Implementación Modelo de Simulación basado en Eventos discretos
Caso de estudio: Minimarket

Angel Salgado Mancilla, angel.salgado@alumnos.uv.cl
Benjamín Morales Carvajal, benjamin.moralesc@alumnos.uv.cl

1. Introducción

La situación consiste en un minimarket que es atendido por una sola caja. Éste tiene solo dos tipos de productos, de tipo A y de tipo B. Cada uno siendo procesado de manera diferente. Los de tipo A tienen una cierta probabilidad de fallo, donde se tienen que ingresar de manera manual en la caja en caso de fallar. En la primera tarea de la asignatura se desarrolló un modelo DES que simulaba el caso de estudio (ver **Fig. 1**), además de plantear una pregunta de investigación: **¿Cuántas personas compran satisfactoriamente en el minimarket, y cuantos tipos de abarrotes compran en total?**

En este informe se mostrará el proceso de implementación utilizando un framework provisto en el lenguaje C++, demostrando las salidas su estructura genera y cómo este responde a las preguntas planteadas.

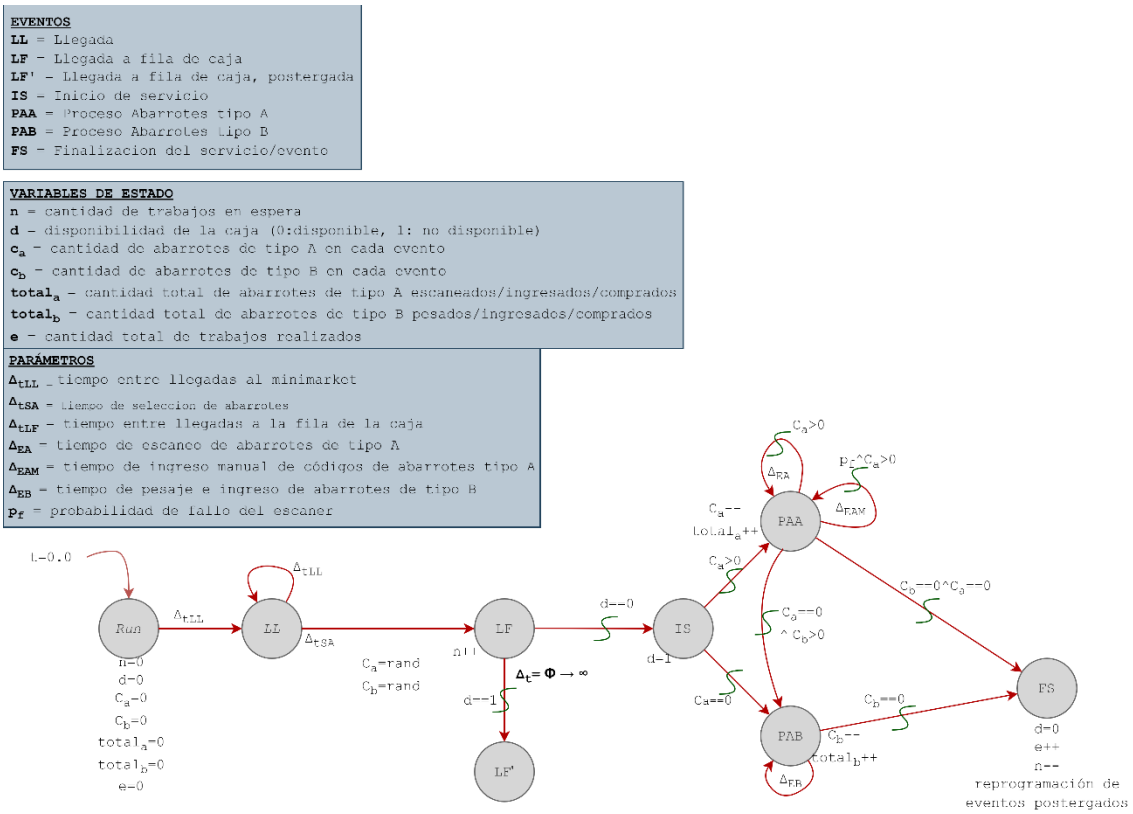

Figura 1. Modelo desarrollado en la tarea 01.

2. Materiales y Métodos

Para la implementación de este modelo se usó el framework basado en eventos provisto, el cual se hizo en C++ y se utilizó programación orientada a objetos. Para editar este simulador se utilizó la herramienta *Visual Studio Code* y la herramienta *make* para compilar y analizar errores.

2.1. Implementación de eventos

Para la implementación se comenzó añadiendo nuevos parámetros para controlar los resultados de la simulación. Posteriormente se implementaron nuevas clases y funciones para representar los eventos presentados en el modelo, estos incluyen:

- La llegada al minimarket.
- La llegada a la fila de la caja única.
- La ocupación de la caja (inicio del servicio).
- El escaneo de productos de tipo A.
- El escaneo de productos de tipo B.
- La finalización del servicio (La liberación de la caja).

2.2. Variables globales y de interés

Cada etapa de escaneo y finalización del servicio utiliza variables globales para contar y responder a la pregunta de investigación planteada en la tarea 01. Estas variables son:

- Total de abarrotes de tipo A
- Total de abarrotes de tipo B
- Cantidad de trabajos totales.

2.3. Estructura del simulador

El código implementado tiene la estructura mostrada en la **Figura 2**, donde se implementaron nuevas clases correspondientes a los eventos del modelo.

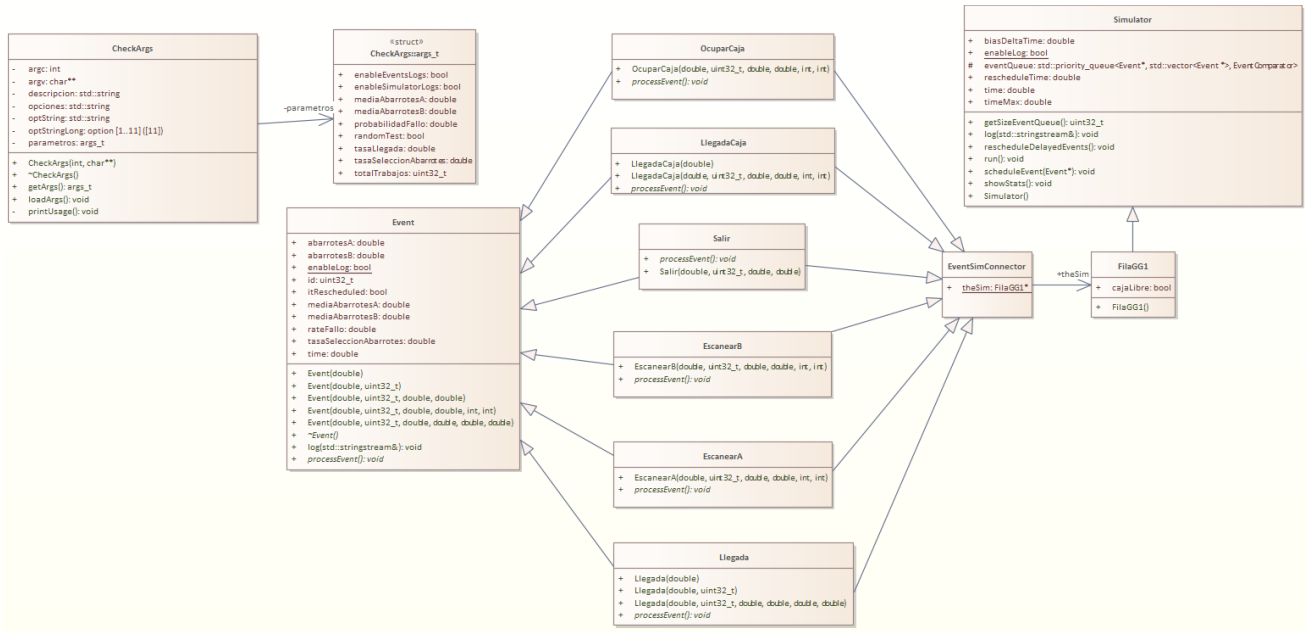

Figura 2. Diagrama de clases UML de la solución implementada.

También se añadió un nuevo método al simulador para mostrar las estadísticas finales. Las cuales incluyen el tiempo final de simulación (en segundos y en minutos), la cantidad total de abarrotes de tipo A vendidos, al igual que los de tipo B y la cantidad de trabajos realizados.

Tabla 1. Métodos del simulador

Método	Descripción
Llegada	Procesa la llegada de un cliente al minimarket. Calcula el tiempo que el cliente se toma en seleccionar abarrotes utilizando una distribución exponencial. También determina la cantidad de abarrotes tipo A y tipo B que el cliente lleva, empleando una distribución normal.
LlegadaCaja	Gestiona la llegada de un cliente a la fila de la caja. Verifica si la caja está libre y, de ser así, pasa al cliente a la caja; si no, reprograma el evento para un tiempo posterior.
OcuparCaja	Procesa el evento cuando un cliente ocupa la caja, y determina el siguiente proceso basado en los abarrotes que lleva el cliente.
EscanearA/EscanearB	Estos métodos manejan el escaneo de abarrotes tipo A y B, respectivamente. Calculan el tiempo que toma escanear cada artículo y manejan posibles fallos durante el proceso.
Salir	Maneja el evento de salida del cliente, liberando la caja y planificando la reprogramación de eventos pospuestos.

[1] ***qrate*** se refiere a la tasa de tiempo que demora un cliente en escoger abarrotes
[1] ***fail*** se refiere a la probabilidad de fallo del escáner. A menor valor, menos probabilidad de que falle.
[1] ***tipoa*** y ***tipob*** son las cantidades medias que lleva cada cliente según tipo de producto.

Esta solución se puede encontrar en el repositorio de GitHub [1] correspondiente.

3. Resultados

Para utilizar el simulador se utiliza el comando explicado en la **Figura 3**.

```
faith@PC-Angel:/mnt/d/GitHub/tarea03-ICI515/simulador$ ./example
Uso: ./example --jobs <total jobs> --rate <rate> --qrate <quantityrate> --fail <failrate> --tipoa <rate> --tipob <rate> [--test][--simlogs][--eventslogs][--help]
Descripción:
        --jobs     cantidad total de trabajos a simular (integer).
        --rate     tasa de llegada de trabajos (cada da 'rate' unidades de tiempo llega un elemento al sistema).
        --test     genera archivo de pruebas de nros aleatorios y termina.
        --qrate    tasa de tiempo promedio de seleccion de abarrotes.
        --fail    probabilidad de fallo en el escaneo de abarrotes A (0 - 100).
        --tipoa    media de cantidad a comprar abarrotes tipo A.
        --tipob    Media de cantidad a comprar abarrotes tipo B.
        --slogs    habilita logs del simulador en pantalla.
        --elogs    habilita logs de los eventos en pantalla.
        -h   Muestra esta salida y termina.

faith@PC-Angel:/mnt/d/GitHub/tarea03-ICI515/simulador$
```

Figura 3. Uso del simulador desde la línea de comandos.

3.1. Ejemplos de ejecución:

Al agregar los argumentos de la **Tabla 2** se obtienen los resultados vistos en la **Figura 4** y **Figura 5:**

Tabla 2. Tabla de valores de parámetros

Parámetro	Valor
Jobs	3
rate	5
crate	20
fail	10
tipoa	10
tipob	5

[1] ***qrate*** se refiere a la tasa de tiempo que demora un cliente en escoger abarrotes
[1] ***fail*** se refiere a la probabilidad de fallo del escáner. A menor valor, menos probabilidad de que falle.
[1] ***tipoa*** y ***tipob*** son las cantidades medias que lleva cada cliente según tipo de producto.

```
faith@PC-Angel:/mnt/d/GitHub/tarea03-ICI515/simulador$ ./example --jobs 3 --rate 5 --qrate 20 --fail 10  --tipoa 10 --tipob 5 --slogs --elogs
Fila de atención simple
Agregando en la FEL evento id=0, timeArrive=3.28478
Agregando en la FEL evento id=1, timeArrive=6.18978
Agregando en la FEL evento id=2, timeArrive=6.43394
3.284781        id: 0   ==> llega al minimarket.
==> id: 0 se toma en seleccionar 8.55647 segundos.
Lleva: 29 abarrotes de tipo A.
       12 abarrotes de tipo B.

6.189779        id: 1   ==> llega al minimarket.
==> id: 1 se toma en seleccionar 6.76487 segundos.
Lleva: 13 abarrotes de tipo A.
       4 abarrotes de tipo B.

6.433939        id: 2   ==> llega al minimarket.
==> id: 2 se toma en seleccionar 23.4076 segundos.
Lleva: 6 abarrotes de tipo A.
       13 abarrotes de tipo B.

11.841246       id: 0   ==> llega a la fila de la caja.
11.841246       id: 0   ==> pasa a la caja.
11.841246       id: 0   ==> Llega a la caja con tiempo:11.8412
11.841246       id: 0   ==> Fallo en el abarrote. +29 segundos
12.954652       id: 1   ==> llega a la fila de la caja.
12.954652       id: 1   ==> caja ocupada, replanificado para t=1012.954652
29.841536       id: 2   ==> llega a la fila de la caja.
29.841536       id: 2   ==> caja ocupada, replanificado para t=1029.841536
40.841246       id: 0   ==> Se escanea abarrote. +7 segundos
```

Figura 4. Resultado del simulador con los valores presentados en la **Tabla 1.**

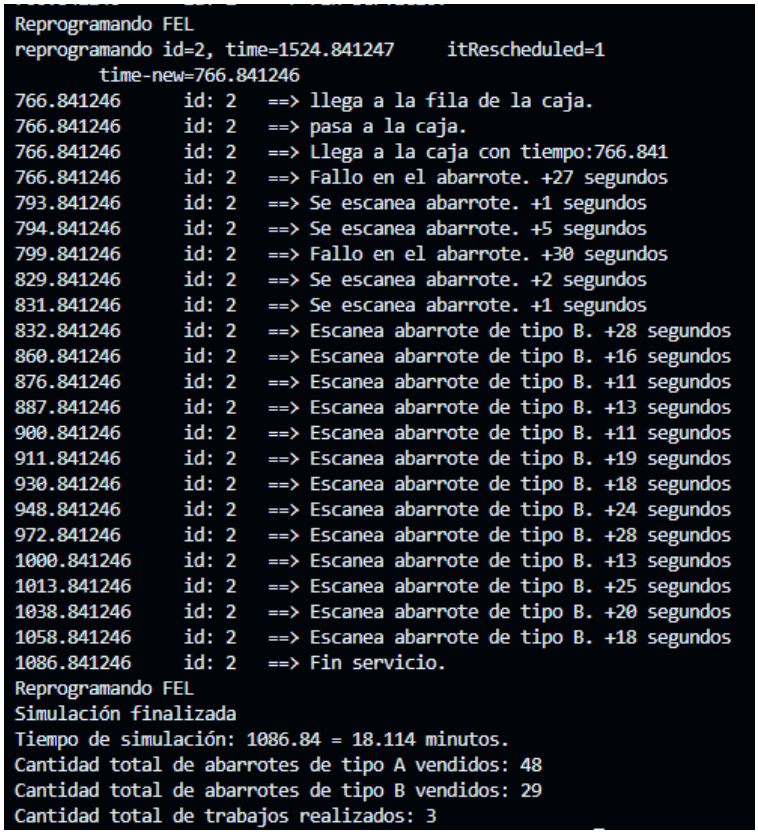

Figura 5 Resultado del simulador con los valores presentados en la **Tabla 1.**

4. Discusión y conclusiones

En este informe se revisó un modelo desarrollado anteriormente y se comprobó la posibilidad de implementarlo en un lenguaje de programación para su uso. Esto dio por resultado un simulador completamente funcional y parametrizado. Según los resultados y le materia estudiada, es posible mejorar este modelo, implementando un sistema multi-caja, posibilidades de abandono y mejoras a las probabilidades y estadísticas usadas en el simulador.

Uno de los parámetros posibles a agregar o modificar es el tiempo de escaneo de cada producto, debido a que se pueden obtener tiempos no sujetos a lo esperado en la realidad.

Sin embargo, los resultados obtenidos tienen coherencia y poseen un alto grado de similitud a su contraparte en la realidad.

5. Referencias

[1] https://github.com/angel-salgado-m/ICI-515-Tarea03