Universidad Francisco de Vitoria

UFV Madrid

Práctica Final de Asignatura Nº 1
Process Auditor

Enunciado

Sois contratados por una empresa de auditorías de transacciones bancarias **UFVAudita** para crear una solución que sea capaz de procesar los reportes de las distintas sucursales bancarias y buscar patrones de operaciones en los datos que permitan identificar en los usuarios conductas irregulares o delictivas. Para ello, a lo largo del día, el sistema debe recoger los ficheros de operaciones que distintas sucursales envían para informar sobre sus actividades bancarias online.

Se os pide **diseñar** y **desarrollar** una solución que actúe como encargada de "***procesar***" dichos ficheros y que escale sin problema a un número creciente de sucursales bancarías.

Actualmente la empresa recibe operaciones de cuatro sucursales que se unen a la lucha contra la actividad delictiva. Cada sucursal enviará los siguientes ficheros de forma aleatoria durante el día:

- SU001_OPE001_DDMMAAAA_No.data
  - Listado de operaciones Tipo 01 online durante el día.
- SU002_OPE002_DDMMAAAA_No.data
  - Listado de operaciones Tipo 02 online durante el día.
- SU003_OPE003_DDMMAAAA_No.data
  - Listado de operaciones Tipo 03 online durante el día.
- SU004_OPE004_DDMMAAAA_No.data
  - Listado de operaciones Tipo 04 online durante el día.

Tenga en cuenta que de cada sucursal se pueden recibir varios reportes durante el día, de ahí que el "No" en el nombre del fichero señale el consecutivo del fichero en un día concreto.

Los ficheros que enviarán las casas de apuestas tienen la siguiente estructura:
IdOperacion;FECHA_INICIO;FECHA_FIN;IdUsuario;IdTipoOperacion;NoOperación;Importe; Estado

OPE001	02/02/2022 15:25	02/02/2022 15:50	USER001	COMPRA01	1	+150.0€	Error
OPE002	02/02/2022 15:25	02/02/2022 15:50	USER001	COMPRA02	1	+200.0€	Finalizado
OPE003	02/02/2022 15:25	02/02/2022 15:50	USER001	COMPRA01	1	+100.0€	Correcto
OPE003	02/02/2022 15:25	02/02/2022 15:50	USER001	COMPRA01	2	-108.0€	Error
OPE003	02/02/2022 15:25	02/02/2022 15:50	USER001	COMPRA01	3	-215.0€	Correcto
OPE006	02/02/2022 15:51	02/02/2022 16:30	USER001	COMPRA02	1	-220.0€	Correcto
OPE006	02/02/2022 15:51	02/02/2022 16:30	USER001	COMPRA02	2	-355.0€	Correcto

Tabla I – Estructura propuesta para los ficheros del sistema

Al final del día, en una hora concreta se debe lanzar el proceso encargado de identificar los patrones sobre el conjunto total de datos consolidados. Para que la solución sea considerada como válida para la práctica de la asignatura, esta debe cumplir los requisitos y las especificaciones que se describen a continuación.

Especificaciones

1. Vuestra solución, en adelante *FileProcessor*, debe ser parametrizable mediante un **fichero configuración llamado "fp.conf"**.
2. La solución debe procesar todos los ficheros que se depositen de **forma aleatoria** y sin previo aviso en un **directorio común**.
3. **Se debe contar con un número de hilos disponibles igual o superior al número de sucursales que se unan al programa de auditoría.**
4. El directorio común debe especificarse en el fichero de configuración.
5. *FileProcessor* estará siempre **activo** (en ejecución) y debe disponer de un **pool de procesos ligeros (hilos)** que se encarguen de atender los ficheros que vayan llegando al directorio. Los

hilos deben estar en ejecución si hay ficheros o en espera si los ficheros no están en el directorio.

6. En un momento dado, se debe garantizar que se procesarán ***n* ficheros** a la vez, dónde ***n*** corresponde al total de sucursales con ficheros disponibles. El objetivo es garantizar que todas las sucursales serán atendidas al tiempo. *Es decir, si hay más de un fichero de una misma sucursal, al menos uno de ellos estará siendo procesado mientras que los otros esperan.*
7. Los ficheros se procesarán una única vez por un único proceso.
8. Todas las instrucciones *printf* o las salidas por pantalla, también deben salir vía fichero de Log. Se pide por ello crear un **fichero de log** donde se escriba la información general de cada fichero procesado (ver Tabla II). Crea una librería en C con el método que reutilices para imprimir por pantalla y escribir en fichero el fichero de Log.
9. Cada vez que llegue un fichero nuevo al directorio, la recepción de este debe mostrarse en pantalla y escribirse en el fichero de log. Usar un mensaje creativo basado en * u otro símbolo.
10. Se pide crear y mantener actualizado un fichero llamado **consolidado** donde se reflejen **todas las operaciones** leídas de los ficheros. El formato de este fichero debe ser CSV.
11. Debe ser posible configurar por parámetro en el fichero de configuración el número máximo de ficheros a tratar al tiempo.
12. Crea un **proceso NO ligero** que sirva como **Monitor** para mostrar (*printf* por consola) los casos identificados como posibles patrones de comportamiento fraudulento o anómalo. La comunicación con este proceso debe ser mediante tuberías.
13. Los patrones de comportamiento serán **procesos ligeros** que se especialicen en buscar dichos patrones. La solución incorpora 5 patrones bien definidos, pero debe ser posible añadir más patrones a la solución para que sean reportados por el **Monitor**. Los patrones solo deben reportar al proceso que los lance, mientras que el **Monitor** debe ser totalmente independiente de la lógica de la solución.
14. **Utilice mecanismos de sincronización** (semáforos) entre procesos necesarios para garantizar el correcto funcionamiento del sistema (utilice al menos dos tipos de semáforos: de exclusión mutua y semáforos normales).
15. Cada proceso simulará un "*retardo aleatorio entre MAX e MIN segundos*", usando los valores del rango indicado en el fichero de configuración *SIMULATE_SLEEP_MAX=4* y *SIMULATE_SLEEP_MIN=1*.
16. Crea scripts en Linux que generen datos de prueba de cada fichero y tipo de operación. Se recomienda un script para cada fichero, por ejemplo:
    a. **./generateData -l 100 -s SU004 -t COMPRA01 -u 10** (-l No. Líneas, -s Sucursal, -t tipo operación, -u No. Usuarios)
    b. El script debe aceptar parámetros de forma corta y larga.
    c. *FileProcessor* puede ser usado mediante parámetros para ejecutar los script.

Patrones de comportamiento: se consideran irregular los siguientes patrones.

- **Patrón 1:** el número de transacciones de un mismo usuario durante una hora concreta. Si hay más de 5 transacciones en una misma hora para un mismo usuario.
- **Patrón 2:** un usuario realiza más de 3 retiros a la vez durante un mismo día.
- **Patrón 3:** un usuario comete más de 3 errores en un mismo día.
- **Patrón 4:** un usuario realiza una operación de cada tipo en un mismo día.
- **Patrón 5:** la cantidad de dinero retirado (-) en un día concreto es mayor que el dinero ingresado (+) ese mismo día.

Universidad Francisco de Vitoria

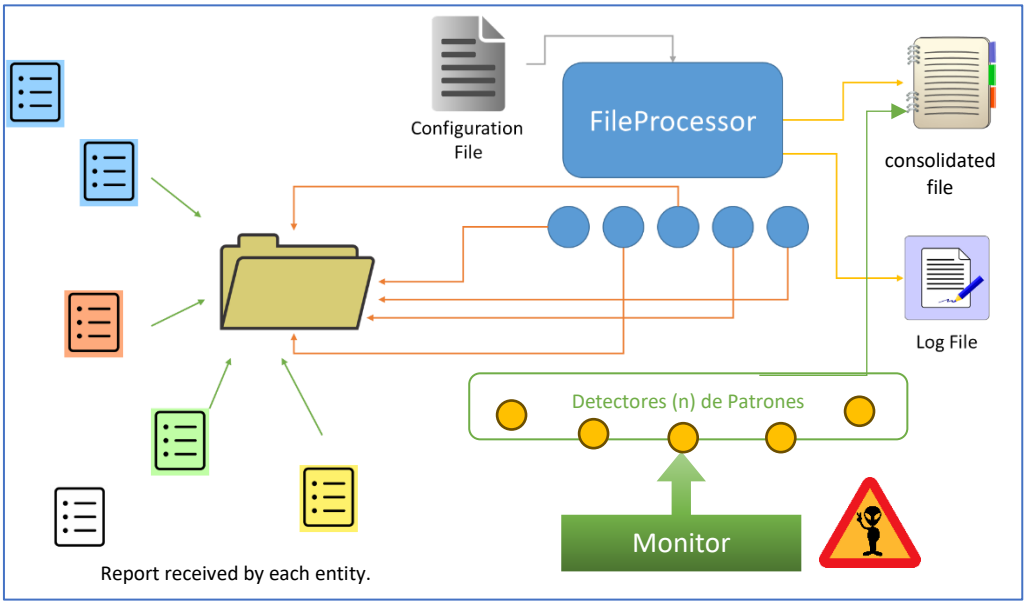
UFV Madrid

Práctica Final de Asignatura Nº 1
Process Auditor

Fichero Consolidado: **IdSucursal;IdOperacion;FECHA_INICIO;FECHA_FIN;IdUsuario;IdTipoOperacion;NoOpe;Importe; Estado**
Fichero Log **FECHA:::HORA:::NoPROCESO:::INICIO:::FIN:::NOMBRE_FICHERO:::NoOperacionesConsolidadas**
Fichero Configuración: **PATH_FILES=C:/mi_ruta** **INVENTORY_FILE=file_001.csv** **LOG_FILE=file_log.log** **NUM_PROCESOS=5** **SIMULATE_SLEEP=1**

Tabla II – Estructura propuesta para los ficheros del sistema

Esquema general del problema propuesto



Entrega Final

- (40% de la nota) **Memoria de la práctica en PDF.** La memoria debe incluir:
  - Elementos comunes de todo trabajo (portada, índice, apartados organizados, etc.)
  - Explicación de la solución.
  - Esquema generar de la aplicación.
  - Capturas de su funcionamiento.
- (10% de la nota) **El plan de prueba de la aplicación.**
- (30% de la nota) **El código fuente de la solución funcionando.**
- (20% de la nota) **Defensa y presentación de la aplicación.**

Universidad Francisco de Vitoria

UFV Madrid

Práctica Final de Asignatura Nº 1
Process Auditor

Evaluación mediante revisión presencial en clases:

- Un grupo al azar debe preparar la presentación de su solución, ejecutar el código e ir explicando el plan de pruebas que haya realizado.
- El profesor durante la exposición revisará la memoria e irá haciendo preguntas sobre la solución desarrollada.
- El profesor podrá pedir modificar la práctica (cambiar algo en la configuración, por ejemplo) y el equipo tendrá que demostrar que conoce su práctica y sabe como mostrar su ejecución real.
- En el caso de que la defensa se haga en público, los grupos presentes en la presentación podrán preguntar, aportar críticas o sugerencias sobre la solución.

Consideraciones IMPORTANTES:

a) El código debe compilar sin errores y funcionar con la configuración que indique el fichero.
b) No se aceptan mensajes de "warning" al compilar.
c) El código debe ir correctamente documentado y/o comentado, explicando qué hace cada función desarrollada.
d) El código debe estar optimizado, sin redundancia de código, bucles innecesarios o condiciones mal definidos (*if repetidos por todo el código*).
e) Se valora el uso de una interfaz que le indique al usuario de qué trata la aplicación.
f) En la memoria, utiliza capturas de pantalla donde se muestre el funcionamiento del sistema.

Organización del equipo de trabajo:

El equipo debe nombrar responsables del **diseño**, **desarrollo** y **pruebas** de la solución. **Esta organización no debe impedir que todo el equipo conozca el funcionamiento de la aplicación en general**.

- ✓ **Responsable de diseño:** es quien define la estrategia de paralelización y concurrencia, así como el esquema general de la solución. Dicha estrategia tendrá que presentarla al profesor.
- ✓ **Responsable de pruebas**: es el encargado de la generación de datos de prueba y de la definición del **Plan de Prueba** de la aplicación. Además de revisar con el profesor la estructura de los ficheros, también debe encargarse de que el equipo procese correctamente los ficheros. También es responsable de establecer el Plan de Prueba de la aplicación.
- ✓ **Responsable de desarrollo:** todo el equipo.