Programación de sistemas distribuidos

Práctica 1: Objetos remotos

En esta primera práctica se suministra un programa con la siguiente funcionalidad:

File Manager

Este programa sirve para listar los archivos de un directorio local llamado “FileManagerDir”, y realizar funciones básicas de lectura/escritura sobre ellos. Está compuesto por los siguientes archivos:

- fileManager.h: Esta clase implementa las funciones de listado de un directorio local al ejecutable que contenga esta clase, pasado por parámetros al constructor. Tiene las siguientes funciones:
  - ListFiles : Devuelve un vector con los nombres de los ficheros encontrados en el directorio local. OJO, no es una lista dinámica, no se actualiza automáticamente si se modifican los archivos, si se escriben/crean nuevos ficheros no se añadirán automáticamente, hay que volver a invocar la función para obtener la lista actualizada.
  - ReadFile: Si se le pasa el nombre de uno de los ficheros encontrados en el directorio (lista devuelta por “ListFiles”), leerá su tamaño y contenido. Lo almacenará en las variables pasadas por parámetros.
  - WriteFile: Análoga al anterior, escribirá el contenido fichero en el directorio local. Si el fichero no existía en el directorio, se añadirá a la lista de ficheros (hay que volver a llamar a “ListFiles” si se quiere mantener la lista actualizada).
- LibFileManager.a: Este archivo contiene la implementación de la clase mencionada anteriormente. Ya está compilado y el proyecto “CMakeLists.txt” está preparado para enlazarlo con el ejecutable.
- Main_fm.cpp: Programa principal de prueba. Inicia una clase de tipo “fileManager” y ofrece un terminal para escribir comandos. Tenéis los siguientes comandos:
  - ls: lista los ficheros locales al programa
  - lls: lista los ficheros locales a la clase “fileManager”. Serán los ficheros que se encuentren en el directorio pasado por parámetros al constructor de esa clase.
  - upload: Este comando copia un archivo local al ejecutable al directorio local a la clase fileManager (el pasado por parámetros al constructor). Este comando recibe un parámetros adicional:
    - Nombre del archivo “origen”, que se encuentra en el directorio local al ejecutable.

    EJ:

    Si se ejecuta el siguiente comando, se copiará el archivo “a.txt” al directorio local a fileManager.

    "upload a.txt”

- exit(): Este comando cierra la terminal y acaba la ejecución del programa.

- download: Este comando copia un archivo local a la clase fileManager (el directorio pasado por parámetros al constructor) al directorio local del ejecutable. Este comando recibe un parámetros adicional:
    - Nombre del archivo "origen", que se encuentra en el directorio local a fileManager.

  EJ:

  Si se ejecuta el siguiente comando, se copiará el archivo "a.txt" al directorio local al ejecutable.

  "download a.txt"

Objetivos:

Se pide distribuir la clase fileManager, de tal manera que sea lo más transparente posible para el programa "main" suministrado. No se pueden modificar los archivos suministrados con esta práctica. El alumno deberá implementar como mínimo dos programas, pueden estar todos los programas en el mismo ordenador usando "localhost" y diferentes puertos:

- Programa 1 (servidor): Actuará como servidor de objetos de tipo "fileManager", ofreciendo la implementación remota de las clases anteriores.
- Programa 2 (cliente): Actuará como cliente, sus programas harán las llamadas necesarias a las clases remotas contenidas en el servidor del programa 1.

Ejercicio extra:

- Crear un sistema "Broker de objetos" para encontrar servicios de un tipo dado y balancear conexiones:
    - Los programas de tipo servidor de fileManager pueden encontrarse en un ordenador desconocido, por lo que el cliente necesitaría un sistema de búsqueda de servicios. Además, para equilibrar las conexiones entre varios ordenadores, puede haber varias instancias del programa servidor en el sistema.
    - Para poder solucionar estos problemas, se pide crear un programa tipo Servidor que funcionará como "Broker de objetos", el cual tendrá la siguiente operación:
        - Esperará a que se conecten programas clientes o servidor de fileManager:
            - Si se conecta un servidor, se deberá identificar con si IP y el Broker lo almacenará
            - Si se conecta un cliente, se le devolverá una IP almacenada de alguno de los servidores conectados anteriormente, y se apuntará esa conexión. Cada nuevo cliente que se conecte se le intentará dar el servidor con menos conexiones.
        - Los programas cliente/servidor funcionarán de la siguiente manera:

    - Programa servidor: Inicia una conexión con el broker y le envía su IP de red privada. A continuación queda esperando conexiones normales de clientes.
    - Programa cliente: Antes de iniciar una conexión con el servidor, conectará con el Broker para pedirle una IP de algún servidor que esté encendido.
    - Ambos programas cliente/servidor pueden conocer tanto su propia IP como la del Broker. El cliente sólo conoce al servidor si se lo da el Broker y viceversa.
- Para poder probar esta parte se pide crear dos máquinas más, una para el servidor de fileManager y otra para el programa Broker.

Entrega

Las prácticas son individuales, se debe entregar el código desarrollado a través de blackboard, y realizar un examen en el que se obtendrá la nota final de la misma.

Las prácticas se deben entregar antes del día indicado en la entrega en Blackboard, en caso de entregarlo más tarde se corre el riesgo de no poder presentarse al examen de la práctica. El contenido entregado en blackboard tendrá la siguiente estructura:

- Fichero ZIP con los archivos generados/editados por el alumno. Cada uno de los ficheros entregados DEBERÁ contener el nombre del integrante del grupo escrito en comentarios de código al inicio de cada archivo. Se deja a elección del alumno la estructura de directorios que contendrá su entrega.
- Nombre del fichero **(OBLIGATORIO)**: PR1SISDIS_Alumno1_Apellido1.zip

Cualquier archivo entregado que no se adecúe a estas prescripciones podrá ser ignorado. Por favor, no entreguéis archivos con otros compresores o nomenclaturas.