Proyecto No. 2

Realizar: un manejador de archivos.
Realizarse: en grupo de 3 o 4 estudiantes**.** Es el mismo grupo utilizado para la investigación corta no. 2.
Fecha de entrega: lunes 30 de noviembre.

Objetivos:
a. Utilizar patrones de diseño en Java.
b. Practicar el uso de estructuras de datos.
c. Empleo de herramientas de builds automáticos y seguimiento de problemas (bug tracking).
d. Seguir una metodología de desarrollo de software.

Programa a realizar:
Desarrollo de un programa manejador de archivos. El programa recibe la definición lógica de un grupo de archivos, expresado en un documento XML. Con base a dicho documento crea el conjunto de archivos físicos en disco necesarios implementar la definición lógica de ellos.

Por ejemplo, se podría definir un archivo que tenga los siguientes campos: a)carnet, carrera, nombre, apellido (todos tipo string). b)indicando que el campo carnet es llave primaria, c)que existe un índice por carrera, d)existe un índice por apellido.

NOTA: los tipos de datos de los campos del archivo podrán ser cualquier tipo de dato primitivo de Java, como string, int, double, etc.

Eso genera los siguientes archivos en disco: a)archivo de acceso directo (random acsess files) con los registros del alumno, con cada registro en la posición física que le corresponde según el hash de su llave primaria (carnet). b)un archivo que contiene el árbol 2-3 con la carrera y apunta hacia la posición física del registro de alumnos. c)un archivo que contiene el árbol 2-3 con el apellido y apunta también hacia la posición física del registro del alumno.

Usaremos también la arquitectura Cliente/Servidor. El Servidor escuchará las solicitudes de operación con los archivos que hace uno o más clientes y devolverá los resultados correspondientes al cliente que lo solicitó.

Las solicitudes de operación con los archivos pueden ser ingresar, modificar o eliminar datos en ellos. Estas solicitudes son expresadas también en XML.

Para la comunicación entre los Clientes y el Servidor se propone el uso de sockets. Para su utilización debe emplearse java.net

El Servidor tiene una cola en la que coloca las solicitudes enviadas por cada Cliente y conforme las va procesando manda las respuestas al Cliente que ha efectuado la solicitud. Cada Cliente tendrá las solicitudes que desea hacer al Servidor almacenadas en un archivo y las respuestas que reciba serán también guardadas en otro archivo. Cada Cliente hará una solicitud a la vez y espera la respuesta, previo a mandar su siguiente solicitud.

Se necesita tener un monitor para ver las solicitudes que se encuentran en la cola del servidor. Debe indicar la hora y minutos de su recepción, a que Cliente pertenece y el texto de la solicitud. Además el Servidor tendrá un log indicando: a)Cliente que hizo la solicitud, b)hora/minutos en que se recibió la solicitud, c)hora/minutos en que se envió la respuesta, d)texto de la solicitud.

Los Clientes pueden ser procesos que están corriendo en la misma computadora y usando la misma JVM. Opcionalmente, si el grupo lo desea, el Cliente puede estar corriendo en otra computadora u otra JVM.

Tareas:
a. Planificar el desarrollo conforme a la metodología XP. Indicar las etapas, roles de los desarrolladores, etc. que se usarán en el proyecto.
b. Diagramas UML que permitan conocer la estructura del sistema Cliente/Servidor (Clases, estados, secuencias, etc.)
c. Control de versiones de todos los documentos y código que seleccione el grupo.

d. Una colección de pruebas unitarias de las principales operaciones del sistema de archivos.
e. Documentos de planificación (Gantt y otros), así como las variaciones a la planificación. Debe mostrar explícitamente que se está empleando una metodología XP. Deben mostrarse los avances semanales y las razones de sus variaciones.
f. Cada semana se debe mostrar en clase los avances realizados. Serán breves exposiciones de 5 minutos por grupo, mostrando productos y posibles riesgos para la próxima iteración.
g. El programa debe utilizar por lo menos un patrón de diseño.
h. Deberá utilizarse una herramienta de build automático para mantener la versión ejecutable más reciente del sistema Cliente / Servidor.
i. Debe contarse con un área de seguimiento de errores (bug tracking) con los errores descubiertos hasta el momento y su situación (asignados, pendientes, solucionados, etc.)
j. Videos demostrativos de todo el proceso de desarrollo y del sistema Cliente / Servidor funcionando.

Calificación: su programa debe funcionar para ser calificado.

Aspecto	Puntos
Estilo de codificación: comentarios, identación, nombres de variables significativas. Documentación generada con Javadoc.	10
Diagramas UML.	10
Casos de prueba en JUnit. Monitoreo de uso de recursos con profiler.	10
Uso de patrones de diseño (mínimo un patrón)	10
Uso de la metodología XP: planificación, roles, productos por iteración.	10
Uso de builds automáticos y bug tracking	10
Uso del repositorio del SCM.	5
Video que muestre el funcionamiento completo del programa y del proceso de desarrollo	5
Funcionamiento del sistema Cliente/Servidor	30
EXTRA: los Clientes corren en una computadora o JVM diferente a la del Servidor.	30
TOTAL:	130

Referencias:
http://home.cogeco.ca/~ve3ll/jatutor9.htm Tutorial de manejo de archivos. Incluye los archivos de acceso directo.

http://publib.boulder.ibm.com/infocenter/iseries/v6r1m0/index.jsp?topic=/rzaha/interpro.htm Ejemplo de comunicación entre procesos empleando sockets.