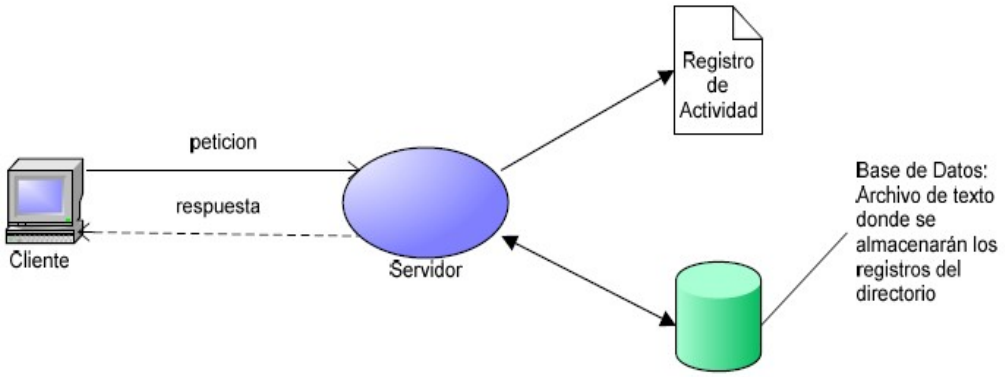
Proyecto del Curso
Introducción a la Programación Orientada a Objetos
Escuela de Ingeniería de Sistemas y Computación
Universidad del Valle

Resumen

Este proyecto consiste en la implementación en C++ de un servicio de directorio básico. Un servicio de directorio es una aplicación que almacena y organiza la información sobre los usuarios de una red de ordenadores. Básicamente, es una aplicación que escucha y responde a peticiones de clientes a través de un puerto de comunicaciones. Esto se conoce como una comunicación cliente-servidor. Para este proyecto, el cliente será un terminal de telnet que enviará peticiones al servidor en el puerto adecuado.

Este documento se organiza así: En la sección 1, se describe la arquitectura deseada; La sintaxis de los mensajes que se intercambiarán el cliente y el servidor se describen en la sección 2; Las secciones 3 y 4 contienen las especificaciones de los archivos de la base de datos y del registro de actividad, respectivamente.

1. **Arquitectura**



Los componentes involucrados en la arquitectura de la aplicación son el cliente, el servidor, los mensajes de petición/respuesta, la base de datos y el registro de actividad. El cliente es quien inicia la comunicación en el puerto que está escuchando el servidor. Una vez establecida la comunicación, el cliente envía mensajes de petición siguiendo la sintaxis descrita en la siguiente sección.

Por su lado, el servidor recibe la petición, la procesa, construye y envía la respuesta de vuelta al cliente. El procesamiento de la petición consistirá en extraer los datos requeridos para realizar la operación requerida en la base de datos (insertar, consultar, borrar). Adicionalmente, se deberán registrar los datos de la petición y de la respuesta en el registro de actividad descrito en la sección 5.

2. **Sintaxis de los mensajes de petición y de respuesta**

Las peticiones y respuestas son los mensajes que se van a intercambiar el cliente y el servidor. Se trata de cadenas de caracteres que cumplen con la siguiente sintaxis o estructura dependiendo de la operación solicitada.

Operación insertar:

- `(peticion :sender <ORIGEN> :content "(service insertar) ( id=<valor>,nombre=<valor>,cargo=<valor>,correo=<valor>,fijo=<valor> ,móvil=<valor>)")`
- `(respuesta :sender <ORIGEN> :content "(service insertar) (id=<valor>,<texto>)")`

Operación borrar:

- `(peticion :sender <ORIGEN> :content "(service borrar)(id=<valor1)")`
- `(respuesta :sender <ORIGEN> :content "(service borrar) (id=<valor>,<texto>)")`

Operación consultar:

- `(peticion :sender <ORIGEN> :content "(service consultar) (<attr1>=<valor1>)")`
- `respuesta :sender <ORIGEN> :content "(service consultar) ( id=<valor>,nombre=<valor>,cargo=<valor>,correo=<valor>,fijo=<valor> ,móvil=<valor>)")`

  ó

- `(respuesta :sender <ORIGEN> :content "(service consultar) (id=<valor>,<texto>)")`

Donde,

- <ORIGEN> es la dirección IP del ordenador en donde se origina el mensaje;
- <valor> es el valor del atributo adecuado;
- <texto> es una de las siguientes cadenas de caracteres según sea el caso:
  - Operación realizada con éxito
  - El registro no existe
  - El registro ya existe
  - Error en el servidor

Ejemplo:

- ***Peticion*** de insersión de un nuevo registro:

```
(peticion :sender 192.168.0.1 :content "(service insertar)
(id=123,nombre=fulanito,cargo=jefe,correo=fulanito@laempresa.com,fijo=1234
56789,móvil=678901234)")
```

- La ***respuesta*** sería algo así:

```
(respuesta :sender 192.168.0.1 :content "(service insertar)
(id=123,operación realizada con éxito)")
```

3. **Base de datos**

Atributo	Tipo	Descripción
id	Cadena de caracteres	Identificador de 3 dígitos o XXX en caso de que se haya marcado con la operación de borrado
nombre	Cadena de caracteres	Nombre completo
cargo	Cadena de caracteres	Cargo en la empresa
correo	Cadena de caracteres	Dirección de correo electrónico
fijo	Cadena de caracteres	Número de teléfono fijo
móvil	Cadena de caracteres	Número de teléfono móvil

Consiste en un archivo de texto en donde se almacenarán los datos personales mostrados en la tabla anterior. Cada línea en el archivo deberá contener los valores de los atributos de la persona separados por comas:

```
id,nombre,cargo,correo,fijo,móvil
```

La inserción de una nueva persona debe hacerse al final del archivo. El borrado, consistirá en reemplazar el id con la cadena XXX. De esta manera, las líneas que comiencen con X no se tienen en cuenta.

4. **Registro de Actividad**

Atributo	Tipo	Descripción
fecha-hora	Cadena de caracteres	fecha y hora de la operación con formato YYYY/MM/DD HH:MM:SS
mensaje	mensaje	texto completo del mensaje de petición o respuesta según sea el caso

Se trata de un archivo donde se deberá registrar los mensajes entrantes y salientes. Cada línea en el archivo debe tener la forma:

```
fecha-hora mensaje
```

5. Entregas

[**30%**] [**28 Abril**] Primera entrega: Análisis y diseño del sistema

[**40%**] [**18 Mayo**] Segunda entrega: Implementación en C++

[**30%**] [**19 de Mayo, 20 de Mayo**] Sustentaciones

NOTAS:

- Realizar en grupos de **3 estudiantes**.
- Comente su código fuente, usando los caracteres de comentario /**/ ó //
- Tabule adecuadamente todas las sentencias de su programa.
- Agregue el siguiente comentario al principio de cada uno de sus archivos de C++:

```
/*
  Asignatura: PROGRAMACIÓN ORIENTADA A OBJETOS (IPOO) 750081M
  Proyecto: Nombre del proyecto
  Archivo:
  Fecha creacion: DD-MMM-AAAA
  Fecha modificacion: DD-MMM-AAAA
  Versión:
  Autor(es):
  ESCUELA DE INGENIERIA DE SISTEMAS Y COMPUTACION
  Universidad del Valle
*/
```