UDS

Mi Universidad

LIBRO

Simulación

Ingeniería en Sistemas Computacionales

Octavo Cuatrimestre

Enero - Abril

Marco Estratégico de Referencia

Antecedentes históricos

Nuestra Universidad tiene sus antecedentes de formación en el año de 1979 con el inicio de actividades de la normal de educadoras "Edgar Robledo Santiago", que en su momento marcó un nuevo rumbo para la educación de Comitán y del estado de Chiapas. Nuestra escuela fue fundada por el Profesor Manuel Albores Salazar con la idea de traer educación a Comitán, ya que esto representaba una forma de apoyar a muchas familias de la región para que siguieran estudiando.

En el año 1984 inicia actividades el CBTiS Moctezuma Ilhuicamina, que fue el primer bachillerato tecnológico particular del estado de Chiapas, manteniendo con esto la visión en grande de traer educación a nuestro municipio, esta institución fue creada para que la gente que trabajaba por la mañana tuviera la opción de estudiar por las tardes.

La Maestra Martha Ruth Alcázar Mellanes es la madre de los tres integrantes de la familia Albores Alcázar que se fueron integrando poco a poco a la escuela formada por su padre, el Profesor Manuel Albores Salazar; Víctor Manuel Albores Alcázar en julio de 1996 como chofer de transporte escolar, Karla Fabiola Albores Alcázar se integró en la docencia en 1998, Martha Patricia Albores Alcázar en el departamento de cobranza en 1999.

En el año 2002, Víctor Manuel Albores Alcázar formó el Grupo Educativo Albores Alcázar S.C. para darle un nuevo rumbo y sentido empresarial al negocio familiar y en el año 2004 funda la Universidad Del Sureste.

La formación de nuestra Universidad se da principalmente porque en Comitán y en toda la región no existía una verdadera oferta Educativa, por lo que se veía urgente la creación de una institución de Educación superior, pero que estuviera a la altura de las exigencias de los

jóvenes que tenían intención de seguir estudiando o de los profesionistas para seguir preparándose a través de estudios de posgrado.

Nuestra Universidad inició sus actividades el 18 de agosto del 2004 en las instalaciones de la 4ª avenida oriente sur no. 24, con la licenciatura en Puericultura, contando con dos grupos de cuarenta alumnos cada uno. En el año 2005 nos trasladamos a nuestras propias instalaciones en la carretera Comitán – Tzimol km. 57 donde actualmente se encuentra el campus Comitán y el corporativo UDS, este último, es el encargado de estandarizar y controlar todos los procesos operativos y educativos de los diferentes campus, así como de crear los diferentes planes estratégicos de expansión de la marca.

Misión

Satisfacer la necesidad de Educación que promueva el espíritu emprendedor, aplicando altos estándares de calidad académica, que propicien el desarrollo de nuestros alumnos, Profesores, colaboradores y la sociedad, a través de la incorporación de tecnologías en el proceso de enseñanza-aprendizaje.

Visión

Ser la mejor oferta académica en cada región de influencia, y a través de nuestra plataforma virtual tener una cobertura global, con un crecimiento sostenible y las ofertas académicas innovadoras con pertinencia para la sociedad.

Valores

- Disciplina
- Honestidad
- Equidad
- Libertad

Escudo

El escudo del Grupo Educativo Albores Alcázar S.C. está constituido por tres líneas curvas que nacen de izquierda a derecha formando los escalones al éxito. En la parte superior está situado un cuadro motivo de la abstracción de la forma de un libro abierto.

Eslogan

"Mi Universidad"

ALBORES

Es nuestra mascota, un Jaguar. Su piel es negra y se distingue por ser líder, trabaja en equipo y obtiene lo que desea. El ímpetu, extremo valor y fortaleza son los rasgos que distinguen.

Simulación

Objetivo de la materia:

Construir un modelo de simulación de un sistema real, para diseñar y evaluar su funcionamiento.

.

Criterios de evaluación:

No	Concepto	Porcentaje
1	Trabajos Escritos	10%
2	Actividades Áulicas	20%
3	Trabajos en plataforma Educativa	20%
4	Examen	50%
Total de Criterios de evaluación		100%

INDICE

UNIDAD I

INTRODUCCIÓN

1.1.- Definición.

1.2.- Conceptos.

1.3.- Tipos de simulación

1.4.- Modelos.

1.5.- Metodología.

UNIDAD II

NÚMEROS PSEUDOALEATORIOS

2.1.- Generación de números pseudoaleatorios.

2.2.- Pruebas estadísticas de aleatoriedad.

2.3.- Generación de variables aleatorias.

2.3.1.- Variables aleatorias discretas.

2.3.2.- Variables aleatorias continúas.

2.3.3.- Pruebas T.

UNIDAD III

MÉTODO DE MONTECARLO

3.1.- Lenguaje de Simulación.

3.1.1.- Introducción.

3.1.2.- Lenguajes de propósito general.

3.1.3.- Lenguajes de propósito especial.

UNIDAD IV

SIMULACIÓN POR COMPUTADORA

4.1.- Con lenguajes de propósito general.

4.2.- Con lenguajes de propósito especial.

4.3.- Sobre sistemas de inventarios, teoría de colas, de producción, etc.

4.4.- Validación.

UNIDAD I

Introducción

El presente documento trata sobre las técnicas utilizadas para imitar o simular el funcionamiento de distintos tipos de instalaciones o procesos. A la instalación o proceso que se pretende estudiar se le denomina sistema y para poderlo analizar se realiza una serie de supuestos sobre su funcionamiento. Estos supuestos, que normalmente se expresan mediante relaciones matemáticas o relaciones lógicas, constituyen un modelo del sistema. Este modelo se utiliza para comprender y prever el comportamiento del sistema real. Si las relaciones matemáticas o lógicas que comprende el modelo son sencillas, entonces será posible utilizar un procedimiento analítico para obtener una solución o respuesta exacta sobre las características de interés del sistema analizado. No obstante, si las relaciones son complejas, puede ocurrir que no se pueda evaluar analíticamente el problema. En este caso, será necesario acudir a la simulación del sistema, evaluando numéricamente el modelo y analizando los datos obtenidos para estimar las características de dicho sistema.

1.1. Definición.

Podríamos decir que simular tiene como objetivo duplicar características y comportamientos propios de un sistema real. Simularemos problemas relacionados con la Organización Industrial a través de la construcción de modelos matemáticos que representen de forma fidedigna la realidad. La utilización de modelos matemáticos permite: Introducir nuevas variables. Hacer variar sus valores. Analizar las consecuencias de estas modificaciones.

1.2. Conceptos.

Modelación.

Es aquello que sirve para representar o describir otra cosa es decir crea prototipos (1° diseño), el modelo puede tener una forma semejante o ser totalmente distinto del objeto real.

Modelo.

Se puede definir como una representación simplificada de un sistema real, un proceso o una teoría, con el que se pretende aumentar su comprensión hacer predicciones y posiblemente ayudar a controlar el sistema.

Existen 3 formas de modelos:

Icónico: Versión a escala del objeto real y con sus propiedades más o menos relevantes.

Analógico: Modelo con apariencia física distinto al original, pero con comportamiento representativo.

Analítico: Relaciones matemáticas o lógicas que representan leyes físicas que se cree gobiernan el comportamiento de la situación bajo investigación. Su utilidad puede tener las siguientes matrices: Ayuda para aclarar el pensamiento acerca de un área de interés.

Como una ilustración de concepto. Como una ayuda para definir estructura y lógica Como un prerrequisito al diseño.

La actividad de diseñar está interesada en definir cómo lograr un determinado propósito. Sin embargo, previamente al diseño esta la etapa de decidir que se va a diseñar. La modelación conceptual es necesaria en esta etapa.

1.3. Tipos de Simulación.

De acuerdo a la naturaleza del modelo empleado, la simulación puede ser por (Fishman, 1978): · **Identidad:** Es cuando el modelo es una réplica exacta del sistema en estudio. Es la que utilizan las empresas automotrices cuando realizan ensayos de choques de automóviles utilizando unidades reales.

· **Cuasi-identidad:** Se utiliza una versión ligeramente simplificada del sistema real. Por ejemplo, los entrenamientos militares que incluyen movilización de equipos y tropas pero no se lleva a cabo una batalla real.

· **Laboratorio:** Se utilizan modelos bajo las condiciones controladas de un laboratorio. Se pueden distinguir dos tipos de simulaciones:

o Juego operacional: Personas compiten entre ellas, ellas forman parte del modelo, la otra parte consiste en computadoras, maquinaria, etc. Es el caso de una simulación de negocios

donde las computadoras se limitan a recolectar la información generada por cada participante y a presentarla en forma ordenada a cada uno de ellos.

o Hombre-Máquina: Se estudia la relación entre las personas y la máquina. Las personas también forman parte del modelo. La computadora no se limita a recolectar información, sino que también la genera. Un ejemplo de este tipo de simulación es el simulador de vuelo.

· **Simulación por computadora:** El modelo es completamente simbólico y está implementado en un lenguaje computacional. Las personas quedan excluidas del modelo. Un ejemplo es el simulador de un sistema de redes de comunicación donde la conducta de los usuarios está modelada en forma estadística.

Este tipo de simulación a su vez puede ser:

Digital: Cuando se utiliza una computadora digital.

Analógica: Cuando se utiliza una computadora analógica. En este grupo también se pueden incluir las simulaciones que utilizan modelos físicos.

1.4. Modelos.

El concepto de sistema en general está sustentado sobre el hecho de que ningún sistema puede existir aislado completamente y siempre tendrá factores externos que lo rodean y pueden afectarlo.

Los objetivos que se persiguen al estudiar uno o varios fenómenos en función de un sistema son aprender cómo cambian los estados, predecir el cambio y controlarlo, todo sistema consta de 3 características; Tienen fronteras, existe dentro de un medio ambiente y tiene subsistemas, el medio ambiente es el conjunto de circunstancias dentro de las cuales esta una situación problemática, mientras que las fronteras distinguen las entidades dentro de un sistema de las entidades que constituyen su medio ambiente.

Conceptos Básicos de Sistemas.

Entidad: "Una entidad es algo que tiene realidad física u objetiva y distinción de ser o de carácter".

Las entidades tienen ciertas propiedades que los distinguen a unas de otras.

Relación: "Relación es la manera en la cual dos o más entidades dependen entre sí". Relación es la unión que hay entre las propiedades de una o más entidades; por consiguiente, el cambio en alguna propiedad de una entidad ocasiona un cambio en una propiedad de otra entidad.

Estructura: Es un conjunto de relaciones entre las entidades en la que cada entidad tienen una posición, en relación a las otras, dentro del sistema como un todo.

Estado: El estado de un sistema en un momento del tiempo es el conjunto de propiedades relevantes que el sistema tiene en este momento. Cuando se habla del estado de un sistema, entiende los valores de los atributos de sus entidades. Analizar un sistema supone estudiar sus cambios de estado conforme transcurre el tiempo.

Modelación de sistemas.

Puede ser una representación formal de la teoría o una explicación formal de la observación empírica, a menudo es una combinación de ambas. Los propósitos de usar un modelo son los siguientes:

Hace posible que un investigador organice sus conocimientos teóricos y sus observaciones empíricas sobre un sistema y deduzca las consecuencias lógicas de esta organización.

Favorece una mejor comprensión del sistema.

Acelera análisis.

Constituye un sistema de referencia para probar la aceptación de las modificaciones del sistema.

Es más fácil de manipular que el sistema mismo.

Hace posible controlar más fuentes de variación que lo que permitiría el estudio directo de un sistema.

Suele ser menos costoso.

Al analizar un sistema podemos observar, que al cambiar un aspecto del mismo, se producen cambios o alteraciones en otros. Es en estos casos en los que la simulación, representa una buena alternativa para analizar el diseño y operación de complejos procesos o sistemas.

La modelación de sistemas es una metodología aplicada y experimental que pretende:

Describir el comportamiento de sistemas.

Hipótesis que expliquen el comportamiento de situaciones problemáticas.

Predecir un comportamiento futuro, es decir, los efectos que se producirán mediante cambios en el sistema o en su método de operación.

Un modelo se utiliza como ayuda para el pensamiento al organizar y clasificar conceptos confusos e inconsistentes. Al realizar un análisis de sistemas, se crea un modelo del sistema que muestre las entidades, las interrelaciones, etc. La adecuada construcción de un modelo ayuda a organizar, evaluar y examinar la validez de pensamientos.

1.5. Metodología

Definición del sistema.

Para tener una definición exacta del sistema que se desea simular, es necesario hacer primeramente un análisis preliminar de este, con el fin de determinar la interacción con otros sistemas, las restricciones del sistema, las variables que interactúan dentro del sistema y sus interrelaciones, las medidas de efectividad que se van a utilizar para definir y estudiar el sistema y los resultados que se esperan obtener del estudio.

Formulación del modelo.

Una vez definidos con exactitud los resultados que se esperan obtener del estudio, se define y construye el modelo con el cual se obtendrán los resultados deseados. En la formulación del modelo es necesario definir todas las variables que forman parte de él, sus relaciones lógicas y los diagramas de flujo que describan en forma completa el modelo.

Colección de datos.

Es importante que se definan con claridad y exactitud los datos que el modelo va a requerir para producir los resultados deseados.

Implementación del modelo con la computadora.

Con el modelo definido, el siguiente paso es decidir si se utiliza algún lenguaje como el fortran,lisp,etc..., o se utiliza algún paquete como Vensim,Stella e atinó, GPSS,Simula,Simscript,Rockwell Arena, etc..., para procesarlo en la computadora y obtener los resultados deseados.

Validación

A través de esta etapa es posible detallar deficiencias en la formulación del modelo o en los datos alimentados al modelo. Las formas más comunes de validar un modelo son:

- La opinión de expertos sobre los resultados de la simulación.
- La exactitud con que se predicen datos históricos.
- 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 de la persona que hará uso de los resultados que arroje el experimento de simulación.

Experimentación.

Se realiza después de que el modelo haya sido validado, consiste en generar los datos deseados y en realizar un análisis de sensibilidad de los índices requeridos.

Interpretación.

Se interpretan los resultados que arroja la simulación y con base a esto se toma una decisión. Es obvio que los resultados que se obtienen de un estudio de simulación ayudan a soportar decisiones del tipo sema-estructurado.

Documentación.

Dos tipos de documentación son requeridos para hacer un mejor uso del modelo de simulación. La primera se refiere a la documentación del tipo técnico y la segunda se refiere al manual del usuario, con el cual se facilita la interacción y el uso del modelo desarrollado.

UNIDAD 2
NUMEROS PSEUDOALEATORIOS

2.1 Generación de Números pseudoaleatorios.

En los experimentos de simulación es necesario generar valores para las variables aleatorias representadas estas por medio de distribuciones de probabilidad.

Para poder generar entradas estocásticas (probabilisticas) para un modelo de simulación, se debe contar con un generador de números pseudoaleatorios. Con estos y métodos de generación de variables aleatorias, se pueden simular las entradas incontrolables para un modelo de simulación.

Inicialmente los números aleatorios se generaban en forma manual o mecánica utilizando técnicas como ruedas giratorias, lanzamientos de dados, barajas. También existen métodos aritméticos que permiten generan un gran conjunto de números aleatorios, pero el advenimiento de la computadora ha permitido crear generadores que permitan generar de manera sucesiva todo los números aleatorios que se requieran.

Un número pseudoaleatorio no es más que el valor de una variable aleatoria x que tiene una distribución de probabilidad uniforme definida en el intervalo (0, 1).

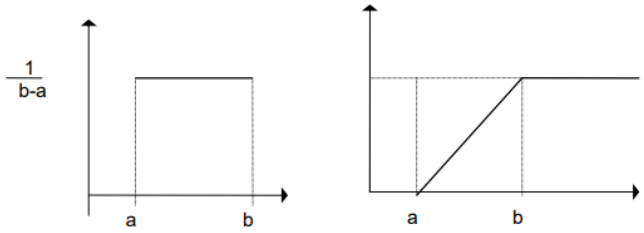
Se sabe que la función de densidad f(x) de una variable aleatoria x con una distribución de probabilidad uniforme en el intervalo [a, b] es:

$$f(x) = \begin{cases} \frac{1}{b-a} & para \quad a \le x \le b \\ 0 & para \quad otros \quad valores \end{cases}$$

La función acumulativa F(x), que representa la probabilidad de que las variable aleatoria x sea menor o igual a un valor específico de x está dada por:

$$F(x) = \begin{Bmatrix} 0 & si & x < a \\ \dfrac{x-a}{b-a} & para & a \le x \le b \\ 1 & si & x > b \end{Bmatrix}$$

La figura 6, muestra la función de densidad y acumulativa para dicha variable aleatoria



El valor esperado y la varianza de una distribución de probabilidad uniforme son respectivamente.

$$\mu = \frac{a+b}{2} \quad , \quad \sigma^2 = \frac{(a+b)^2}{12}$$

Al definir la función de densidad de la distribución de probabilidad uniforme en el intervalo [0, 1], una variable aleatoria R tendría una función de densidad f(R) y una función acumulada F(R), dadas por:

$$f(R) = \begin{Bmatrix} 1 & para & 0 \le R \le 1 \\ 0 & para & otros & valores \end{Bmatrix}$$

$$F(R) = \begin{cases} 0 & si \quad R < a \\ \dfrac{R-a}{b-a} & para \quad 0 \le R \le 1 \\ 1 & si \quad R > b \end{cases}$$

Los valores de la media y la varianza, están dados por:

$$\mu = \frac{1}{2} \quad , \quad \sigma^2 = \frac{1}{12}$$

La variable aleatoria R es continua y debe ser estadísticamente independiente. Finalmente para que para que un conjunto de números sean considerados aleatorios deben cumplir las siguientes características:

- Deben estar uniformemente distribuidos.
- Deben ser estadísticamente independientes.
- Su media debe ser estadísticamente igual a ½.
- Su varianza debe ser estadísticamente igual a 1/12.
- Deben ser reproducibles.

2.2 Pruebas Estadísticas de Aleatoriedad.

La aleatoriedad no es algo que tan solo tenga una aplicación técnica, desde siempre el hombre se ha preguntado si su destino está no o escrito1. Los deterministas2 niegan al hombre el derecho de obrar libremente de acuerdo con su voluntad enfrentados a los no deterministas que opinan justamente lo contrario que el hombre obra libremente y que sus actos no están escritos. A día de hoy, estamos en un estadio en el que todo está escrito, pero en el que el hombre tiene el poder de cambiarlo, (¿por qué alguien va a ser encarcelado por sus actos, si no puede hacer nada para evitarlo?.

En su destino ya estaba escrito que esto sucedería, no ha sido él quien a cometido un crimen, simplemente a cumplido su destino, en nuestro sistema jurídico actual, cuando una persona no tiene capacidad de decisión no va a la cárcel, va a un centro psiquiátrico, ..), un invento de lo más práctico, aunque no parezca de lo más lógico. Estamos ante la sociedad de lo práctico y no de lo lógico. A nosotros nos interesa la "aleatoriedad", y podríamos estar discutiendo sobre si existen procesos aleatorios o no al igual que los deterministas y los no deterministas, pero vamos a tirar por el camino práctico y no por el filosófico.

Consideraremos que algo es aleatorio, cuando no se puede volver a reproducir con los medios actuales, en un tiempo más o menos corto, en exactamente las mismas condiciones que se hizo anteriormente. De ahí el título del trabajo: la aleatoriedad como principio filosófico y la pseudoaleatoriedad3 que no se plantea la existencia o no de la aleatoriedad pero que intenta imitarla cumpliendo los requisitos anteriormente expuestos: no repetición del hecho, en exactamente las mismas condiciones.

En numerosas ocasiones de desconocer cuál es el proceso y lo único que se tiene es el resultado, para lo cual contamos con una batería de test que nos dicen si esos resultados provienen o no de un proceso "aleatorio", esto ya es Estadística. Para la Informática, y en concreto en el campo de la seguridad en redes tiene mucha importancia la aleatoriedad, ya sea en sistemas de cifrados asimétricos como en los simétricos. Este trabajo de investigación, solo es eso, un trabajo de investigación, ya que se puede seguir trabajando en él, desarrollando algunos puntos concretos para realizar una posible tesis doctoral. Por ejemplo hacia donde pueden ir los nuevos caminos en lo que ha cifrado se refiere: en un primer momento se usó cifrado simétrico, ahora estamos en cifrado asimétrico con clave pública-privada, quizás el futuro pase por la teoría del caos, al igual que en otras ciencias donde se ha pasado del determinismo a la estadística y de la estadística a la teoría del caos para poder explicar ciertos fenómenos. Existen diversos autores que dicen que el azar no existe que el caos sí.

En el campo de la seguridad-informática es muy importante la aleatoriedad, en los últimos años es algo que se tiene muy en cuenta. Ahora ya no sucede como antaño cuando se asignaban login a los alumnos de un determinado centro, su contraseña se formaba a partir de una política fija, con lo que cualquiera que supiera los datos de otro podía averiguar su contraseña. Es importante por tanto que aunque el login de usuario siga una política fija, la contraseña sea aleatoria. Este hecho lo encontramos en casi todos los campos, ya que cada vez más tenemos login y contraseña para casi cualquier cosa.

2.3 Generación de Variables Aleatorias.

La generación de cualquier variable aleatoria se va a basar en la generación previa de una distribución uniforme (0,1). Y las transformaciones de dichos números generados en valores de otras distribuciones.

La mayoría de las técnicas utilizadas para la generación se pueden agrupar en:

Ø Método de la transformada inversa

Ø Método de aceptación-rechazo

Ø Método de composición

Ø Método de convolución

Método de la transformada inversa

Es el método más directo para generar una variable aleatoria. Sea

$$F(z), a \leq z \leq b$$

Una función de distribución cuya función de distribución inversa es:

$$F^{-1}(u) := inf\{z \in [a,b] : F(z) \geq u, 0 \leq u \leq 1\}$$

Sea U una variable aleatoria de

$$\mathcal{U}(0,1),$$

Se verifica que

$$Z = F^{-1}(U)$$

Tiene la función de distribución F. La prueba se sigue de la observación de que

$$pr(Z \leq z) = pr[F^{-1}(U) \leq z] = pr[U \leq F(z)] = F(z)$$

Esto sugiere inmediatamente el siguiente esquema de generación:

Algoritmo del método de la transformada inversa

Propósito: Generar Z aleatoriamente de

$$F(z), a \leq z \leq b.$$

Entrada: Capacidad para evaluar

$$F^{-1}(u), 0 \leq u \leq 1.$$

Salida: Z

Método: Generar aleatoriamente U de

$$\mathcal{U}(0,1),$$

$$Z \leftarrow F^{-1}(U).$$

Ejemplo. La distribución exponencial

Supongamos que tiene una distribución exponencial de media beta. La función densidad de probabilidad es:

$$f(x) = \begin{cases} (1/\beta)e^{-x/\beta} & \text{si } x \geq 0 \\ 0 & \text{en el caso contrario} \end{cases}$$

La función de distribución (acumulativa) es:

$$F(x) = \int_{-\infty}^{x} f(t)dt = \begin{cases} 1 - e^{-x/\beta} & \text{si } x \geq 0 \\ 0 & \text{en el caso contrario} \end{cases}$$

Método de aceptación rechazo

Este método es más probabilístico que el anterior. Los métodos de inversión, composición y convolución son métodos de generación directos, en el sentido en que tratan directamente con la función de distribución. El método de aceptación-rechazo es menos directo en su aproximación.

Se va aplicar este método en el caso de que la variable aleatoria sea continua, el caso discreto es análogo.

En este caso tenemos la función de densidad *f(x)* de la variable y necesitamos una función *t(x)* que la acote, es decir *t(x)³f(x)* "*x*. Hay que notar que *t(x)* no es, en general, una función de densidad

$$c = \int_{-\infty}^{+\infty} t(x)dx \geq \int_{-\infty}^{+\infty} f(x)dx = 1$$

Pero la función *r(x)=t(x)/c*, si es claramente una función de densidad. (Suponemos que *t* es tal que *c*<¥). Debemos de poder generar (esperamos que de forma fácil y rápida) un valor de la variable aleatoria que sigue la función *r(x)*. El algoritmo general queda como sigue:

Generar *x* que siga la distribución *r(x)*

Generar *u~U(0,1),* independiente de *x*

$$\text{Si } u \leq \frac{f(x)}{t(x)}$$

Entonces devolver *x* si no volver a repetir el algoritmo

El algoritmo continúa repitiéndose hasta que se genera un valor que es aceptado.

Para hacer que se rechacen el menor número de puntos posibles la función *t(x)* debe ser la mínima función que acote a *f(x)*.

Método de composición

Este método va a poder ser aplicado cuando la función de densidad es fácil de

$$f(x) = \sum_{i=1}^{n} t_i(x)$$

Siendo *n* el número de trozos en los que se ha dividido la función.

Cada uno de los fragmentos se puede expresar como producto de una función de distribución y un peso

$$t_i(x) = f_i(x) w_i$$

y la función de distribución global la podemos obtener como

$$f(x) = \sum_{i=1}^{n} w_1 f_1(x) \text{ con } \sum_{i=1}^{n} w_i = 1 .$$

El método consiste en generar dos números aleatorios, uno sirve para seleccionar un trozo y el otro se utiliza para generar un valor de una variable que sigue la distribución de dicho trozo. El valor de la variable obtenida es el valor buscado.

El algoritmo general queda como sigue:

Generar u1,u2~U(0,1)

Si u1=w1 entonces generar x~f1(x)

Si no

Si u1=w1+w2 entonces generar x~f2(x)

Método de convolución

Mochas variables aleatórias incluyendo la normal, binomial, poisson, gamma, erlang, etc, se pueden expressar de forma exalta o aproximada mediante lá suma lineal de otras variables aleatorias.

El método de convolución se pede usar siempre y cuando lá variable aleatória *x* se pueda expresar como una combinación lineal de *k* variables aleatórias:

$$x=b_1x_1+ b_2x_2 +...+b_kx_k$$

En este método se necessita generar *k* números aleatórios (*u1,u2,...,uk*) para generar (*x1,x2,...xk*) variables aleatórias usando algunos de lós métodos anteriores y aís poder obtener un valor de lá variable que se desea obtener por convolución.

2.3.1 Variables Aleatorias Discretas.

Al realizar un experimento generalmente estamos interesados en alguna función del resultado más que en el resultado en sí mismo. Así, por ejemplo, al arrojar un dado dos veces podríamos estar interesados sólo en la suma de los puntos obtenidos y no en el par de valores que dio origen a ese valor de la suma. Esa cantidad de interés, o más formalmente esa

función a valores reales definida sobre el espacio maestral se denomina variable aleatoria. Variable porque toma distintos valores y aleatoria porque el valor observado no puede ser predicho antes de la realización del experimento, aunque sí se sabe cuáles son sus posibles valores. Dado que el valor de una variable aleatoria (en adelante lo abreviaremos v.a.) es determinado por el resultado de un experimento, podremos asignar probabilidades a los posibles valores o conjuntos de valores de la variable. Ejemplo: Se arroja dos veces un dado equilibrado. Un espacio maestral asociado es:

$$S = \{(x_1, x_2) / x_i \in \{1,2,3,4,5,6\}\}$$

Posibles v.a. asociadas con este experimento son:

X: "número de caras pares"

Y: "máximo puntaje"

Z: "suma de puntos"

Definición: Sea S un espacio muestra asociado con un experimento aleatorio. Una variable aleatoria X es una función que asocia a cada elemento w ∈S un número real X(w)=x, es decir:

X:Z → R

Como se observa, en general representaremos a las v.a. con letras mayúsculas: X, Y, Z, etc. y sus valores con letras minúsculas, es decir X(w)=x significa que x es el número real asociado al resultado w ∈S a través de X.

Ejemplos: 1) Volviendo al ejemplo anterior,

X((2,5)) = 1

Y((2,5)) = 5

Z((2,5)) = 7

X((1,3)) = 0

Y((1,3)) = 3

Z((1,3)) = 4

Y((2,2)) = 2

Z((2,2)) = 4

2.3.2 Variables Aleatorias Continuas.

Se dice que una variable aleatoria X es continua si su conjunto de posibles valores es todo un intervalo (finito o infinito) de números reales. Por ejemplo, una v.a. continua puede ser el tiempo de retraso con el que un alumno o un profesor llega al aula de clases ó también el peso o la estatura de los estudiantes de la FE.

La función f(x) es una función de densidad de probabilidad para la variable aleatoria continua X, definida sobre el conjunto de los números reales, sí:

1.- $f(x) \geq 0 \quad \forall x \in R$

2.- $\int_{-\infty}^{\infty} f(x)dx = 1$

3.- $P(a \leq X \leq b) = P(a < X \leq b) = P(a \leq X < b)$

$$= P(a < X < b) = \int_{a}^{b} f_X(x)dx$$

2.3.3 Pruebas T.

La prueba t de 1 muestra se utiliza para estimar la media de procesos y compararla con un valor objetivo. Esta prueba se considera un procedimiento robusto debido a que es extremadamente sensible al supuesto de normalidad cuando la muestra es moderadamente grande. De acuerdo a la mayoría de los libros de texto de estadística, la prueba t de 1 muestra y el intervalo de confianza t para la media son apropiados para cualquier muestra con un tamaño de 30 o más.

En este apartado, describimos las simulaciones que realizamos para evaluar esta regla general de un mínimo de 30 unidades de muestra. Nuestras simulaciones se enfocaron en el impacto de la no normalidad en la prueba t de 1 muestra. Queríamos evaluar el impacto que tienen los datos poco comunes en los resultados de la prueba.

Con base en nuestra investigación, el Asistente realiza automáticamente las siguientes verificaciones de sus datos y muestra los resultados en la Tarjeta de informe:

- Datos poco comunes
- Normalidad (¿Es la muestra suficientemente grande para que la normalidad no represente un problema?)
- Tamaño de la muestra Para obtener información general sobre la metodología correspondiente a la prueba t de 1 muestra, consulte Arnold (1990), Casella y Berger (1990), Moore y McCabe (1993) y Srivastava (1958).

Nota Los resultados de este documento también se aplican a la prueba t pareada del Asistente, debido a que la prueba t pareada aplica el método de la prueba t de 1 muestra a una muestra de diferencias pareadas.

UNIDAD 3
PRUEBAS DE MONTECARLO

El método de Montecarlo permite resolver problemas matemáticos mediante la simulación de variables aleatorias. John Von Neumann, en los años 40 y con los primeros ordenadores, aplica la simulación para resolver problemas complejos que no podían ser resueltos de forma analítica. Montecarlo y su casino están relacionados con la simulación. La ruleta, juego estrella de los casinos, es uno de los aparatos mecánicos más sencillos que nos permiten obtener números aleatorios para simular variables aleatorias.

La simulación de Monte Carlo es un método que emplea números aleatorios uniformemente distribuidos en el intervalo [0,1] que es utilizado para resolver problemas donde la evolución con el tiempo no es de importancia. A continuación, se analizarán dos ejemplos para comparar una solución analítica con una solución obtenida por simulación.

Determinación del área de una figura

Cuando se desea calcular el área de un círculo de radio r = 10 cm no existen mayores problemas, ya que tanto el área a como su perímetro p pueden evaluarse analíticamente con las siguientes fórmulas:

$$a = \pi r^2$$

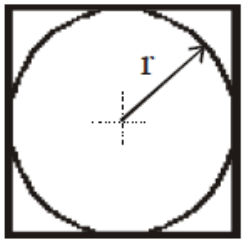
$$p = 2\pi r$$

En este caso la solución es $a = 314.16$ cm^2 y $p = 62.83$ cm.

Sin embargo, cuando se desea determinar el área de una forma irregular, por ejemplo la superficie plana de Argentina, el problema debe necesariamente ser resuelto con un método numérico; es decir, simulación. La determinación del área del círculo utilizando la simulación de Monte Carlo implica la siguiente secuencia:

1. Crear un cuadrado de lado 2.*r* que encierre al círculo (Figura 1).
2. Colocar *n* puntos al azar dentro del cuadrado.
3. Asignar a *c* el número de puntos que quedaron dentro del círculo.
4. Como la probabilidad de colocar un punto dentro del círculo es igual al cociente del Área del círculo dividida el área del cuadrado, el área del círculo se puede estimar en función del área del cuadrado (fácilmente calculable) con:

$$a_{círculo} = \frac{c}{n} a_{cuadrado} = \frac{c}{n}\left(4r^2\right)$$



Es importante notar que para un dado *n*, el resultado será distinto cada vez que se realice la simulación. Es decir, que el resultado será un número aleatorio. A medida que *n* aumente, la varianza del resultado disminuirá y el valor medio se aproximará a la solución analítica. Para un *n* = 100, el resultado de una simulación es 320 cm2; mientras que para *n* = 10000, un resultado es 313 cm2.

El mismo principio se puede aplicar para figuras complejas como se muestra en la Figura 2. Conociendo la escala, se puede fijar un cuadrado arbitrario y calcular el área de Argentina. Sin embargo, la determinación del perímetro de la figura es un problema de mayor magnitud para el cual se necesita recurrir a la teoría de fractales.

Evaluación de integrales

Suponga que se desea evaluar la siguiente integral que no tiene solución analítica:

$$I = \int_a^b g(x)\,dx$$

Si bien para este caso en particular existen mejores métodos para hacerlo, cuando se deben resolver integrales múltiples con integrando mal condicionados la simulación de Monte Carlo puede ser una buena alternativa.

Suponga que *x* es un número aleatorio con distribución uniforme continua en el intervalo [a,b], *f*(*x*) es la correspondiente función de densidad de probabilidad que es igual a 1/(*b*-*a*); entonces, el número *y* = *g*(*x*) es también un número aleatorio cuyo valor medio (E(*y*) o my) está dado por:

$$E(y) = \int_a^b g(x)\,f(x)\,dx = \int_a^b g(x)\frac{1}{b-a}dx = \frac{1}{b-a}\int_a^b g(x)\,dx = \frac{I}{b-a}$$

Por lo tanto:

$$I = (b-a)E(y)$$

Sin embargo, E(*y*) no es conocido; sólo puede ser estimado con el promedio de una muestra. Por el mismo motivo, *I* sólo puede ser estimado por el número aleatorio *Y* que se calcula de la siguiente manera:

$$Y = (b-a)\frac{\sum_{i=1}^{n} y_i}{n} = (b-a)\frac{\sum_{i=1}^{n} g(x_i)}{n}$$

Note que E(*Y*) = *I* y Var(*Y*) = (*b*-*a*).Var(*y*)/*n*, donde Var(y) (o σ_y^2) es la varianza de *y*.

La Tabla 1 muestra los resultados obtenidos para la siguiente integral:

$$I = \int_0^{\pi} sen(x)\,dx$$

n	10	20	40	80	160
Y	2.213	1.951	1.948	1.989	1.993

3.1 Lenguaje de Simulación.

A mediados de los años 60 se empezó a vislumbrar el uso de las computadoras para la simulación de problemas del mundo real, estos problemas estaban llenos objetos normalmente muy complejos, los cuales eran difícilmente traducidos a los tipos de datos primitivos de los pocos lenguajes de la época.

Así fue como a partir de esta necesidad a dos Noruegos se les ocurrió el concepto de "OBJETO" y sus colecciones CLASES DE OBJETOS. Nació así el lenguaje SIMULA, un lenguaje que contiene el embrión de lo que hoy se conoce como la PROGRAMACIÓN ORIENTADA A OBJETOS.

Sus creadores fueron Kristen Nygaard y Ole-Johan Dahl del Centro Noruego de Computación en Oslo, y su desarrollo se extendió desde 1962 a 1967. El objetivo inicial era definir un lenguaje de propósito específico para aplicaciones de simulación.

De hecho, realizaron una primera versión, bajo contrato con la empresa UNIVAC, que no incluía conceptos novedosos desde el punto de vista de programación -aunque sí desde el punto de vista de simulación- con respecto al lenguaje más avanzado de esos años, Algol 60.

La versión de 1967 tenía como uno de sus objetivos ahorrar esfuerzo de programación. Nygaard y Dahl habían desarrollado grandes programas de simulación con la primera versión, y habían detectado dos deficiencias:

I. Las entidades proceso y estación, útiles en simulación, eran entes dinámicos que se creaban y destruían a lo largo de una ejecución. El Concepto de bloque derivado de Algol 60, era insuficiente para reflejar este dinamismo. Por otra parte, cada entidad tenía asociadas un conjunto de variables y un conjunto de operaciones que las manipulaban. Con lo que, el código del programa no reflejaba claramente esta relación.

2. El código de muchas entidades era bastante semejante, pero el lenguaje no proporcionaba un mecanismo que permitiera reutilizar las partes comunes.

El primer hallazgo de Nygaard y Dahl fue la distinción entre una clase de entidades -un texto suministrado por el programador- y los objetos que se derivan de ella -los ejemplares de la misma creados y destruidos dinámicamente a lo largo de una ejecución concreta. Una clase en Simula 67 consiste en una colección de procedimientos, asociados a un conjunto de

declaraciones de variable. Cada vez que se crea un objeto de una clase, se asigna memoria para contener una colección de dichas variables. Esta idea, hoy familiar, exigía dos innovaciones con respecto a los lenguajes de la época:

• Escapar de la estructura de bloques. A diferencia de ésta, un objeto ha de "sobrevivir" al procedimiento que lo crea. Varios objetos de una misma clase han de poder coexistir en un mismo ámbito.

• Necesidad de un tipo de datos "referencia a un objeto" que permitiera designar objetos distintos en distintos momentos. Este tipo, llamado ref en el lenguaje, no era otra cosa que un puntero.

Con el comenzaron a introducir abstracciones de datos a los lenguajes de programación.

Además pensaron en la posibilidad de comenzar a reutilizar código, con sus respectivas y oportunas modificaciones, así se comenzó a formar la idea de Jerarquías de HERENCIA DE CLASES.

Fueron ellos también los que introdujeron el concepto de POLIFORMISMO introducido vía procedimientos virtuales, idea que derivó en el concepto de PROMOCION NUMERICA y según la cual los objetos pueden ser clasificados o tipificados en una serie de súper clases de forma que se establece la Jerarquía de Clase.

Todas estas ideas revolucionarias fueron plasmadas en una primera versión del lenguaje SIMULA, a principios de los 60, alcanzando su madurez con una versión final en el año 1967, dando lugar a SIMULA67.
Aunque pensado como un lenguaje de propósito general, Simula tuvo su mayor excito en las aplicaciones de simulación discreta, gracias a la clase SIMULATION que facilitaba considerablemente la programación.

3.1.1 Introducción.

El entorno macroeconómico al que tienen que enfrentarse las empresas es cada vez más incierto. Paralelamente, desde el punto de vista de la empresa misma, esta ha de hacer frente a una mayor competencia, y relacionarse con clientes cada vez menos cautivos al disponer de un elevado grado de información sobre el mercado. Ello se traduce en una irremisible bajada de resultados y una incertidumbre que comporta elevados niveles de riesgo. Ante esta situación surge la necesidad de manejar nuevos instrumentos para mejorar la planificación estratégica de las empresas.

Nuestro trabajo presenta la alternativa de aplicar modelos de simulación en los que se consideren los distintos escenarios posibles en las actividades clave de una empresa. Se trata de permitir a las empresas predecir, comparar y optimizar el comportamiento de sus procesos simulados en un tiempo muy breve sin el coste ni el riesgo de llevarlos a cabo, haciendo posible la representación de los procesos, recursos, productos y servicios en un modelo dinámico. Con la ayuda del correspondiente soporte informático, el modelo de simulación tiene la capacidad de considerar complejas tareas interrelacionadas y proyectarlas mediante la realización de muchas combinaciones alternativas en cuestión de segundos. Además, la interacción de los recursos con los procesos, productos y servicios sobre el tiempo se traduce en un gran número de escenarios y de posibles resultados imposibles de abarcar y valorar sin la ayuda de un modelo de simulación computarizado.

En este trabajo de investigación vamos a analizar la teoría de la simulación, sus antecedentes, los procesos, métodos y lenguajes de programación para la modelización a medida para cada empresa. En futuras investigaciones procederemos a implementar dichos modelos y a estudiar su validación.

3.1.2 Lenguaje de Propósito General.

El desarrollo de lenguajes de simulación comenzó a finales de los años 50; inicialmente los lenguajes que se usaron, fueron los de propósito general, los cuales tenían las siguientes ventajas:

La situación a analizar se puede modelar en forma más o menos sencilla para el programador por el conocimiento del lenguaje.

El proceso se puede describir con tanta precisión como les sea posible en el lenguaje conocido.

Se pueden realizar todas las depuraciones posibles.

Cualquier lenguaje de programación puede ser empleado para trabajar en simulación, pero los lenguajes,

Especialmente diseñados presentan las siguientes propiedades:

Acaban la tarea de programación.

Generan una guía conceptual

Colaboran en la definición de entidades en el sistema

Manejan le flexibilidad en los cambios

Ayudan a analizar y determinar la relación y el número de entidades en el sistema.

Hay un creciente número de lenguajes de programación disponibles para la implementación de modelos de simulación.

Entre los lenguajes de simulación destacan: gpss (general purpose simulation system), slam (simulation lenguaje for alternative modeling), simulan (simulation analysis) y simscript. Muchos lenguajes de propósito general son completamente adecuados para la simulación.

La masiva utilización de la informática en la enseñanza y en el entorno industrial, la sorprendente y revolucionaria evolución de los computadoras personales en cuanto a tamaño, costo, velocidad, software, etc. han ayudado sin lugar a dudas a que la simulación

digital o simulación por computadora sea hoy en día la herramienta más utilizada para realizar experimentos de simulación de sistemas. un programa de simulación de computadora se puede definir como una secuencia de instrucciones que el usuario define para resolver un problema que puede estar plasmado en unas ecuaciones que describen a un sistema que previamente hemos modelizado mediante dichas ecuaciones.

Visión general witness.

Witness es uno de los programas más punteros simulación de procesos dinámicos y cuya eficacia está avalada por varios centenares de compañías multinacionales y nacionales de gran prestigio. Se trata de una potente herramienta de simulación que permite modelar el entorno de trabajo, simular las implicaciones de las diferentes decisiones y comprender cualquier proceso, por muy complejo que éste sea. el resultado es obtener la mejor solución de negocio para su empresa antes de abordar cualquier inversión o cambio.

Es una herramienta fundamentalmente sencilla que incluye numerosas funcionalidades

Diseño sencillo y potente a través de bloques

Estructura jerárquica y modular

Facilidad de uso con implementación estándar en windows

Extremadamente interactiva

Potente conjunto de opciones de control y lógica

Elementos para producción discreta, industrias de procesos, bpr, comercio electrónico,

Call centers, salud, finanzas y gubernamentales

Extensas entradas e informes estadísticos

Visualizadores gráficos de calidad

Grandes enlaces a bases de datos (oracle, sql server, access, etc), enlaces de entrada y salida directos a hojas de cálculo, formatos xml, informes html, enlaces a bpm y aplicaciones cad, etc.

La familia de productos witness incluye:

Ediciones del software de modelado para producción o servicios y procesos.

Vistas 3d/vr completamente integradas o postprocesado vr

Optimización inteligente de modelos

Opcional— algoritmos únicos para encontrar la mejor respuesta rápidamente (optional)

Edición para desarrolladores (simba) para desarrollar aplicaciones de simulación con interfaces a medida (incluye un modelo de objeto completo, visores activex y software de visualización especial)

Un conjunto de soluciones de enlace con microsoft visio

Un conjunto de enlaces directos a cad

Opciones de cumplimiento de hla para aplicaciones militares y otras

Características principales:

witness posee una interfaz gráfica que permite comprender y mejorar nuestros procesos. witness es un programa para asistir a la evaluación de alternativas, apoyar importantes iniciativas estratégicas y mejoras continuas. su enfoque se basa en la creación de representaciones visuales de los sistemas de la vida real que, a través de modelos dinámicos, consiguen transformar simples datos en medidas productivas al mismo tiempo que fomentan el trabajo en equipo y la creatividad.

Entre sus prestaciones, destaca:

Dibujo del proceso de su negocio.

Técnicas y métodos de optimización.

Visualización en 3d.

Análisis de minería de datos.

Predicciones, planes y scheduling.

Información del fabricante

El fabricante de witness es lanner group. lanner es una empresa de software de simulación de procesos que proporciona a los directores de negocio una tecnología superior que mejora la comprensión de los procesos y soporta la optimización de procesos que resulta en mejores decisiones. lanner añade valor en cada etapa de la jornada del cliente, ofreciendo consultoría que proporciona descubrimiento guiado y análisis experto del problema; aplicaciones que dan potencia a los procesos de los usuarios e incrementan la capacidad de una organización para mejorar la productividad y ahorrar dinero.; y componentes de simulación automatizados incrustados en suites de software prominente. el software avanzado de simulación de lanner es proporcionado a los profesionales de simulación a través de su marca witness®. la marca l-sim™ de lanner se ha establecido rápidamente por sí misma como el motor de simulación de procesos embebidos principal utilizado en las suites de empresa por los mejores proveedores de soluciones. la tecnología de lanner también está incrustada dentro de su creciente rango de herramientas individuales, aplicaciones software de simulación de tareas específicas y planificación a través de una vasta formación de sectores de la industria incluyendo fabricación, automoción, farmacéutica y nuclear.

Áreas de aplicación

Por áreas de aplicación se identifican los siguientes campos:

Industria del automóvil.

Industria financiera.

Industria aeroespacial.

Industria alimentaria.

Industria del petróleo y el gas.

Industria electrónica.

Industria farmacéutica.

3.1.3 Lenguaje de Propósito Especial.

La masiva utilización de la informática en la enseñanza y en el entorno industrial, la sorprendente y revolucionaria evolución de los computadoras personales en cuanto a tamaño, costo, velocidad, softwares, etc. han ayudado sin lugar a dudas a que la simulación digital o simulación por computadora sea hoy en día la herramienta más utilizada para realizar experimentos de simulación de sistemas. Un programa de simulación de computadora se puede definir como una secuencia de instrucciones que el usuario define para resolver un problema que puede estar plasmado en unas ecuaciones que describen a un sistema que previamente hemos modelizado mediante dichas ecuaciones.

La construcción de un modelo de simulación ha pasado, de ser una labor reservada a especialistas en programación, de difícil y costosa realización, basada en procesos de lotes y en una interpretación en general elaborada a partir del procesado de tediosos listados, a ser un ejercicio estructurado alrededor de la utilización de entornos cada vez más amables y flexibles que permiten aprovechar la característica más destacable de la simulación : la posibilidad de estudiar la evolución dinámica de los sistemas a lo largo del tiempo.

Hoy en día al ingeniero se le abren un amplio abanico de posibilidades para resolver estos problemas y para programas estas operaciones necesarias para realizar la simulación. El abanico corresponde a los distintos lenguajes que podemos utilizar para traducir nuestros modelos en una computadora y posteriormente resolverlos para obtener la simulación del comportamiento del sistema modelado. Podemos utilizar lenguajes de programación general, lenguajes específicos para simulación (Lenguajes de propósito especial) o paquetes de software de simulación especialmente preparados para la misma. Aunque se han utilizado para realizar el ejercicio de la simulación ciertas herramientas como el EXCELL y Paquetes Integrados de Métodos Cuantitativos para la Toma de Decisiones, estos la limitan en su alcance.

A la hora de elegir una herramienta u otra hay que tener en cuenta primeramente la velocidad de la ejecución de los programas y la utilización de recursos necesaria (memoria, coprocesadores, etc.).

Hay distintos niveles de lenguajes, en el más bajo nivel se encuentra el lenguaje máquina cuyas instrucciones se escriben en la notación binaria que corresponden directamente con las funciones u operaciones elementales. Este lenguaje es sin duda el más tedioso y menos práctico de utilizar. En un nivel superior se encuentran el lenguaje ensamblador que utiliza símbolos (caracteres) nemónicos para representar dichas funciones

Los lenguajes de alto nivel o lenguajes de propósito general tales como C, Fortran, Basic, Cobol, Lisp, Algol, Pascal, etc. normalmente alejan al programador de las tareas de bajo nivel de la computadora y suelen ir apoyados en un conjunto de librerías que en el caso de la simulación facilitan mucho la tarea de modelizar los sistemas y reducen normalmente el tiempo de ejecución del programa.

En los años sesenta se realizaban estudios de simulación cuyos costos se medían en años-hombre y su duración en meses. En los setenta aparecieron diversos lenguajes específicamente orientados a la simulación tales como SIMSCRIPT, etc. La década de los

ochenta supuso la adaptación sobre PC de productos ya existentes y la aparición de nuevos productos como SIMAN.

Los noventa han protagonizado hasta ahora una auténtica explosión de nuevos productos de manejo más intuitivo bajo entornos gráficos como Windows. Es el caso de Simfactory, ProModel, Witness, Arena, Taylor II o Simvox, por ejemplo. La evolución de los computadoras y del software comercial se dirige hacia sistemas que puedan ser manejados por personas no-especialistas, con máquinas cada vez más potentes a menor coste. Las técnicas orientadas al objeto conducen a programas de utilización más intuitiva. Todo ello nos sugiere un incremento considerable de la aplicación de las técnicas de simulación.
Sin embargo, a pesar de todo, se estima que en el mercado norteamericano , que es el más desarrollado, sólo se tiene en cuenta la aplicación de técnicas de simulación en un 30% de los casos en los que podría aplicarse , y de este porcentaje , sólo en el 10% de los casos se utiliza regularmente. En Europa las cifras son menores, situándose en torno al 3%, a excepción de Inglaterra donde dicho porcentaje se eleva al 15%.

En el caso de utilizarse un lenguaje específico de simulación, la limitación está en que no permite desarrollar más allá de para lo que está pensado y diseñado el software, pero como contrapartida está que el usuario sólo precisa disponer de los conocimientos de programación relativos al producto. Los productos de modelización visual permiten realizar prototipos en tiempos récord siempre que los objetos a utilizar coincidan exactamente con los disponibles en el producto. En la medida que se requieran objetos específicos hay que recurrir a la programación.

La **Simulación Visual Interactiva,** que puede definirse como aquella que posibilita la creación gráfica de modelos de simulación, permite mostrar por pantalla dinámicamente el sistema simulado, así como la interacción entre el usuario y el programa en ejecución. La interacción implica que o bien se detiene la simulación y solicita información al usuario, o bien que éste puede parar la simulación a su voluntad e interaccionar con el mencionado programa ; esto último se puede realizar ”off-line” o “on-line”, es decir sin interrumpir la

simulación, e introduciendo las variaciones oportunas tanto en los modelos, como en los valores de las variables en el siguiente ciclo de scan del proceso de ejecución del programa en la computadora que para esto debe tener una estructura multitarea que permita este tipo de operaciones. Algunos productos del mercado son :SIMFACTORY DE CACI Inc. , PROMODEL de ProModel Corporation , ARENA de Rockwell Software Inc., WITNESS de ATT & Istel , o FACTOR/AIM de Pritsker Corporation , FIX DEMACS de Intellution (Fisher-Rosemount). Todos ellos son productos orientados primordialmente a la utilización de la simulación para la resolución de problemas en el ámbito de la producción. Utilizables desde entorno Windows, y ejecutables sobre computadoras personales o sobre plataformas más potentes como Estaciones de trabajo (Workstations).

Estos permiten construir modelos complejos de manera incremental, a partir de la selección de componentes del sistema de entre un repertorio limitado a la extensión de las librerías que contienen unas entidades predefinidas, si bien las últimas tendencias añaden a estos paquetes editores para crear nuevas plantillas con características a gusto del consumidor, introduciendo además utilidades de todo tipo incluidas las gestiones de configuración y control de las comunicaciones con un sistema de control real al que se puede conectar el equipo.

Ventajas:

• Sirven para comunicar la esencia del modelo de simulación a los directivos.

• Puede ayudar a corregir errores del programa de simulación, o a mostrar que el modelo no es válido.

• Puede ayudar a entender el comportamiento dinámico del sistema.

Inconvenientes:

• No puede sustituir a un cuidadoso análisis estadístico de los resultados.

• Sólo una parte de la lógica del modelo de simulación puede verse en la animación, y no se puede concluir a partir de ese corto periodo de tiempo que el modelo está bien definido.

• Aumenta el tiempo para desarrollar el programa de simulación.
• Muy lenta la animación en directo.

Otro enfoque se puede derivar de los lenguajes de simulación y de los simuladores es el de los Sistemas Híbridos que combinan la flexibilidad de un lenguaje de simulación con la facilidad de uso de un simulador como lo son el ARENA y el QUEST.

Los simuladores y lenguajes de simulación pueden adoptar uno de los diferentes métodos o estrategias. Existen tres estrategias que son generalmente reconocidas:

• **Enfoque de modelado basado en eventos.** La orientación basada en Eventos (ES) es gobernada por un calendario y ejecución de subrutinas (eventos) que como consecuencia programa la ejecución de otras subrutinas Los eventos son los instantes de tiempo en los cuales un cambio en el sistema ocurre y coincide con el inicio o terminación de las actividades. Bajo este enfoque segmentos del programa son empleados para definir cada evento en el modelo. Después de inicializado el modelo, las rutinas de ejecución revisan los tiempos de ocurrencia de los eventos y avanzan el reloj de la simulación hacia el tiempo en el cual ocurrirá el próximo evento. Debe existir una subrutina para cada tipo de evento,

• **Enfoque de modelado basado en actividades. La** orientación basada en Seguimiento de Actividades (SA) bajo este enfoque un segmento del programa es empleado para definir cada actividad en la cual las entidades se ven involucradas y las condiciones bajo las cuales la actividad puede realizarse. Dicho segmento incluye una serie de pruebas para determinar si la actividad ha sido iniciada en un punto del tiempo y define las acciones que se deben ejecutar si la actividad ha sido iniciada,

• **Enfoque de modelado basado en procesos.** La orientación basada en Interacción de Procesos (IP) es desarrollada desde el punto de vista de las entidades (transacciones) que

fluyen en el sistema. Bajo este enfoque las entidades se clasifican en transacciones o clientes, servidores o recursos (entidades permanentes y entidades temporales). En este enfoque, existen segmentos del programa que son empleados para describir los procesos en los cuales se ven involucradas las entidades.

UNIDAD 4
SIMULACION POR COMPUTADORA

Este texto se centrará en la simulación por computadoras. Un simulador por computadora está compuesto por las siguientes partes:

· **Un modelo:** Es un modelo simbólico. Puede ser un conjunto de ecuaciones, reglas lógicas o un modelo estadístico.

· **El evaluador:** Es el conjunto de procedimientos que procesarán el modelo para obtener los resultados de la simulación. Puede contener rutinas para la resolución de sistemas de ecuaciones, generadores de números aleatorios, rutinas estadísticas, etc.

· **La interfaz:** Es la parte dedicada a interactuar con el usuario, recibe las acciones del mismo y presenta los resultados de la simulación en una forma adecuada. Esta unidad puede ser tan compleja como la cabina utilizada en los simuladores de vuelos profesionales.

Resolución analítica vs. Simulación

Algunos modelos simbólicos pueden resolverse analíticamente. La ventaja de una solución analítica es que da una visión integral sobre la conducta del sistema. Variando sus parámetros es posible identificar fácilmente cambios importantes en el comportamiento, detectar puntos críticos y sacar conclusiones generales para el tipo de sistema analizado. Por ejemplo, la solución analítica del movimiento pendular permite concluir que el periodo (*T*) de cualquier péndulo es independiente de la posición inicial, pero depende de la longitud (*l*) del mismo:

$$T = \pi \sqrt{\frac{l}{g}}$$

En el caso del movimiento de un resorte, variando el coeficiente de fricción se puede Identificar dos tipos de respuestas características: la oscilatoria (con fricción nula) y la oscilatoria amortiguada (con fricción no nula).

Cuando se desea calcular las raíces del polinomio cuadrático:

$$P(x) = a x^2 + b x + c$$

Se dispone de la siguiente solución analítica:

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Esta solución analítica permite calcular fácilmente las nuevas raíces cuando se varían los coeficientes del polinomio. También, es claro que habrá problemas cuando el argumento de la raíz cuadrada se haga negativo.

Sin embargo, no siempre es posible obtener una solución analítica, ya sea por la naturaleza del modelo o de los experimentos que se desean realizar. En este caso, el modelo deberá ser tratado por algún tipo de método numérico. Esto es, el modelo será resuelto para un caso particular, y la solución será un número, un vector o una matriz; pero no se tendrá una función analítica. Debido a esto, el análisis de los resultados es más complejo que el requerido por una solución analítica. A continuación se da un ejemplo utilizando la simulación de Monte Carlo.

4.1 Con el Lenguaje de Propósito General.

Los lenguajes de simulación facilitan enormemente el desarrollo y ejecución de simulaciones de sistemas complejos del mundo real. El lenguaje de simulación de propósito específico es un paquete de computadora que permite realizar la simulación para un ambiente específico, no requiriendo esfuerzo en programación, es decir, produce un código más legible, modificable y menos largo.

Estos lenguajes permiten una orientación basada en procesos o basados en eventos. Los eventos son los instantes de tiempo en los cuales un cambio en el sistema ocurre y coincide con el inicio o terminación de las actividades y la orientación basada en procesos es desarrollada desde el punto de vista de las entidades (transacciones) que fluyen en el sistema.

Características de los lenguajes de simulación:

- Los lenguajes de simulación proporcionan automáticamente las características necesarias para la programación de un modelo de simulación, lo que redunda en una redacción significativa del esfuerzo requerido para programar el modelo.
- Proporcionan un marco de trabajo natural para el uso de modelos de simulación. Los bloques básicos de construcción del lenguaje son mucho más afines a los propósitos de la simulación que los de un lenguaje de tipo general.
- Los modelos de simulación son mucho más fácilmente modificables.
- Proporcionan muchos de ellos una asignación dinámica de memoria durante la ejecución
- Facilitan una mejor detección de los errores.
- Los paquetes de software especialmente diseñados para simulación contienen aplicaciones diversas que facilitan al simulador las tareas de comunicaciones, la depuración de errores sintácticos y de otro tipo de errores, la generación de escenarios, la manipulación "on-line" de los modelos, etc.

- Son muy conocidos y en uso actualmente
- Aprendizaje lleva cierto tiempo
- Simuladores de alto nivel
- Muy fáciles de usar por su interfaz gráfica
- Restringidos a las áreas de manufactura y comunicaciones
- Flexibilidad restringida puede afectar la validez del modelo

Algunos tipos de lenguajes de simulación específicos

Simulación SIMAN

En Ingles SIMAN significa Análisis, modelación y simulación (Simulation Modeling and Análisis). Este lenguaje fue desarrollado por C. Dennos Pedgen, Systems Modeling Corp.,Sewickley, PA. EUA.

Simulador SIMSCRIPT II.5

El SIMSCRIPT II.5 de la compañía de productos CACI, es un lenguaje que permite modelar y que puede ser orientado a eventos o orientado a procesos.

Simulador SLAM II

SLAM II (simulación lenguaje For Alternative Modeling) es un lenguaje de simulación por el cual se pueden construir modelos con orientación al proceso o al evento.

Ventajas de lenguajes específicos:

- Programas más cortos:

- Están preparados para llevar la contabilidad de los distintos parámetros.- Tienen módulos para generar números aleatorios de las distintas distribuciones, reloj, etc.

- Al tener menos líneas de código:

- Es más fácil detectar posibles errores.- Se reduce el tiempo de programación.- Es más fácil cambiarlo si queremos introducir alguna variación.

- Existen una serie de errores típicos que suelen ser identificados y chequeados de forma automática accesibles.

- Integra funciones como generación de números aleatorios, análisis estadístico y gráficas.
- Tienen una alta fiabilidad que conduce a una validación de resultados sencilla y rápida.

Desventajas/Inconvenientes de lenguajes específicos:

- Son menos flexibles, y su sintaxis tiende a ser menos natural.
- Las ejecuciones son más lentas.
- Son menos conocidos por analistas y programadores, y los compiladores son menos accesibles.
- Es necesario invertir en la adquisición del software.

Lenguaje de Simulación SIMAN

Versión original (Simulation and Analysis) desarrollada por Dennis Pegden, en la Universidad de Alabama. Cuando era líder del grupo de desarrollo de la versión original de SLAM (basada en los software de GASP y Q~GER-r de Pristker and Associates). Más tarde, Pegden Inicia su trabajo en el Pennissylvania State University donde lo diseña como un lenguaje de modelamiento para propósitos generales. Desde su implementación inicial en 1994, ha sido continuamente refinado por System Modeling Corporation , y en 1998 y 1989 el lenguaje fue completamente rediseñado dando origen a SIMAN/Cinema.
El ambiente de modelamiento en SIMAN se desarrolla entre:

- El **Modeling**, se describen los componentes del sistema y sus interacciones.
- El **Experiment**, donde se definen las condiciones del experimento (longitud de la corrida, condiciones iniciales).

SIMAN modela un sistema discreto usando la orientación al proceso; es decir, en un modelo de sistema particular, se estudian las entidades que se mueven a través del sistema. Una entidad para SIMAN es un cliente. Un objeto que se mueve en la simulación y que posee características únicas conocidas como atributos.

4.2 Con el Lenguaje de Propósito Especial.

La importancia de escribir modelos de simulación en lenguajes de propósitos generales como FORTRAN radica en:

- Permite conocer los detalles íntimos de la simulación.
- Es imprescindible, cuando no se dispone de software de simulación.
- Algunos modelos en lenguajes de simulación permiten interfaces con lenguajes generales, específicamente FORTRAN (ocurre con SLAM II, SIMAN, GPSS).

Por otra parte, los lenguajes de simulación ofrecen mayores ventajas, porque:

- Automáticamente proveen muchas de las facilidades necesarias en la simulación del modelo.
- Proveen un natural ambiente para modelamiento de la simulación.
- Son fáciles de usar.

Clasificación de los software para simulación Proveen una gran interacción entre edición, depuración y ejecución. Alcanzando algunos de ellos implantación de la ingeniería de software.

Existen en el mercado dos grandes clases de software para simulación: los lenguajes y los simuladores. Un lenguaje de simulación es un software de simulación de naturaleza general y posee algunas características especiales para ciertas aplicaciones, tal como ocurre con SLAM 11 y SIMAN con sus módulos de manufactura. El modelo es desarrollado usando las instrucciones adecuadas del lenguaje y permitiendo al analista un gran control para cualquier clase de sistema.

Un simulador (o de propósitos especiales) es un paquete de computadoras que permite realizar la simulación para un ambiente específico, no requiriendo esfuerzo en programación. Hoy en día existen simuladores para ambientes de manufactura y sistemas de comunicación permitiendo un menor tiempo en el desarrollo del modelo, así como también contar con el personal sin experiencia en simulación.

Los simuladores son actualmente muy utilizados para análisis en alto nivel, requiriéndose únicamente agregar detalles en un cierto nivel, puesto que lo demás es estándar.

CACI Products Company autor de SIMSCRIPT 11.5 es también autor de los simuladores SIMFACTORY 11.5, NETWORK 11.5 y COMNET 11.5, muy utilizados en estos últimos tiempos para simulaciones de sistemas de manufacturas, redes de computadoras y redes de telecomunicaciones.

Para procesar transacciones en espera de un ordenamiento, un lenguaje de simulación debe proporcionar un medio automático de almacenamiento y recuperación de estas entidades. Atendiendo a la orientación del modelamiento de una simulación discreta, existen tres formas:

1. Programación de eventos.
2. Procesos.
3. Examinación de actividades.

Una programación al evento es modelada, identificando las características del evento y luego se escriben un juego de rutinas para los eventos con la finalidad de describir detalladamente los cambios que ocurren en el tiempo en cada evento. Lenguajes como SIMSCRIPT 11.5 y SLAM 11 están orientados al evento.

Una interacción al proceso es una secuencia de tiempos interrelacionados, describiendo la experiencia de una entidad a través del sistema. Por ejemplo, en un modelo de colas esta "historia" se traduce en el paso del tiempo del ingreso a la cola, ingreso al servidor, paso del tiempo en el servicio y fin del servicio. GPSS, SIMAN y SIMNET son orientados al proceso.

En el examen de actividades, el modelador define las condiciones necesarias al empezar y finalizar cada actividad en el sistema. El tiempo es avanzado en iguales incrementos de tiempo y en cada incremento de tiempo, las condiciones son evaluadas para determinar si alguna actividad puede estar empezando o terminando. El ESCL, es un lenguaje de simulación muy popular en Europa y fue desarrollado en FORTRAN.

GASP IV

Es una colección de subrutinas FORTRAN, diseñadas para facilitar la simulación de secuencia de eventos. Cerca de 30 subrutinas y funciones que proveen numerosas facilidades, incluyendo:

- Rutinas de avance del tiempo,
- Gestión de listas de eventos futuros,
- Adición y remoción de entidades.
- Colección de estadísticas.
- Generadores de variables aleatorias.
- Reporte estándar.

El programador únicamente provee un program main, una rutina de actualización, rutinas de eventos, generadores de reportes personalizados y una subrutina denominada EVNTS. El programa main debe incluir la sentencia CALL GASP; siendo GASP una subrutina que determina el eminente evento, invocando a EVNTS escrita por el usuario y obtiene el índice NEXT.

GASP IV es un lenguaje de simulación desarrollado por Alan B. Pristker y N. Hurst en 1973. Es un lenguaje híbrido porque puede ser usado para programadores de simulación discretos, continuos y combinados; siendo el primero en integrar completamente estos dos ambientes de función del tiempo. GASP IV es un derivado del GASP II, y se diferencia por la definición del evento espacio-estado (state space event).

SIMSCRIPT II.5

Desarrollado en la RAND Corporation por H. Markowtz en los inicios de los sesenta. SIMSCRIPT 11.5. Es un lenguaje de simulación con orientación al evento y al proceso, es híbrido porque posee facilidades para simulación de sistemas discretos y continuos. Un programador SIMSCRIPT 11.5 consiste de las siguientes partes:

- Preamble
- Main program
- Rutinas de eventos.
- Rutinas ordinarias.

SIMSCRIPT 11.5, producido por CACI Products Company (La Jolla, California), fue utilizado en el pasado en grandes y complejas simulaciones, como es el caso de los modelos no orientados a colas; por ejemplo modelos de combates militares. Se encuentra disponible en versión PC destacando su ambiente de S11VIGRAPHICS.

SIMSCRIPT 11.5 está basado en entidades, atributos y conjuntos. Visualiza el mundo a ser simulado como un conjunto de entidades que pueden ser descritas a través de sus atributos y los eventos que aparecen en el tiempo.

SIMAN/Cinema

La versión original del SIMAN (Simulation and Analysis) fue desarrollada por Dennis Pegden, en la Universidad de Alabama, cuando era líder del grupo de desarrollo de la versión original de SLAM (basada en los software de GASP y Q~GER-r de Pristker and Associates). Más tarde, Pegden inicia su trabajo en el Pennisylvania State University donde lo diseña como un lenguaje de modelamiento para propósitos generales, incluyendo facilidades de manufactura muy útiles en modelamiento de sistemas complejos de manufactura.

Desde su implementación inicial en 1984, ha sido continuamente refinado por System Modeling Corporation, y en 1998 y 1989 el lenguaje fue completamente rediseñado dando origen a SIMAN/Cinema.

El ambiente de modelamiento en SIMAN se desarrolla entre el Modeling y el Experiment; en el primero se describe las componentes del sistema y sus interacciones y en el segundo se definen las condiciones del experimento (longitud de la corrida, condiciones iniciales).

SIMAN modela un sistema discreto usando la orientación al proceso; es decir, en un modelo de sistema particular, se estudian las entidades que se mueven a través del sistema. Una entidad para SIMAN es un cliente, un objeto que se mueve en la simulación y que posee características únicas conocidas como atributos. Los procesos denotan la secuencia de operaciones o actividades a través del que se mueven las entidades, siendo modeladas por el diagrama de bloques.

Usted construye un diagrama de bloque en un flowchart gráfico, seleccionando y combinando bloques. Después, interactivamente, usando un editor especial se activa el generador automático de las sentencias del modelo desde el ambiente gráfico. Los bloques de SIMAN se clasifican en 10 tipos básicos.

SLAM II

El SIMPSCRIPT y el GASP IV son los lenguajes de programación de eventos más destacados.

SLAM es un descendiente de GASP IV que ofrece también recursos de simulación de redes y continuos, estando ambos codificados en FORTRAN.

Desde los lenguajes orientados a los procesos, existe representación de modelos en bloques como GPSS y SIMAN y los basados en redes como Q-GERT y SLAM.

Con la llegada del PERT, se plantearon situaciones de redes complejas, en tanto a ramificación por efecto de una decisión y loop para conseguir que varias actividades se realicen de modo repetitivo, trayendo consigo el desarrollo del GERT (Graphical Evaluation and Review Technique), por Pritoker y Elaghraby; quienes lo aplicaron para el programa Apolo.

El lenguaje Q-GERT significó la respuesta al cálculo de estimación de probabilidades de terminación en cada nodo y la distribución de tiempos y costos para la realización de cualquier nodo, la estructura básica de un modelo de simulación Q-GERT es una red compuesta de nodos y actividades (bifurcaciones). SLAM es una variante de QGERT que ofrece recursos de eventos de redes y discretos (y también simulación continua).

SLAM II (Simulation Languaje for Alternative Modeling) es un lenguaje de simulación por el cual se pueden construir modelos con orientación al proceso o al evento. SLAM fue desarrollado en 1979 por Dennis Pedge y Alan Pritsker y es distribuido por Pritsker Corporation (indianapolis, Indiana). La parte de SLAM que se orienta a los procesos emplea una estructura reticular compuesta por símbolos de nodos y ramas tales como colas, servidores y puntos de decisión. Modelamiento significa incorporar esos símbolos a un modelo de red que representa el sistema y en donde las entidades (ítems) pasan a través de la red. SLAM contiene un procesador que convierte la representación visual del sistema a un conjunto de sentencias.

La parte orientada a los eventos permite incluir rutinas en FORTRAN para las relaciones lógicas y matemáticas que describen los cambios en los eventos.

Un modelo continuo es especificado por las ecuaciones diferenciales o de diferencia, el que describe la conducta dinámica de las variables de estado. El modelador codifica esas ecuaciones en FORTRAN, empleando un juego especial de arreglos de almacén SLAM.

El SLAM simplifica el modelamiento de sistemas complejos, combinando el uso fácil de lenguaje de proceso como GPSS y Q-GERT con la potencia y flexibilidad del lenguaje de eventos GASP IV.

4.3 Sobre Sistemas de Inventario, Teoría de Colas, de Producción, etc.

Introducción

Un Director de Operaciones gestiona recursos limitados para dar servicio a los diferentes requerimientos que la organización tiene. En función de la calidad de su gestión (y de los recursos disponibles) el tiempo de espera (de clientes, productos y recursos) será mayor o menor. Desde ese punto de vista se podría decir que la función de un Director de Operaciones es decidir quién (o qué) debe esperar a qué (o a quien).

Todos hemos experimentado en alguna ocasión la sensación de estar perdiendo el tiempo al esperar en una cola. El fenómeno de las colas nos parece natural: esperamos en el coche al estar en un tapón, o un semáforo mal regulado, o en un peaje; esperamos en el teléfono a que nos atienda un operador y en la cola de un supermercado para pagar....

Pero a veces las esperas son buenas. Nos hacen visualizar la importancia del producto o servicio que vamos a adquirir, nos permiten pensar y reconfigurar nuestro requerimiento.

Pero en general como clientes no queremos esperar, los gestores de los citados servicios no quieren que esperemos.... ¿Por qué hay que esperar? ¿Cuánto hay que esperar?

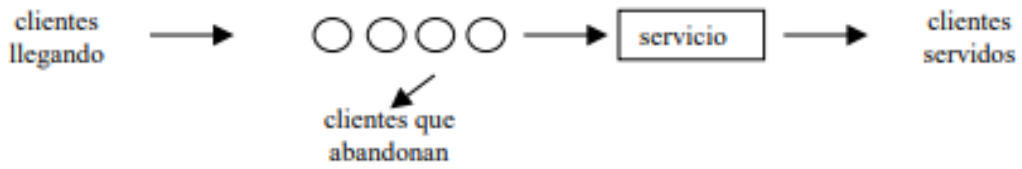
La respuesta es casi siempre simple, en algún momento la capacidad de servicio ha sido (o es) menor que la capacidad demandada. Esta limitación se puede eliminar invirtiendo en elementos que aumenten la capacidad. En estos casos la pregunta es: ¿Compensa invertir en máquinas? ¿O mejor invertimos en salas de espera? En ese caso ¿cómo de grandes?

La teoría de colas intenta responder a estas preguntas utilizando métodos matemáticos analíticos.

Descripción de un sistema de colas

Un sistema de colas se puede describir como sigue. Un conjunto de "clientes" llega a un sistema buscando un servicio, esperan si este no es inmediato, y abandonan el sistema una vez han sido atendidos. En algunos casos se puede admitir que los clientes abandonan el sistema si se cansan de esperar.

El término "cliente" se usa con un sentido general y no implica que sea un ser humano, puede significar piezas esperando su turno para ser procesadas o una lista de trabajo esperando para imprimir en una impresora en red.



Aunque la mayor parte de los sistemas se puedan representar como en la figura 1, debe quedar claro que una representación detallada exige definir un número elevado de parámetros y funciones.

La teoría de colas fue originariamente un trabajo práctico. La primera aplicación de la que se tiene noticia es del matemático danés Erlang sobre conversaciones telefónicas en 1909, para el cálculo de tamaño de centralitas. Después se convirtió en un concepto teórico que consiguió un gran desarrollo, y desde hace unos años se vuelve a hablar de un concepto aplicado aunque exige un importante trabajo de análisis para convertir las fórmulas en realidades, o viceversa.

Características de los sistemas de colas

Seis son las características básicas que se deben utilizar para describir adecuadamente un sistema de colas:

a) Patrón de llegada de los clientes

b) Patrón de servicio de los servidores

c) Disciplina de cola

d) Capacidad del sistema

e) Número de canales de servicio

f) Número de etapas de servicio

4.4 Validación.

En Estadística, al igual que en otras teorías de la Matemática, además de desarrollar modelos matemáticos que regularicen las observaciones en series estadísticas, es importante ver cómo se adaptan estos modelos a la realidad; así en mecánica se deducen las leyes de Kepler, que son leyes matemáticas, y una segunda cuestión es ver si se adapta a la realidad y con qué aproximación.

Al aplicar métodos estadísticos para obtener nuevos conocimientos de fenómenos naturales se pueden considerar cuatro etapas: descripción, modelización, verificación y predicción.

1.- Descripción estadística, que se propone la recogida, clasificación y presentación resumida de datos relativos a un fenómeno.

2.- Para explicar los hechos observados se formulan hipótesis, teorías o modelos, que expresan en forma matemática las relaciones que se han observado en los datos estadísticos.

3.- Una tercera fase se refiere a la verificación del modelo mediante la recogida de nuevos datos estadísticos relativos al fenómeno estudiado. Si la ley se confirma con los nuevos datos, puede utilizarse en lo sucesivo; si no, habrá de comenzar nuevamente el ciclo y modificar convenientemente las hipótesis, teorías o modelos.

4.- La teoría o hipótesis establecida permite hacer deducciones y obtener consecuencias, no obtenidas directamente de los experimentos sino mediante razonamientos lógicos. Esta teoría es la base de previsiones relativas a nuevos experimentos o sucesos.

Hoy en día, y con el desarrollo de nuevas tecnologías, los científicos han utilizado estas cuatro fases, con el fin de desarrollar modelos de simulación sobre el comportamiento de cualquier fenómeno, y así inicialmente describen el fenómeno, lo modelizan con ecuaciones matemáticas cuyo acoplamiento suele ser complejo, y una vez validado dicho modelo, intentan predecir el comportamiento futuro del fenómeno. Existen modelos de simulación, sobre fabricación de aviones, construcción de edificios... pero, aquí y aunque inicialmente no se tenga en cuenta, también se encuentran las técnicas estadísticas, sobre todo en la validación del modelo. Esta fase es imprescindible debido a que es necesario conocer hasta qué punto el modelo representa bien la realidad y si no es así, conocer en qué procesos falla y a qué son debidos, pues no se debería comenzar una previsión futura antes de ser validado el modelo.

El proceso de validación de un modelo siempre está sujeto a un caso experimental dado. Por lo que, parece ser un tanto irrealista el decir "este modelo es válido" siempre habrá que decir para que tipo de experimento lo es.

Aquí se presentan los pasos a seguir dentro de un proceso de validación empírica. Como caso práctico, se presenta también, la validación realizada sobre, un modelo de simulación

térmica de edificios, el ESP, ante el comportamiento de una Célula Test, caseta prefabricada para el estudio térmico de edificios.

La validación a realizar es en sentido residual, es decir, se trata de realizar comparaciones de las salidas del modelo con las salidas medidas del caso experimental y observar si existen diferencias entre las mismas, y poder explicar a qué son debidas.

A la hora de comparar las predicciones del modelo con los datos observados se localizan muchas dificultades que se pueden reducir a dos problemas bien generalizados:

1. La cantidad de información del sistema físico que contienen los datos.
2. La calidad de esta información.

El diseño de un experimento para la validación de un modelo no es una tarea sencilla. Las decisiones sobre las señales a medir, donde, cuando y con qué frecuencia medirlas, todo ello condicionará la cantidad de información física sobre el sistema que será posible extraer de los datos recogidos. Muchas veces, el experimento no es lo suficientemente informativo ante los propósitos de la validación global del modelo y solamente se pueden llevar a cabo validaciones parciales. Por otra parte, las medidas casi siempre se encuentran afectadas por ruidos, siendo esto un aspecto importante a tener en cuenta al intentar validar el modelo.

También se pueden encontrar dificultades relacionadas con la naturaleza intrínseca de los códigos de simulación térmica. La complejidad y dimensión de tales códigos implican la necesidad de un experimento muy informativo para poder realizar el proceso de validación. Por otra parte, una gran cantidad de parámetros de entrada son necesarios para poder definir el modelo de un sistema térmico en una simulación detallada del mismo (dimensiones geométricas, propiedades térmicas y ópticas...). El valor particular de muchos de estos parámetros es "apriori" desconocido o su conocimiento suele estar afectado por ciertas incertidumbres, y en este caso, la definición de la incertidumbre está muchas veces basada sobre juicios o experiencias subjetivas.

Debido a las dificultades anteriormente expuestas, es de esperar que se obtengan ciertas desviaciones entre los datos medido y los esperados mediante la simulación, siendo difícil calificar a un modelo o detectar las causas de un comportamiento deficiente. Ahora bien, ciertas herramientas estadísticas son útiles ante los propósitos de la validación del modelo.

Dentro del proceso de validación de modelos se pueden destacar tres partes bien diferenciadas:

1. Análisis de sensibilidad

Los modelos físicos suelen representar un sistema multivariable tanto de entradas como de salidas (se trata de sistemas con multientradas y multisalidas), aunque el sistema sea físicamente sencillo, por lo que es imprescindible llevar a cabo un primer estudio (análisis de sensibilidad) para saber qué variables de entrada influyen sobre el comportamiento de las salidas, con el fin de simplificar el problema, ya que de este modo se podrán reducir los posteriores estudios a aquellas entradas que hacen sensible a la salida seleccionada. Dicho análisis viene a ser un primer estudio que guiará la validación del experimento y el diseño del mismo.

2. Búsqueda de dominios de definición de los parámetros de entrada.

Un deficiente comportamiento de las salidas simuladas frente a las experimentales puede ser debido a 2 causas bien diferenciadas:

Una mala definición de los parámetros de entrada

Un deficiente ajuste del modelo a la realidad.

Antes de asegurar que el mal ajuste de salidas simuladas y experimentales es debido a un problema intrínseco del modelo, es necesario asegurarse que los parámetros de entrada son óptimos. Debido a que el valor particular de muchos parámetros de entrada suele ser desconocido, o éste está afectado por ciertas incertidumbres, cuya definición está basada en juicios subjetivos, puede ocurrir que las inexactitudes, que se producen entre las salidas del modelo y las medidas en el caso experimental, sean debidas a una mala definición de los dominios de entrada de estos parámetros.

El dominio de estos no tiene porqué ser único, y se consideran como aceptables aquellos que hacen mínima la diferencia entre las salidas experimentales y teóricas. Por lo que se trata de una búsqueda de dominios que optimicen una función objetivo, desconocida a priori. Para ello, se expone un procedimiento que compagina métodos de Monte-Cario junto con Técnicas Clusters. Haciendo así una búsqueda jerarquizada de dichos dominios.

3. Análisis de Residuos

A pesar, de haber conseguido definir buenos dominios de los parámetros de entrada, pueden seguir existiendo inexactitudes (siempre con un cierto margen de error aceptable), siendo entonces necesario, caracterizar las inadecuaciones del modelo, con el fin de saber a qué son debidas, y poder corregir el modelo. Lo que hace imprescindible un análisis de las principales características de los residuos.

Bibliografía básica y complementaria:

Banks J., Carson J.S., Nelson B.L, 1996, "Discrete-Event System Simulation. Second Edition.", Prentice-Hall, New Jersey. Fishman G.S., 1978, "Conceptos y métodos en la simulación digital de eventos discretos", Limusa, México. Kelton W.D., Sadowski R.P., Sadowski D.A., 1998, "Simulation with Arena", Mc Graw Hill, Boston. Ogunnaike B.A., Harmon Ray W., 1994, "Process Dynamics, Modeling and Control", Oxford, New York. Shannon R.E., 1988, "Simulación de Sistemas. Diseño, desarrollo e implementación", Trillas, México. Law A.M., Kelton W.D., 1991, "Simulation Modeling & Analysis", Second Edition, McGraw-Hill, New York.

http://es.scribd.com/doc/73475784/Manual-Software-Witness

http://www.buenastareas.com/materias/witness-simulador/0

http://wwwdi.ujaen.es/asignaturas/computacionestadistica/pdfs/tema5.pdf

Coss Bu, Raúl; *Simulación: un enfoque práctico*; Editorial Limusa; 2003

http://jair.lab.fi.uva.es/~pablfue/leng_simulacion/materiales/func_alea_0405_resumen.pdf
descomponer en un conjunto de trozos,
Publicado por Carolina en 9:14

Banks, J. [1994),"Software for simulation," en 1994 winter Simulation Conference Proceedings, ed. J.D. Tew, S.Manivannan, D.A. Sadowski, A.F. Seila, Association for computing Machinery, New York, NY, pag. 26-33