7506

Organización de Datos

Lic. Servetto

Trabajo práctico

1er Cuat. 2010

Marco:

Los sistemas de archivos han mostrado una gran evolución en los últimos 10 años en función de implementar nuevas estructuras de datos. Este proyecto se trata de aplicar algunas mejoras de ese tipo a un sistema de archivos que no las posee. La forma técnica de lograrlo es a través de un Buffer-Caché que intercederá en todos los accesos a disco y buscará los datos en su almacenamiento, físico, con las estructuras mencionadas.

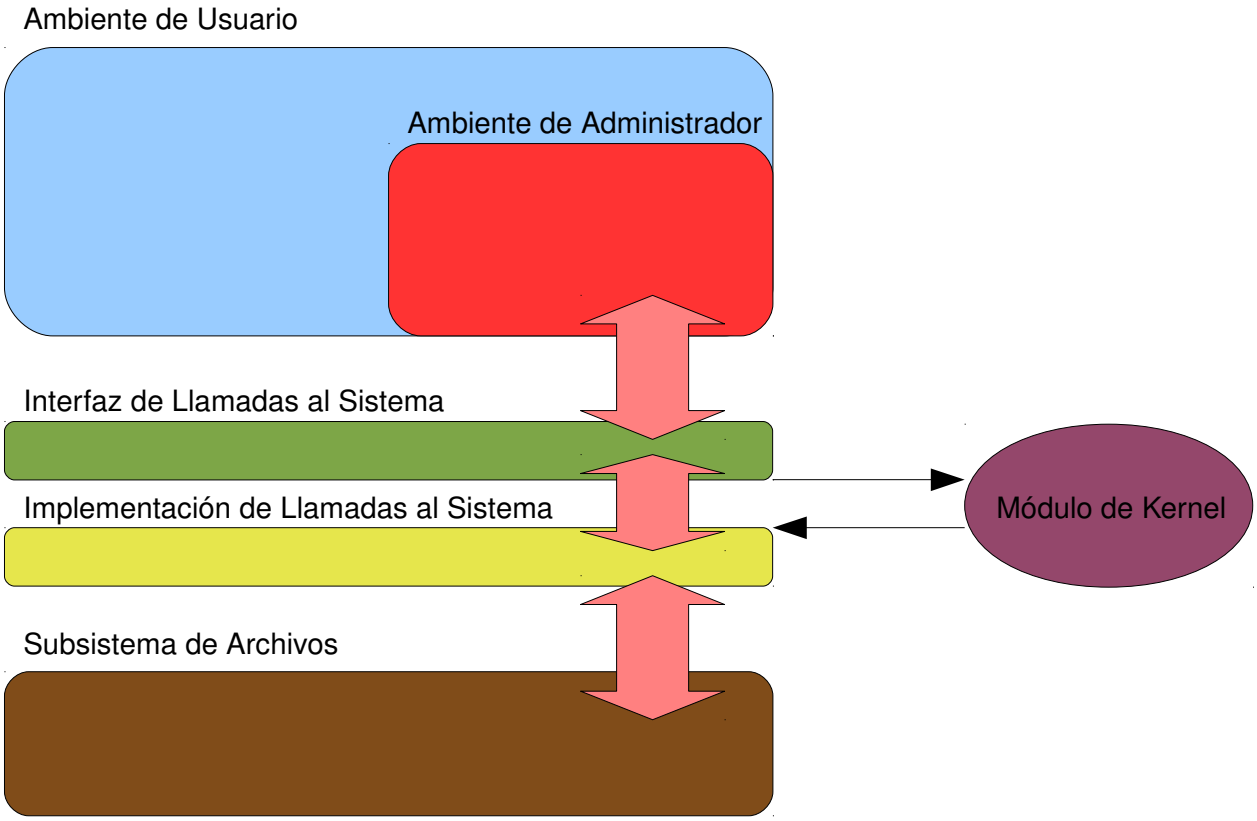
Objetivo:

Se pretende aplicar los conocimientos de Organización de Archivos, Sistemas de Archivos y Compresión de Datos en una implementación de un sistema de buffer-caché. En particular, que se apliquen las estructuras de datos de Hashing Extensible y Árbol B+; además que utilicen y extiendan las Llamadas al Sistema (System Calls) de accesos a almacenamientos del Sistema Operativo, que se apliquen políticas de liberación de recursos de almacenamiento y se implemente un compresor estadístico para agregar una nueva funcionalidad al Sistema de Archivos.

Herramientas:

Se utilizarán los lenguajes C y C++ (a elección, donde el Sistema Operativo lo permita). Sobre una arquitectura Linux, kernel 2.4, Debian Sarge. El desarrollo se soportará por herramientas basadas en Linux (cualquier distribución).
La compilación se realizará a través de un Makefile con el compilador GCC.

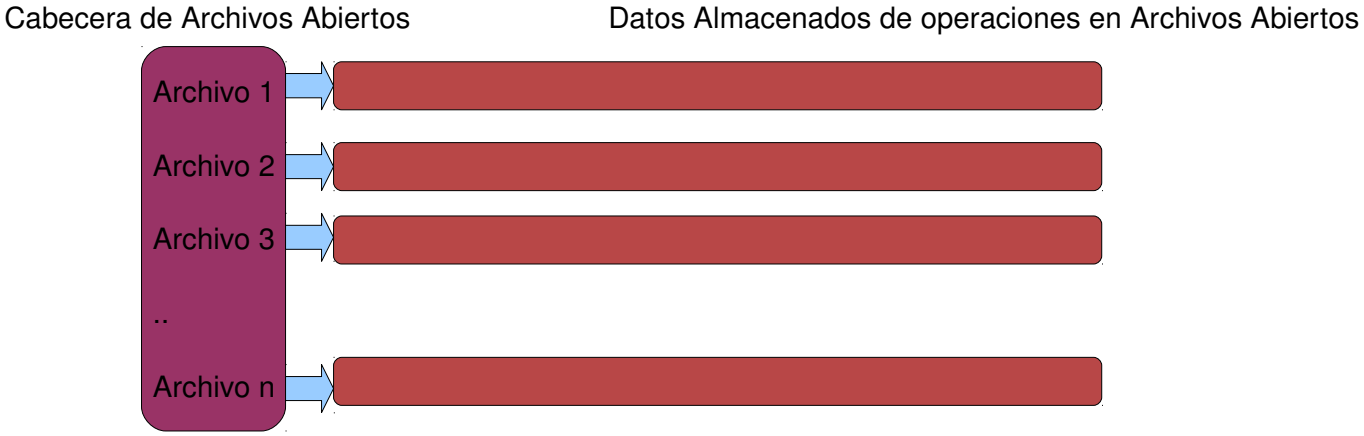
Descripción de la funcionalidad general esperada:



El Ambiente de Usuario es donde corren las aplicaciones que utilizamos, como los programas que hacemos y el Gnome, Desde este lugar se llaman, a través de las librerías de lenguaje o de las Llamadas al Sistema (System Calls), a los servicios que ofrece el Sistema Operativo (procesamiento, almacenamiento, comunicaciones, etc.). Y, dependiendo de qué servicios se utilicen, el Sistema Operativo utilizará su Subsistema de Archivos.

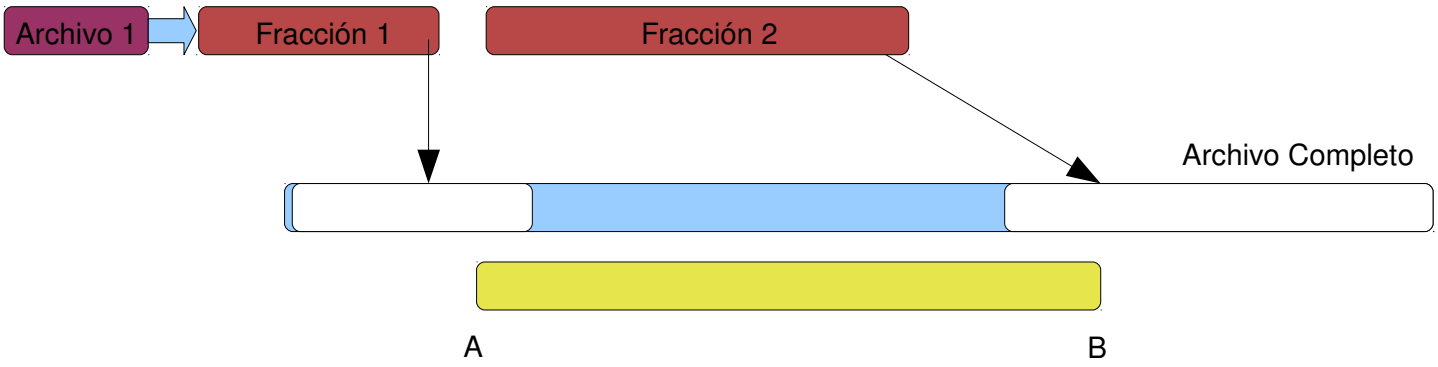
Nosotros programaremos un Módulo del Kernel que es una aplicación a la que se le permite registrarse en el corazón del Sistema Operativo, el Kernel, y funciona como parte de él desde ese momento y hasta que se des-registre. Es una manera que tiene el Sistema Operativo de extender su funcionalidad central.

En nuestro caso el Módulo implementado, interferirá las llamadas al sistema de almacenamiento y aplicará un sistema de Buffer-Caché para acelerar la búsqueda de datos y registrar las operaciones realizadas sobre esos datos.



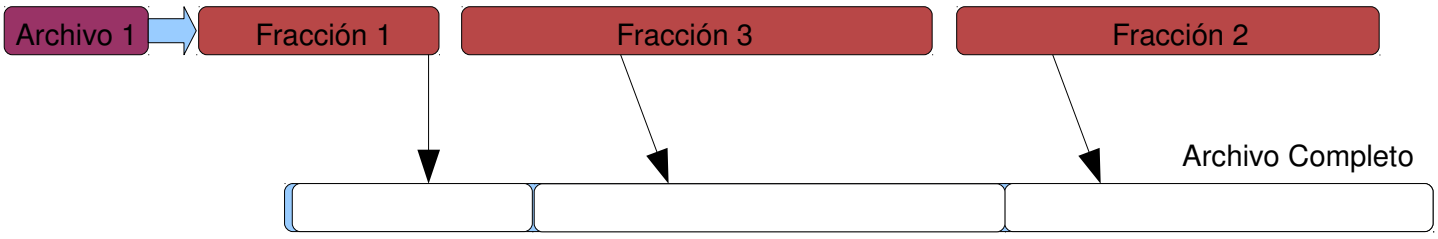
El Buffer-Caché es una estructura que sirve para almacenar los datos que se leen desde el almacenamiento persistente en un almacenamiento temporal, con estructuras que le permiten un acceso más rápido, y que mantiene una estructura de datos a tal efecto.

Se representan en él un grupo de archivos abiertos y sus fracciones de datos leídas. Llevando control del espacio ocupado y de liberación del mismo a través de un algoritmo específico.



Tenemos las Fracciones 1 y 2 en memoria que cubren la parte del Archivo 1 marcada en blanco. Pero se solicita leer el espacio cubierto que va desde el byte A hasta el B, que cubre parte de cada una de estas fracciones y los datos que están entre ellas.

Entoces se identifica qué parte de los datos solicitados ya están en el Buffer-Caché y cuáles no. Buscando sólo los que no están y armando una fracción nueva con cada segmento contiguo de datos que se lean, en este caso la fracción 3.



La funcionalidad de Compresión de Datos queda a definir.

Descripción de la primer entrega:

Se entregarán los fuentes, el makefile y la documentación correspondiente a:
• Una estructura de Hashing Extensible con claves numéricas y datos de tipo indefinido a priori y tamaño variable.
•Una estructura de Árbol B+ con claves numéricas y datos de tipo indefinido a priori y tamaño variable.
•Una aplicación de registro de transacciones en formato texto plano (Logger). Esta aplicación escribirá en archivos de texto plano y al llegar a un tamaño definido se cerrará el mismo, se renombrará para agregarle un número secuencial a final del nombre (ejemplo: logger.txt pasa a ser logger1.txt porque es el primer corte) y se abrirá un nuevo archivo con el nombre del originarl (ejemplo: logger.txt) resultando una acumulación de archivos de registro.

Estas estructuras se operarán a través de aplicaciones que se ejecuten desde consola con los siguientes parámetros.
•-B = Buscar una clave o cadena de caracteres (para el logger) en la estructura o archivo.
•-I = Ingresar datos a la estructura.
•-M = Modificar un dato ya existente en la estructura.
•-Q = Quitar un dato de la estructura.
•-S = Volcar todos los datos de la estructura en un archivo de texto plano para que pueda ser analizado en una observación inmediata (sin datos en binario).
•-h = Ayuda para la operación con la aplicación.

Hashing y Árbol B+ implementan todas las operaciones. Logger implementa -B, -I, -S y -h

Se tomarán los datos por la Entrada Estándar (Standar Input), para las claves se utilizará la siguiente nomenclatura: (clave;valor) y se pueden hacer listas de estas unidades para operaciones masivas. La ejecución está acompañada del nombre del archivo que almacena la estructura, como primer parámetro, de la operación a realizar, como segundo y de la asignación por Entrada Estándar de los valores.

Ejemplos: btree path-nombreArchivo -B < (12;)
cat archivo_de_claves | btree path-nombreArchivo -B
obs: “< (12;)” y cat archivo_de_claves |” son comandos para ingresar valores por stdio.

Especificaciones de los parámetros a considerar dentro del archivo principal (main) de la aplicación:
•Tamaño de Bloques (sean nodos o no) en relación a 515*2^x Bytes.
•Máxima longitud del archivo del Logger para el cuál se cerrará y se creará uno nuevo.

Descripción de la segunda entrega:

Se proveerá al grupo de una máquina virtual, bajo tecnología VirtualBox, para que compile su aplicación e instale el módulo, será el ambiente de pruebas, en el que se realizará la corrección también.
– Se completará ---