

9 Clase Sockets UDP en C++

Elaborado por: Ukranio Coronilla

Haciendo uso del código en C para construir sockets UDP tratado en el manual “Programación de sistemas Linux”, elaboraremos un cliente y un servidor que realicen la misma tarea, pero con el código orientado a objetos en lenguaje C++, para ello debemos considerar las siguientes restricciones:

a) El **mensaje** que se transmite debe ser una instancia de la siguiente clase (cuya interfaz se debe encontrar en el archivo `PaqueteDatagrama.h` y su implementación en `PaqueteDatagrama.cpp`):

```
class PaqueteDatagrama {
public:
    PaqueteDatagrama(char *, unsigned int, char *, int );
    PaqueteDatagrama(unsigned int );
    ~PaqueteDatagrama();
    char *obtieneDireccion();
    unsigned int obtieneLongitud();
    int obtienePuerto();
    char *obtieneDatos();
    void inicializaPuerto(int);
    void inicializaIp(char *);
    void inicializaDatos(char *);
private:
    char *datos;           //Almacena los datos
    char ip[16];           //Almacena la IP
    unsigned int longitud; //Almacena la longitud de la cadena de datos
    int puerto;            //Almacena el puerto
};
```

El primer constructor se utiliza para crear un paquete de envío tipo datagrama. Sus parámetros son la cadena a enviarse (datos), la longitud de la cadena, la dirección IP de envío y el puerto de envío. El segundo constructor solo crea un mensaje vacío de una longitud determinada para la recepción. En ambos constructores se debe reservar memoria dinámica para los datos con `new`, por lo que el destructor se encargará de liberar dicha memoria con `delete`.

El método `obtieneDireccion` regresa la dirección IP de la maquina a la cual este datagrama se está enviando o desde la cual el datagrama se ha recibido.

El método `obtieneLongitud` regresa la longitud de los datos que serán enviados o la longitud de los datos recibidos.

El método `obtienePuerto` regresa el número de puerto en el host remoto al cual se envía el datagrama o desde el cual el datagrama se ha recibido.

El método `obtieneDatos` regresa los datos almacenados en el miembro `datos`. Los siguientes dos métodos inicializan el puerto y la IP en los datos privados de la clase.

El método `inicializaDatos` inicializa los datos que se almacenan en el miembro `datos`. Los siguientes dos métodos inicializan los datos privados de la clase.

b) La **conexión** se establecerá mediante una instancia de la clase `SocketDatagrama` (cuya interfaz se debe encontrar en el archivo `SocketDatagrama.h` y su implementación en `SocketDatagrama.cpp`):

```
class SocketDatagrama{
public:
    SocketDatagrama(int);
    ~SocketDatagrama();
    //Recibe un paquete tipo datagrama proveniente de este socket
    int recibe(PaqueteDatagrama & p);
    //Envía un paquete tipo datagrama desde este socket
    int envia(PaqueteDatagrama & p);
private:
    struct sockaddr_in direccionLocal;
    struct sockaddr_in direccionForanea;
    int s; //ID socket
};
```

El constructor crea un socket tipo datagrama y su parámetro es el puerto local, el cual no se podrá cambiar en ningún momento posterior. Si el parámetro es cero, el puerto será asignado por el sistema operativo.

Los métodos envían y reciben mensajes hacia, o en el puerto que se haya especificado; y los valores devueltos son los que regresan las operaciones `recvfrom` y `sendto` respectivamente.

Por último tanto cliente como servidor deberán imprimir puerto e IP, de quien les ha enviado un datagrama.

Recomendaciones: Dado que se trata de varios archivos es necesario utilizar un archivo `Makefile` y la utilería `make` para compilar y enlazar todos los programas.

Al crear un objeto `PaqueteDatagrama` será necesario copiar la información que se desea enviar hacia el objeto. Dicha información puede ser de cualquier tipo por consiguiente no se recomienda utilizar `strcpy`, y en su lugar usar `memcpy`.