**Universidad de Costa Rica**

**Facultad de Ingeniería**

**Escuela de Ciencias de la Computación e Informática**

**Investigación de Operaciones**

**Simulador de un sistema de colas sin prioridad adquirida**

**Profesor**

Raúl Hernández

**Estudiantes:**Luis Carlos Chavarría A61521

Ricardo Alvarado Villalobos A60289

**Martes 13 de Julio del 2010**

**Planteamiento del proyecto:**

Se pide construir una herramienta que simule una cola prioritaria sin prioridad adquirida, que tiene un solo servidor y atiene a 3 tipos diferentes de clientes. Los tiempos entre llegadas deben de ser definidos por el usuario, y pueden ser diferentes entre ellos, y para efectos práctico siempre con distribución exponencial. Asimismo, el usuario debe definir los tiempos de atención, que deben ser iguales para todos los tipos de clientes, con distribución exponencial.

De este planteamiento se quieren obtener los tiempos de espera, así como el tamaño de la cola.

**Modelado de la solución:**

Para solucionar este problema creamos un sistema que para lograr cumplir con los requerimientos, tiene los siguientes elementos:

**Cliente**: Representa un cliente en el sistema. Con tiene los datos propios de su hora de llegada, hora de salida, el tiempo que espero en la cola, el tipo de cliente que es, y un identificador único.

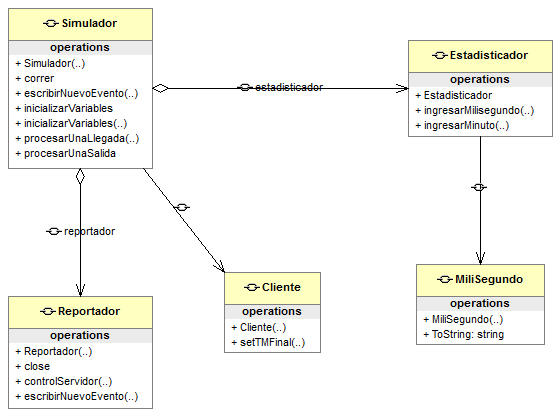
**Milisegundo**: Es un elemento que usamos para grabar el estado en un instante del sistema. Contiene el milisegundo que marca el sistema, el estado de la cola, y el estado del servidor. Es utilizado por el estadisticador para llevar el control de los eventos en el sistema.

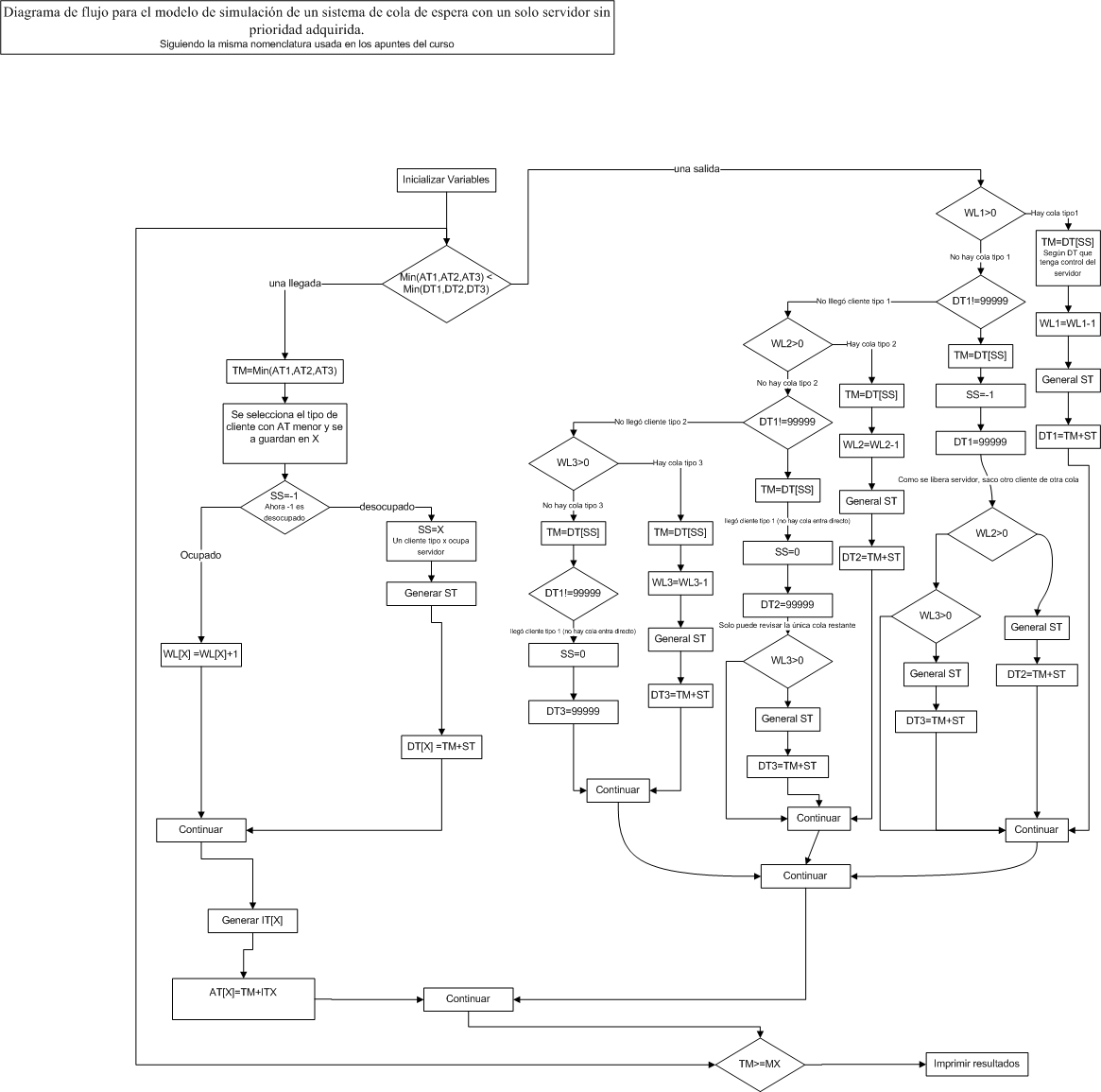
**Reportador**: Maneja una bitácora de eventos. Por cada evento que ocurre en el sistema, el reportador se encarga de escribirlo y dejarlo registrado en una bitácora. Esta bitácora en un archivo que se puede encontrar a la par del ejecutable usado para correr el sistema.

**Estadisticador**: Es el elemento que realiza los cálculos de tiempos entre eventos del sistema. Cada evento que sucede es reportado a este, y se encarga de hacer los cálculos y poder ponderar los tiempos. Al final del programa se reportan en la pantalla principal del programa.

**Simulador**: Es el cerebro del modelo. Se encarga de controlar el ingreso de clientes, manejo de colas, salida de clientes. Llama al estadisticador cuando sucede un nuevo evento, y se encarga de desplegar las estadísticas del estadisticador en pantalla una vez terminada la simulación.

Este diseño se ve reflejado en el diagrama de clases del sistema:



Diagrama de flujo de la solución:

El modelo inicia cuando el usuario define los datos del tiempo entre llegadas de los diferentes clientes, y el tiempo de atención para cada uno de ellos, así como la cantidad de ciclos que se desea que se realicen en la simulación. Una vez hecho esto, el sistema revisa si el tiempo de llegada del siguiente cliente es menor al tiempo de salida del próximo cliente en la cola. Si el tiempo de llegada es menor, entonces el sistema procesa una entrada. Al procesar una entrada se revisa si el servidor está ocupado. Si está ocupado, se envía el cliente a la cola, en caso contrario el cliente es atendido, sale inmediatamente del sistema y se continúa con el siguiente, sea de la cola o de una nueva entrada.

En caso de procesar una salida, el sistema revisa primero si hay algún cliente tipo 1 en la cola. Si hay, se procesa la salida, se genera el tiempo de servicio para ese cliente y se continúa con la simulación. En caso de que no hayan clientes tipo 1 en la cola, se procede a revisar si hay clientes tipo 2, de igual forma si no hay tipo 2, se buscan los tipo 3. Finalmente una vez que se han atendido a todos los clientes se genera un reporte y termina la simulación.

**Explicación de algoritmos importantes utilizados en este modelo:**

* **Cálculo de tiempo esperado en la cola (Wq):**

Como estamos en un modelo con distintos tipos de clientes, cada uno con una prioridad distinta, se calculan tres Wq llamados Wq1, Wq2 y Wq3 por tener 3 tipos de clientes.  
En el programa se usa la estructura de datos List, que es una simple lista para almacenar objetos. Se usan tres de estas cada una para cada tipo de cliente. Estas listas historialClienteTipo1, historialClientetipo2, historialClienteTipo3 se usan de la siguiente manera:  
Cuando un cliente entra en el sistema y pasa a la cola, se crea un objeto cliente. Este almacena el tiempo (TM) en que el cliente entró a la cola, qué tipo de cliente es, y que número de cliente es. Este objeto se guarda en la cola (pues el servidor está ocupado) y espera su turno para ser atendido. Eventualmente el cliente será sacado de la cola, y por ende el objeto asociado al cliente también será sacado de la cola. A este objeto (que contiene los datos antes mencionados) ahora se le guarda el tiempo en el cual está saliendo de la cola. De esta forma se puede obtener cuanto tiempo estuvo esperando ser atendido.

Si este objeto es un cliente tipo 1, se manda a la lista historialClienteTipo1, y así respectivamente con los otros tipos de clientes. Se hace este proceso con cada uno de los clientes que entra alguna vez a la cola, por lo que   
entonces al final tenemos 3 listas, cada una con cada cliente que estuvo alguna vez en la cola y cuanto tiempo estuvo. Ahora sólo sumamos todos los tiempos que estuvieron los clientes de cada tipo en la cola y los dividimos entre el tamaño de la cola. Así obtenemos el tiempo esperado que pasa un cliente tipo k en la cola (Wqk).

* **Cálculo de cantidad clientes esperados en la cola. (Lq):**

Como se ve en la bitácora que imprime el sistema al final, en cada evento se registran los diferentes variables del sistema como el estado de las colas (Lq1, Lq2, Lq3). Por ejemplo: al final del evento 3, tenemos el estado de cada cola, y la hora del evento. Desafortunadamente el evento 4 no ocurre en un tiempo constante respecto al anterior. Es decir, el tiempo entre eventos no es constante, ya que las llegadas son aleatorias, pues existen distintos tiempos en que ocurren los eventos. Es por eso que el peso de cada evento para ponderar el tamaño de las colas es variado.

Para solucionar este problema se hicimos algo similar a una interpolación (quizás abusando del término): Se creó una lista donde íbamos a guardar cada milisegundo del sistema (es la resolución escogida por eficiencia).

A continuación un ejemplo (en minutos para mejor comprensión):  
Sí en el evento tres que fue en el minuto 3, habían 3 clientes tipo 1 en la cola. Se agrega ese minuto a una lista. Pero si el siguiente evento (evento 4) ocurrió en el minuto 7, tenemos 4 minutos los cuales no tenemos información. Pero de hecho sí sabemos lo que pasó en esos 4 minutos, pues no es otra cosa que lo mismo lo que pasó al tercer minuto. Ya que si no hubo ningún evento, esto quiere decir que ninguna variable cambió, entonces antes de agregar ese sétimo minuto a la lista, agregamos los 4 minutos restantes con la exacta misma información que tenemos sobre el tercer minuto.   
Esto es lo que se hace en el programa, solo que en lugar de minutos usamos milisegundos, para una mayor resolución. Con mayor resolución quizás hubiera dado resultados más precisos, pero eran muchísimo más lentas las pruebas. Por esta razón seleccionamos milisegundos.

Es de esta forma que obtenemos nuestros Lq1, Lq2, Lq3

**Análisis de resultados:**

**Experimento 1:**

****

Para esta primera simulación utilizamos los valores 0.12 para la frecuencia de llegada de C1, 0.012 para la de C2, 0.9 para la frecuencia de llegada de de C3. Definimos que se hicieran 20000 simulaciones, y que el E(S) fuera 3, o sea que la frecuencia de atención seria de 0.3333333.

Viendo estos resultados, vemos que aplicando la formula, los resultados sobre todo del tiempo de espera en la cola de los clientes tipo 2 y de los clientes tipo 3 en la simulación están bastante lejos de la realidad. Esto puede deberse a una falta de más ciclos, ya que si vemos la cantidad de clientes en promedio si estuvo parecida, por lo que puede deberse a una falta de más datos en el experimento para poder acercar al resultado propuesto por la teoría.

**Experimento 2:**



En esta simulación vemos que aunque siguen existiendo pequeños desajustes en la aproximación entre los valores de la simulación y el modelo analítico, este margen de error se ha reducido, pues en todos los casos es menor a 1. El error sigue existiendo, pues ahora la diferencia entre llegadas es de varios minutos entre cada cliente, por lo que hay pocos elementos en la cola para realizar los cálculos. Si volvemos a realizar este experimento con una cantidad más grande de simulaciones (10000 en vez de 1000), veremos una mejoría en los resultados:

**Experimento 3:**



Es una diferencia poco apreciable, pero han disminuido en casi todos los casos los márgenes de error entre un resultado y el otro. El único caso en que no disminuyo fue el caso del tiempo de espera en cola de los clientes 2, que se mantuvo igual. Esto nos refleja que se necesita una cantidad muy grande de resultados para poder aproximar con exactitud los tiempos con una simulación.

De todos estos experimentos podemos concluir que la simulación es una buena herramienta para aproximar resultados y que, con grandes cantidades de procesamiento es posible tener respuestas muy acertadas que pueden ser útiles y confiables.