        TERMINATE 2
```

Usar el valor 5 para el operando A en el comando de control START, ¿en qué momentos saldrán las transacciones de los bloques GENERATE y cuál será su identificador?

25.Programar los siguientes programas:

```
*A                          *B
        GENERATE 20,30              GENERATE 100
        TERMINATE 1                 TERMINATE 0
```

Emplear el valor de 5 para el operando A, ¿cuál es el problema con estos modelos y qué ocurrirá al correrlo en forma interactiva?

26.Examinar los siguientes programas:

```
*A                          *B
        SIMULATE                    SIMULATE
        GENERATE 35,15              GENERATE 35,15
        TERMINATE 1                 GENERATE 20,10
        GENERATE 20,10              TERMINATE 1
        START 3                     START 3
        END                         END
```

¿Qué ocurrirá cuando se corran estos programas? Si existe algún error en los problemas modificarlos y ejecutarlos en forma *test*.

27.Crear los siguientes programas:

```
*A                          *B
        GENERATE 50                 GENERATE 100,50
        ADVANCE                     ADVANCE 125,110
        250,250                     ADVANCE 75,40
        TERMINATE 1                 TERMINATE 1
```

El operando A del comando START tiene por valor cinco, ejecutar una simulación en modo *test*, ¿en qué momentos las transacciones avanzarán al bloque TERMINATE, en los dos ejemplos?; ¿en qué instante terminará la simulación?

28.Diseñar el siguiente programa:

```
*
        GENERATE 50
        ADVANCE 50
        TERMINATE 1
        GENERATE 75
        ADVANCE 0
        TERMINATE 1
```

El operando A del comando START tiene por valor tres, ejecutar una simulación en forma interactiva, ¿en qué momento la simulación se da por finalizada?, ¿qué bloque TERMINATE dará fin a la simulación, el primero o el segundo?

29.Hacer los siguientes programas:

*A GENERATE 100,,25 TERMINATE 1	*B GENERATE 25,,25 TERMINATE 1
*C GENERATE 100,50,25 ADVANCE 50 TERMINATE 1	*D GENERATE 100,,0 ADVANCE 100 TERMINATE 1

El operando de A del comando START tiene por valor 3, ¿cuál es el valor del operando B, para los ejemplos A, B y D?, ¿en qué momento saldrán las transacciones de los programas anteriores? y ¿en qué momento la simulaciones darán fin?

30.Diseñar los siguiente modelos:

*A GENERATE 100,,,1 ADVANCE 700 TERMINATE 1 GENERATE 250 TERMINATE 1	*B GENERATE 25,,5,2 TERMINATE 1 GENERATE 20 TERMINATE 1

Utilizar el valor de 5 para el operando A del comando START. ¿En qué momentos saldrán transacciones de los bloques GENERATE, en los dos ejercicios? y ¿cuándo la simulaciones darán fin?

31.Generar los siguiente modelos:

*A GENERATE 50,25,-10 TERMINATE 1	*B GENERATE 100,,,4 TERMINATE 1

Para el operando A usar el valor de cinco, ¿cuál es el problema en cada modelo y qué pasará con la simulación de cada modelo?

32.Considerar el siguiente modelo:

	GENERATE 10,,,4
SIGUE	ADVANCE 40,10
	TRANSFER ,SIGUE
	GENERATE 200
	TERMINATE 1

Crear un programa, y al comando START en el operando A usar el valor de uno, ¿en qué momentos las cuatro transacciones del primer bloque GENERATE avanzarán al bloque

ADVANCE? y ¿saldrán del bloque ADVANCE las cuatro transacciones en el orden como entraron?

33.Diseñar el siguiente modelo:

```
GENERATE 50,,,3
TERMINATE 1
GENERATE 25,,,7
TERMINATE 1
```

Utilizar el valor seis en el operando A del comando START, ¿en qué momentos las transacciones saldrán de los dos bloque GENERATE y cuáles serán sus identificadores? y ¿en qué momento la simulación dará fin?

34.Considerar el siguiente modelo:

```
*A                          *B
      ADVANCE 25,5                GENERATE 60,20
      GENERATE 60,20              TERMINATE 1
      TERMINATE 1                 ADVANCE 25,5
```

Ninguno de los dos modelos es lógico, ¿Por qué? y ¿qué ocurrirá al ejecutar los modelos?

35.Examinar el siguiente modelo:

```
      SIMULATE
      GENERATE 75,40
      TRANSFER ,ZAPA
      GENERATE 35,5
      ADVANCE 10
      TERMINATE 1
ZAPA  START 5
      END
```

¿En qué sentido el modelo no tiene lógica? y ¿qué pasará al tratar de ejecutar el modelo en GPSS/H?

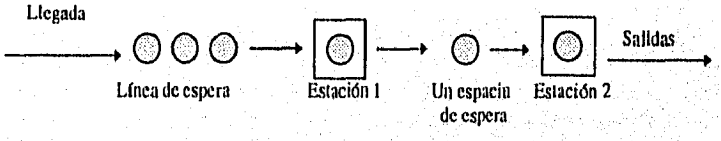
36.¿Cuándo finalizará la simulación del siguiente modelo?

```
      SIMULATE
      GENERATE 25,,0
      SEIZE LLAVE
      ADVANCE 25
      RELEASE LLAVE
      TERMINATE 0
      GENERATE 10
      SEIZE LLAVE
      RELEASE LLAVE
      TERMINATE 1
      START 3
      END
```

37.Juan es dueño de una estética, donde se corta el cabello y se corta la barba. Dos tipos de clientes llegan a la sala; los que vienen a cortarse el cabello solamente, llegan cada 35±10 minutos; y los que llegan a cortarse el cabello y a cortarse la barba llegan cada 60±20 minutos. A Juan le toma 18±6 minutos para cortar el cabello y 10±12 minutos para cortar barba, atiende a los clientes conforme van llegando, la sala abre de las 9:00 a.m. y cierra a las 5 p.m., generalmente Juan come cuando no hay clientes.

Usar un minuto como tiempo base, modelar la estética, diseñar el modelo para simular un día de trabajo, ejecutar el modelo en modo *batch*, ¿al final del modelo Juan se encuentra atendiendo a un cliente?, si así es, ¿qué esta haciendo, cortando el cabello o la barba? ¿Cuántos clientes lo están esperando?

38.Un sistema de producción consiste de dos estaciones de trabajo consecutivas, estación 1 y estación 2, en cada división una persona es la encargada. El trabajo que llega, primero pasa a la estación 1 y luego a la estación 2 y luego sale del sistema. Existe el suficiente espacio de espera para que el trabajo que llega a la estación 1 permanezca ahí hasta que llegue su turno. Sin embargo, el espacio de espera entre la estación 1 y la estación 2, solo puede contener una sola pieza de trabajo. Esta situación se muestra en el siguiente esquema:



La estación 1 tiene que poner la pieza terminada en el espacio de espera entre la estación 1 y 2 antes de comenzar con una nueva. Esto forza algunas veces a la estación 1 a estar libre de trabajo, aunque tenga trabajo. Las piezas llegan a la estación 1 cada 30±20 minutos, la estación 1 tarda con cada pieza 25±10 minutos y la estación 2 tarda 30±10 minutos. Los tiempos requeridos para trasladar de la estación 1 a la 2 se suponen lo suficientemente pequeños. Diseñar el modelo de este sistema con una simulación de 100 piezas de trabajo, ¿en qué momentos la simulación finaliza?

39.En una universidad, los alumnos pasan por los siguiente pasos para registrar sus materias para el siguiente semestre:

- Completar cierta documentación inicial les toma 20±10 minutos, sin tomar el tiempo de espera.
- Van a la ventanilla uno, y una trabajadora revisa sus papeles, le toma aproximadamente 7.5±2.5 minutos, sin tomar el tiempo de espera.
- Van a la ventanilla dos, los alumnos le pagan a una recepcionista que pone un sello, esto les lleva como 10±5 minutos, sin contar el tiempo de espera.
- Regresan a la ventanilla uno para finalizar la inscripción y esto les lleva como 5±2 minutos sin el tiempo de espera.
- Después de esto el estudiante se va.

El sistema opera desde las 9 a.m. hasta las 4:30 p.m. Los estudiantes llegan cada 12.5±10 minutos independientemente del momento del día. Crear un modelo para este sistema, usar un minuto como base de tiempo y simular un día, analizar y discutir los resultados. ¿Cuando la simulación finaliza cuántos estudiantes hay en el sistema?

40. Los pacientes de un doctor llegan al consultorio cada 15±10.5 minutos. El doctor le toma aproximadamente 14±5 minutos atendendiendo a cada enfermo. Diseñar el modelo de este sistema, recolectando información desde el momento que llegan los pacientes hasta que empiezan a ser atendidos por el doctor, con la entidad queue. Simular la atención de 50 pacientes.

41. Del consultorio anterior, suponer que ahora son tres doctores, los pacientes llegan cada 5±3.5 minutos, y a cada doctor le toma 14±5 minutos atendiendo a cada cliente. Este ejercicio es un sistema de una línea con múltiples servidores, modelar el sistema y recolectar información desde que los pacientes llegan hasta que empiezan a ser atendidos por alguno de los doctores, simular a 50 pacientes. Comparar los dos reportes de los ejercicios anteriores, ¿a la larga el número de pacientes que esperan a en la fila será igual para los dos modelos y porqué? y ¿a la larga el tiempo de espera por paciente de los dos modelos será igual y porqué?

42. En un departamento de bomberos, las llamadas de emergencia llegan cada 30±5 minutos. 12% de las llamadas son falsas, requiriéndose unos 30±15 minutos atendiendo a estas. 20% de las llamadas son muy serias requiriéndose unos 200±60 minutos atendiendolas. El otru 68% son llamadas algo serias (de rutina) y les toma unos 90±30 minutos en atenderlas. En el sistema no se sabe cual categoría de llamada llega primero hasta que el camión de bomberos responde a la llamada y llega al lugar. Si se supone que nunca hay falta de camiones, crear el modelo anterior. Utilizar .120 y .200 para el operando A de los bloques condicionales TRANSFER. Simular con el modelo 500 llamadas atendidas. Determinar que porcentaje de llamadas fueron falsas, muy serias y de rutina. Comparar estos porcentajes con los datos dados en la descripción del sistema. ¿Cuántas llamadas llegaron al final de la simulación? ¿En promedio cuántos camiones fueron necesarios? ¿Cuál fue el máximo número de camiones en uso?

43. Del ejemplo anterior suponer que sólo hay cinco camiones disponibles. Si todos los camiones están ocupados al llegar una llamada de emergencia, debe esperar su turno (PLPS). Modificar el modelo del ejercicio anterior para que corresponda a este sistema. Coleccionar cuatro conjuntos de información queue para llamadas que esperan ser atendidas: un conjunto para las alarmas falsas, uno para las llamadas graves, uno para las de rutina y uno para los tres tipos de llamadas. Hacer una simulación hasta que 500 llamadas sean atendidas. ¿Cuántas llamadas, a lo más esperaron en un solo instante para un camión? ¿Cuál fue el tiempo promedio de utilización de los camiones? ¿Qué porcentaje de los tres tipos de llamadas no tuvo que esperar para ser atendido?

44. Del modelo anterior, suponer ahora que sólo existen cuatro camiones. Y en otro modelo suponer que sólo hay seis camiones. Analizar las diferencias entre los resultados.

45. La media del tiempo entre llegadas de trabajos a una central es de 10±5 minutos. Treinta porciento de los trabajos son de tipo A y el resto es de tipo B. Los dos tipos de trabajo llegan a la estación 1 donde se toma 20±10 minutos de tiempo de servicio. Luego los trabajos de tipo A van a la estación 2 donde se toma 30±10 minutos de atención. Después pasan a la estación 4 para una atención final que requiere de 25±10 minutos.

Los trabajos de tipo B de la estación 1 pasan a la estación 3 para llevar a cabo este paso les toma 45±5 minutos de tiempo de servicio. Luego pasan a la estación 4 para una atención final que requiere el mismo tiempo de proceso que del trabajo A.

a) Diseñar un modelo para el sistema anterior, asumiendo que el orden de servicio de PLPS (primero en llegar primero en salir) y en todo el sistema recolectar información queue de los tiempos de espera de los trabajos de cada estación. Asumir que en cada estación hay un limite de trabajadores atendiendo los trabajos. Simular el modelo con 100 trabajos que hayan dejado el sistema. ¿Cuál es el promedio de trabajadores en cada estación?

b)Ahora redondear el promedio de número de trabajadores en uso en la estación 1 al siguiente entero y cambiar el número de servidores. También hacer lo anterior para la estación 2, 3 y 4 respectivamente. Repitir la simulación y ahora comparar los dos resultados de las dos simulaciones. ¿Cuántos trabajos había esperando en promedio en cada estación y como máximo cuánto esperaron en cada estación?

46. Discutir qué es lo que ocurre con los siguiente ejemplos:

*A JUAN TRANSFER .375,JUAN, PEDRO	*B OTRA TRANSFER .375,,OTRA
*C SINFI TRANSFER .375,SINFI,SINFI	*D TRANSFER .0, CA1, CA2

En el ejemplo C, si una transacción llega a este bloque, ¿qué hará parar a esta simulación?

47. En el centro de cómputo de la escuela existen varias computadoras que son utilizadas por los estudiantes con base en que el primero en llegar será el primero en ser atendido. Los estudiantes llegan cada 10±5 minutos. El uso de las computadoras por estudiante es de 90±30 minutos en promedio. Si todas las computadoras están en uso, dos de tres estudiantes esperarán. Los estudiantes que se van, regresarán 60±30 minutos después para intentar entrar otra vez. Los que intentan entrar otra vez tienen que hacer trabajos finales y no tienen otra opción que esperar, si es necesario, en su segundo intento.

a) Diseñar el modelo de este sistema. En el bloque condicional TRANSFER, usar como valor del operando A .333. Asumir que no existe límite al número de computadoras, esto significa que ningún estudiante se podrá ir sin hacer uso de una computadora, sin

embargo hacer la opción de que los estudiantes puedan irse sin usar la computadora. Simular el modelo con 250 estudiantes. ¿Cuántas computadoras serán usadas en promedio? ¿Cuántas como máximo serán usadas en un momento dado?

b) Ahora repitir el modelo anterior, el número de computadoras estará restringido al valor entero mayor que al promedio en uso de las computadoras en el ejercicio anterior (ejemplo: si el valor fue de 5.3 entonces será 6). ¿Qué porcentaje de estudiantes no pudieron usar la computadora la primera vez que intentaron? ¿Qué porcentaje se fue y regresó después?

c) Volver hacer el modelo a), pero ahora restringiendo el número de computadoras a un nivel mayor que el del ejemplo b) (ejemplo: si en el ejemplo fue 6 en este caso será 7). ¿Qué porcentaje de estudiantes no pudieron entrar al centro de cómputo la primera vez que intentaron? ¿Qué porcentaje se fue y regresó después?

d) Modificar el modelo del ejercicio a) utilizando dos niveles mayores que el ejercicio b) (si en el ejercicio b fueron 6 en este caso serán 8). ¿Qué porcentaje de estudiantes no obtuvieron una máquina la primera vez que intentaron? ¿Qué porcentaje se fue y después regresó?

48. En un banco hay dos cajas abiertas, los clientes llegan cada 1.5±1.0 minutos. El tiempo de servicio de una de las cajeras es de 2.5±1.5 minutos pues es una cajera con experiencia. La otra cajera es nueva y se tarda por cliente en promedio 3.25±2.5 minutos. Una sola línea y con varias servidoras se usa en este sistema.

Suponer que, si ambas cajeras están libres cuando un cliente llega, escogen al azar la cajera. Diseñar un modelo de esta situación, de forma que la información que queue coleccione sea el tiempo que se tarden en ser atendidos. Simular el modelo para 100 clientes que sean atendidos. ¿Cuántos clientes no tuvieron que esperar para ser atendidos? ¿Existe alguna forma de saber cuantos clientes llegaron cuando las dos cajeras estaban libres (idles) y de éstos, cuántos prefirieron a la cajera con experiencia?

49. Los enfermos llegan a la sala de emergencia de un hospital cada 30±30 minutos. Primero son atendidos por una enfermera, quien los clasifica de la siguiente manera: 45% de ellos pueden esperar y el otro 55% necesitan atención urgente. Sólo hay una enfermera que los puede clasificar y se lleva a cabo la clasificación en 10±5 minutos.

Dos doctores trabajan en esta sala, los pacientes de atención urgente tienen mayor prioridad que los que pueden esperar. Un enfermo con alta prioridad se tarda con el doctor aproximadamente 45±30 minutos. El 75% de estos pacientes después tiene que esperar a unos exámenes, mientras el 25% restante se van. Cuando hay la necesidad de hacer unos exámenes les toma 30±10 minutos más.

Después de estos exámenes, los pacientes de atención urgente tienen que ver al doctor nuevamente, pero ahora tienen una prioridad menor que los pacientes que pueden esperar

y que no han visto al doctor. Ellos reciben la atención del doctor y se tardan 15±5 minutos con el doctor y se van a casa. Se debe hace notar que los pacientes de atención urgente potencialmente pueden esperar a los doctores dos veces, y los enfermos que pueden esperar, sólo ven al doctor una vez. Como se explicó, su prioridad es menor que los que necesitan atención urgente y que no han visto al doctor, pero tienen mayor prioridad que los que necesitan atención pero que van a ser examinados por segunda ocasión. La atención a los enfermos que pueden esperar es de 30±15 minutos en promedio.

Diseñar el modelo de esta situación para obtener información de los cuatro puntos potenciales de espera de los enfermos en el sistema. Simular en el modelo diez periodos de 24 horas, suponer que no existen cambios en un periodo de los tiempos de llegadas, el patrón de clasificación, el número de doctores, etc. Discutir los resultados de esta simulación

50. Los siguientes incisos se resuelven utilizando la función RVEXPO:

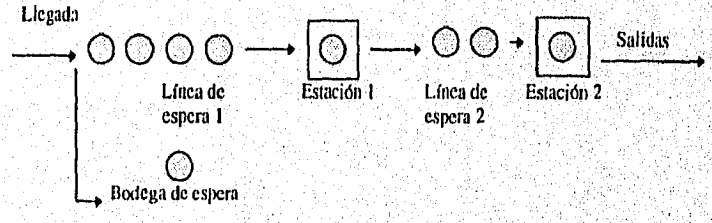
a) Generar un modelo que contenga un bloque GENERATE que introduzca transacciones al modelo de forma exponencial, si la media de tiempo de llegada es 15 llegadas por hora. Suponer que la base de unidad de tiempo es un segundo, utilizar el generador RN5 para obtener la muestra.

b) Repetir lo anterior, pero ahora inferir la base de unidad de tiempo como 0.5 minutos y utilizar el generador RN1.

c) Un proceso de servicio se supone de forma exponencial con media de 36 minutos. Diseñar un modelo que contenga el bloque ADVANCE que puede ser usado como tiempo de servicio en un modelo que tiene como base de unidad de tiempo 1 hora con el generador RN2.

d) Repetir el modelo anterior pero ahora asumir que el tiempo de servicio se distribuye como una Erlang de orden 2.

51. La siguiente figura muestra el esquema de un sistema de mantenimiento.



El sistema consiste de dos estaciones de trabajo en serie, las estaciones sólo pueden trabajar por unidad, por el límite de espacio de espera, el máximo de unidades que pueden

esperar en el sistema es de 6. Contando las unidades en las estaciones la capacidad del sistema es de 8 unidades. En el esquema, cuatro espacios de espera están asignados al la estación 1 y dos a la estación 2. Cuando ya no hay espacio las nuevas llegadas son enviadas a la bodega de espera.

Una unidad terminada de la estación 1 no puede dejar la estación hasta que exista espacio para avanzar a la estación 2, mientras la estación 1 esta bloqueada no puede trabajar en otra unidad.

Las llegadas se distribuyen exponencialmente con una media de 0.4 por unidad de tiempo. Los tiempos de servicio de las dos estaciones también se distribuyen exponencialmente con media 0.25 unidades de tiempo para la estación 1 y 0.50 unidades de tiempo para 2. Los tiempos de viaje son lo suficientemente pequeños.

Modelar el sistema, calculando la utilización de las estaciones, del espacio de espera, tiempo en el sistema y la fracción del tiempo que la estación 1 estuvo bloqueada y el número de unidades que fueron mandadas a la bodega de espera. Simular 300 unidades de tiempo y analizar el reporte de salida.

52.Generar un modelo que contenga el bloque GENERATE que introduzca a las transacciones a un modelo de fonna triangular con un tiempo de llegada menor de 10, con media 18 y con un máximo de 30. Asumir su unidad de tiempo de 1 minuto y utilizar el generador RN7.

53.Repetir el modelo anterior pero ahora suponiendo que los tiempos entre llegadas son triangulares y simétricos con un máximo y mínimo correspondiente al del ejercicio anterior.

54.Diseñar un modelo con el bloque ADVANCE que mantenga de 5 a 50 minutos un 99.7 porciento del tiempo y que este distribuida normalmente. Asumir que el tiempo base es de un minuto y que el generador de muestra es el RN2.

55.Repetir el modelo anterior pero suponer que se mantiene de 5 a 50 minutos con un 98% de tiempo.

56.Los siguientes ejercicios se basan en funciones discretas y de acuerdo a la información de la siguiente tabla. La tabla resume el tiempo necesario para pagar ciertos utensilios en una caja de una tienda, el tiempo depende del número de aparatos a comprar.

Número de Utensilios a Comprar	Tiempo de Servicio (minutos)	Frecuencia Relativa
1	0.5	0.10
2	0.75	0.15
3	0.95	0.20
4	1.10	0.25
5	1.25	0.15
6	1.40	0.10
7	1.50	0.05

a) Definir la función discreta para sacar una muestra del tiempo de servicio, utilizar el generador RN5 para sacar la muestra, no cambiar el orden de la información y suponer el tiempo base en un minuto.

b) Utilizando la definición anterior mostrar un bloque ADVANCE que pueda ser usado para simular el tiempo de servicio.

c) Gráficar la función discreta de ejercicio a).

d)Asumir que la función discreta es evaluada y que el generador RN5 produce un valor de 0.35, ¿qué valor producirá la función del ejercicio a)?

e) Repetir el problema d), pero ahora suponer que el generador produce un valor de 0.25.

f) ¿De la tabla, cuál es el tiempo esperado de servicio?

g) ¿Cómo cambiará los resultados si se supone que el tiempo base es 0.5 minutos?

h) Acomodar la información para que el orden de la frecuencia relativa disminuya, y repetir los ejercicios del a) al e).

57.En los siguiente ejercicios se usan funciones continuas y se basan en la siguiente tabla. La tabla resume los tiempos entre llegadas de aviones al espacio de un aeropuerto, los tiempos entre llegadas en los intervalos son los más frecuentes.

Rango de Tiempo entre llegadas (minutos)	Frecuencia Relativa
≤0	0.0
>0-≤1	0.10
>1-≤2	0.15
>2-≤3	0.20
>3-≤4	0.25
>4-≤5	0.20
>5≤6	0.10
≥6	0.0

a) Definir una función continua que puede ser usada para sacar una muestra de los tiempos entre llegadas de los aviones al espacio aéreo, utilizar el generador RN3 como base para obtener la muestra. Asumir un tiempo base de un minuto.

b) Utilizando la función de a) generar un bloque GENERATE que introduzca las transacciones de los aviones al modelo.

c) Gráficar la función del ejercicio a).

d) Suponer que la función del ejercicio a) es evaluado, si el generador RN3 produce un valor de 0.40, ¿qué valor producirá la función?

e) Repetir el ejercicio anterior, suponiendo que el generador RN5 produce un valor .90.

f) Gráficar un histograma de los tiempos entre llegadas de la tabla.

g) De la información de la tabla, ¿cuál es el valor de espera de los tiempos entre llegadas?

h) ¿Cómo cambiarían los resultados si se asume que el tiempo base es de 0.5?

58. En los siguientes ejercicios se usa la definición de funciones continuas de dos puntos para muestras uniformes:

a) Obtener la muestra de la distribución de tiempos entre llegadas en el bloque GENERATE 25,10, utilizar el generador RN1. Si se desearа utilizar otro generador como el RN2 como base para la muestra, ¿cómo se haría lo anterior, demostrar que es posible puede utilizar la función del ejercicio a)?

b) Mostrar una gráfica de la función del ejercicio a).

c) ¿Si la función del ejercicio a) evaluada y el generador RN2 produce un valor de 0.45, qué valor produciría la función?

d) Repetir los pasos del a) al c) con el uso del bloque ADVANCE 50,50.

59. Considerar el siguiente bloque:

```
GENERATE RVTRI(3,4,0,6,0,9,0),2
```

a) ¿Cuántos números aleatorios son utilizados para generar el tiempo entre llegadas en este bloque?

b) ¿En qué intervalo se encuentran los valores de este bloque?

60. Considerar el siguiente bloque, donde FACTOR es una función C2 definida con los pares ordenados 0,3 y1,6 y tiene el generador RN7 como argumento:

```
ADVANCE 5,FN(FACTOR)
```

a) ¿En qué intervalo los tiempos de retención están distribuidos? ¿Qué tipo de distribución es seguida de los tiempos de retención?

b) ¿Qué patrón de tiempo de retención tomará si se cambia C2 pro D2 en la definición de la función FACTOR?

61. Utilizando alguno de los modelos en los ejercicio vistos hasta ahora, generar varias réplicas utilizando las diferentes formas vistas en el trabajo.

62.Los ejercicios que se plantean a continuación se basan en la siguiente información, la cual se basa en un ejemplo de número de mecánicos que esperan una orden de trabajo de dos alternativas de servicio: PLPA (primero en llegar primero en ser atendido) y TMP (tiempo menor de procesamiento), las simulaciones fueron hechas en GPSS/H:

Réplica	PLPA	TPM
1	1.705	1.501
2	3.627	2.033
3	2.780	2.456
4	5.502	7.584
5	3.494	3.060
6	4.114	0.967
7	4.704	5.536
8	4.326	2.489
9	1.362	4.163
10	3.737	2.890
Media de la muestra	3.535	3.268
Desviación estándar de la muestra	1.286	1.999

a) Determinar un intervalo de confianza con un nivel de 90% para la muestra del PLPA.

b) Determinar con un nivel de confianza del 80% y 95% los intervalos para la muestra del PLPA. ¿Qué pasará con respecto al resultado del ejercicio a)?

c) ¿Qué se espera que suceda al calcular con un nivel de confianza 90% el intervalo de la muestra TPM al compararlo con el resultado del ejercicio a)? ¿Será más angosto o más ancho y por qué?

63.De la siguiente información calcular el intervalo de confianza con un nivel de 95%, se basa en el tiempo requerido para que 100 aspiradoras pasen la inspección final, las simulaciones fueron hechas en GPSS/H:

Réplica	Minutos Requeridos
1	586.1
2	607.3
3	588.0
4	549.6
5	590.7
Media de la muestra	584.3
Desviación estándar de la muestra	21.2

64.Si se supone que en el ejercicio anterior no son cinco sino 20 réplicas, con la misma media y desviación estándar, ¿Cómo sería el intervalo de confianza con 95% de la variable de salida, más ancha o angosta con respecto al ejercicio anterior y por qué? Ahora suponer que son 80 réplicas, ¿Qué pasaría?

65.Tan solo un coche puede ser lavado a la vez en un lugar de lava coches. Los autos llegan de forma Poisson con una media de tiempo entre llegadas de 5 minutos. El lavado del

coche se lleva a cabo de forma exponencial con media de 4 minutos. Los clientes potenciales que no encuentran espacio de espera llevan sus automóviles a otros lugares.

Modelar el sistema, úsarlo para simular tres alternativas: un espacio de espera, dos espacios de espera y tres espacios. De cada alternativa diseña 20 réplicas, basando cada réplica en ocho horas de trabajo continuo. Asumir que la fila esta vacía y no hay coches lavándose al principio de cada réplica. Para cada alternativa determinar un nivel de confianza con un 90% de confianza para la fracción de los coches que llegan y esperan a ser atendidos. Comparar resultados.

66. A una persona se le ha asignado el mantenimiento de tres máquinas. Cada una tiene un media de tiempo duración hasta que llega la primera falla de nueve horas, distribuida exponencialmente. El tiempo de reparación está distribuido de forma también exponencial y tiene una media de dos horas.

Diseñar el modelo de la situación anterior, y utilizarlo para estimar el valor esperado de que no están descompuestas, hacer 20 réplicas basándose cada una en 250 horas de operación. Asumir que cada réplica comienza con una máquina descompuesta, otra tiene 10 horas antes de no servir y la tercera tiene 3 horas antes de no correr bien. Utilizar los resultados para determinar un intervalo de confianza del 95% del número de maquinas que sirven. ¿El intervalo contiene al valor esperado de 2.28?

67. La siguiente información se basa en un ejemplo del número de mecánicos que esperan una orden de trabajo de dos alternativas de servicio: PLPA y TMP:

Réplica	PLPA	TPM	Diferencia
1	3.826	1.326	2.500
2	1.529	3.668	-2.139
3	1.805	0.737	1.068
4	4.140	1.619	2.521
5	2.116	3.572	-1.456
6	5.010	1.290	3.720
7	5.411	2.723	2.688
8	1.974	2.229	-0.255
9	6.585	1.395	5.190
10	2.550	1.346	1.204
Media de la muestra			1.504
Desviación estándar de la muestra			2.293
80% Intervalo de confianza			[0.50,2.50]
90% Intervalo de confianza			[0.18,2.83]
95% Intervalo de confianza			[-0.14,3.14]

a) Utilizar la aproximación de pares no correlacionados con el estadístico *t* vista en el trabajo de tesis para computarizar el intervalo de confianza del 90% de las diferencias. ¿Los puntos mayores y menores son iguales a los de la tabla anterior?

b) Determinar los intervalos Welch con 80%, 90% y 95% de las diferencias de la tabla anterior. Es importante recordar que para estos intervalos no se requiere que las varianzas sean iguales, comparar los resultados con los obtenidos en la tabla. ¿El intervalo Welch de 95% contiene al cero?

BIBLIOGRAFÍA

- Banks Jerry, Carson S. John
  Discrete-Event System Simulation
  Ed. Prentice Hall
  Estados Unidos, 1984.

- Canavos C. George
  Probabilidad y Estadística, Aplicaciones y Métodos
  Ed. Mc. Graw Hill
  México, 1988.

- Coss Bu Raúl
  Simulación, Un Enfoque Práctico
  Ed. Limusa
  México, 1982.

- Estrine Elliott
  Directory of Simulation Software
  Ed. Simulation Councils
  Estados Unidos, 1991.

- Fishman, G.
  Conceptos y Métodos en la Simulación Digital de Eventos Discretos
  Ed. Limusa
  México, 1982.

- González Videgaray, MariCarmen
  Modelos y Simulación; Un Enfoque Computacional con Aplicaciones Actuariales y de Optimización
  Ed. Exa Ingeniería
  México, 1993.

- Gordon, G.
  Simulación de Sistemas
  Ed. Diana
  México, 1980.

- Hillier, Frederick S. y Lieberman, Gerald J.
  Introducción a la Investigación de Operaciones
  Ed. Mc Graw Hill
  Tercera Edición, México, 1991.

- Hoover, Stewart V., Perry, Ronald F.
  Simulation a Problem Solving Approach
  Ed. Addison Wesley
  Estados Unidos de Norte America, Agosto 1990.

- Law Averill M. y Kelton W. David
  Simulation Modeling and Analysis
  Ed. Mc Graw-Hill International Editions
  Segunda Edición, Singapore, 1991.

- Macaluso Pat
  Learning Simulation Techniques on a Microcomputer
  Ed. Tab Books
  U.S.A., 1983.

- Meier Newell
  Técnicas de Simulación en Administración y Economía
  Ed. Trillas
  México, 1975.

- Naylor, Thomas H.
  Técnicas de Simulación en Computadoras
  Ed. Limusa
  México, 1985.

- Payne J.A.
  Introduction to Simulation
  Ed. Mc Graw-Hill
  U.S.A., 1982.

- Raczynski Stanislaw
  Simulación por Computadora
  Ed. Limusa
  México, 1993.

- Schriber, Thomas J.
  An Introduction to Simulation Using GPSS/H
  Ed. John Wiley and Sons
  Estados Unidos de Norte America, 1991.

- Taha A. Hamdy
  Operation Research, An Introdution
  Cuarta Edición, Ed. Mac Millan Publishing
  London, Britain 1987.

- Watson & Blackstone
  Computer Simulation
  Ed. Wiley & Sons
  Estados Unidos, 1989.

- Wolverine Software
  Using Proof Animation
  Ed. Wolverine Software Corporation
  Estados Unidos, 1992.