Application Client-Server with Socket and API of Barkeley

LaTeX

Project Area

English Version

Class 3F

Year 2015/2016

Contents

1 Introduction

We're guys of the third technical informatic institute IISS "Galileo Galilei", Bolzano. We collaborated to design a project area with the purpose to create a client-server application that executes a mathematical calculation. In these pages you'll find the basis for the client-server operation with an understandable language.

2 Network protocols

[1] The protocols are a set of rules used by the two machines to exchange informations and specify what should be showed, how it should be and when it should be.

If the two machines are remote, then we talk about internet protocol. A common feature of Network Protocols is that they are structured in overlapping layers.

The top level runs requests to the bottom level and the same levels to different machines converse with the same protocol. One of the advantages of the network protocols is the design of a layer that is investigate only some aspects of the problem.

Every layers is not depending from the implementation of others layers.

Every layers provides common services of every functions of the top layers.

2.1 ISO/OSI Model

[2] The OSI model is a standard for the network calculators that establishes the logic network architectur. A structure composed from a stuck of network comunication protocols divided into 7 levels. These levels executes the network funtctionality:

Level	Old model	New model
Level 7	Application level	Application level
Level 6	Presentation level	Presentazionl level
Level 5	Session level	Session level
Level 4	Transport level	Message level
Level 3	Network level	Packet level
Level 2	Connection data level	Frame level
Level 1	Physical level (Hardware)	Physical level

- Physical level: Defines the way where the data are physically converted in digital signals in the mass media (electrical pulse, light modulation, etc).

- Data connection level: Defines the interface with the network card and the sharing from the transmission media
- Network level: Allows to managed the addressing and the data routing, di gestire l'indirizzamento e il routing dei dati, i.e. They're sending through the network.
- Transport level: It's commited in the file transport, in their packets split and in the management of some transmissions errors.
- Session level: Defines the opening and the closing of the comunication sessions between the network terminals.
- Presentation level: Defines the format of the data managed by the application level (their represantion, their compression and their coding) indenpendently from system.
- Application level: secure the interface with the applications. It deals then with the closest level with the users, managed directly from some softwares.

2.2 TCP/IP

[2] The two protocols TCP and IP got created by Bob Kahn and Vinton Cerf in the year 1974, and the goal was to allow the interconnection of computer networks like the military ARPANET, SATANET and other technologically networks.

Represents in a way the set of rules of communication on internet and it's based on the idea of IP addressing, that means that provides an IP address on every Network Terminal in order to send data packets.

The TCP/IP model, based on the model, contains 4 levels:

TCP/IP model	OSI model
Application level	Application level
	Presentazion level
	Session level
Transport level (TCP)	Transpor level
Internet Level (IP)	Network level
Connection data level	Network access level
	Physical level

- **Physical**:Includes the physical interface between the transmission device and the middle transmission or the network. Defines the features of the middle transmission, nature signals, transmissions rates and the data encoding scheme.

- **Network Access**: Describes the exchange of data between a node and the network to which it is connected.

  The specific software used in this level it's depending from the type of network that is used.

  It defines the mode of identification of the recipient and the quality of the service.

  The network access level is the first level of TCP/IP, represents the ways to realize a data transmission through a physical network. Is doing these functions:

  1. Sending connection data.
  2. Coordination of data transmission.
  3. The format of the data.
  4. Converting signals (analog/digital)
  5. Checking errors when it arrives.

- **Internetworking**: Describes the transmissions between two nodes to the network that are the packet switching and the not-reliable transmission.

  It defines the format of the pacckets, the global addressing system and the packet routing mechanism.

  The internet level is the most important level given and it's that level that defines data packets and manages the IP addressing.

  All of this allows to the data packet to get send through remote terminals.

  Includes 5 protocols, the most important protocols are IP, ARP and ICMP. **Transport** Describes the comunications between two nodes of network and guarantees the correct order of the pacckets and sends reliable data.

  The transport level provides a logic-reliable channel of comunications called end-to-end for the packets.

  The transport levelallows at the applications that are running around remote terminals to comunicate.

  Contains two protocols that are allowing to the two applications to exchange indipendently data from the lower level.

These protocols are:

- TCP: TCP TCP provides a reliable transport layer oriented to the connection.

  Reliability means that before sending data to the server, there's an execution called handshaking (asks to the client if it's ready to receive it). Connection-oriented means that the client will receive the packets in order. This is used for have more security.

- UDP: UDP provides a transporter service called datagram-oriented (not reliable) and i'ts not oriented to the connection. The packets are invited without asking any permissions to the client in a casual way. This is used for have more speed.

Applications: Contains the required logic for supporting some user application. Provides some network services that are the remote login, file transfer, email and web.

An application program interacts with one of the protocols of transport layer for receive data or sending by asking

. The application level contains network applications that allows the communications thanks to the lower levels.

The software of this level is comunicating thanks to TCP or UDP. The applications of this level have different types, but the most of these are network services, i.e. Some applications provides to the user to secure the interface with the operating system.

2.3 Network application

The network applications are composed by different elements, in executions on different machines, that are working in a indipendent way and they can exchange informations. The applications are communicated and distributed process.

The comunication happens when is using the offered services from the subsystem of comunication.

The cooperation can be implemented according to different models. The most popular model is the client/server.

3 Socket

3.1 What's socket?

[3] *Socket*,in informatic, is an abstraction software, managed by the operating system, that represents a channel of network comunications between a process and resource. For a programmer, a socket is a particular object where you can ready and write data that you want to transmit or receive.

3.2 Functions and uses of Socket

The socket, in the modern operating systems, is needed for use the API Standard, shared through the network. It allows to send and receive data, between remote hosts or local process.

Local socket and remote comunications, are forming a pair, composed by an address and a port of the client and server. Usually the operating systems provide these API for allow to the applications to control and use the sockets of the network.

The protocols used for the implementation of sockets are:

- TCP (Transfer Control Protocol).)
- UDP (User Datagram Protocol).)

Both protocols rely on the IP protocol (Internet Protocol)). The sockets can be implemented with various programming languages like C, C++ and Java.

3.3 Birth of Socket

The origin of sockets dates from 1983, when they were introduce in the BSD (BSD- Berkeley Software Distribution) in the University of Berkeley in California. The Advanced Research Project Agency funded the University of Berkeley for the implementation of TCP/IP in the operating system Unix.

The researches of Berkeley developed the original set of functions that were called "Socket Interface".

3.4 Socket interface

The interface of the applications it's not usually defined inside the protocols of comunications, but it's part of the residents operating systems where their limits is to receive general lines of the protocols. So, the specifications of a certain protocol could suggest, if necessary, an operation that allows to the applications to transfer data, and API of the operating system establish the name of the relative functions and the type of their arguments.

Although this freedome of choice, a lot of operating systems, from Windows to various versions of UNIX, they chosed API also called socket. Many manufacturers of computers have adopted the system of BSD as basis for the development of commercial operating systems, that made the interface socket the standard for the market.

3.5 API Socket

It's said that the applications client server are comunicating through transport protocols. These applications must provide to the protocols a lot of informations, for example the intention to act as a server or as a client, and more details related to the data to transmit or to receive.

In the end, the applications will use the Applications Interfaec (API – Application Program Interface) that defines the operations that these can run.

API defines not only the available instruments for the comunication, but also the difficulty that is found in the coding of a program that use these instruments.

4 Client-Server Application

4.1 Definition

[4] The structure of an informatic system can be the client-server system. A client-server system is a network architecture formed by two types of forms: the client and the server, that generally are execute on different connected machines in the network.

Much easier, the client/server systems are an evolution of systems based on a simple sharing of sources. The presence of a server allows to a number of client to share sources, leaving to the server to manage the access to the sources. All of this avoid the conflicts.

A server is an informatic member that provides services to others members (client) through a Network.

The word server as the word client can be referred as software member. A server perfom necessary operation to realize a service (an example is that manages bank data, manages the update of the data). A server is a strong computer and is able to manage big companies, schools ecc. It's able to command a lot of connected computers in the network or through internet less strong than the client.

A client indicates a component that accesses services or sources of another component, server, for make some operations. In this context we can talk about client relating to the hardware or to the server. Usually the client can manage the user interface of the application and checks the data insert and provides to send to the server

the requests of the user. Also the client can manage the local sources, like the keyboard, monitor, CPU and hardwares. In the end, the client is that part of the application where the user sees and interacts with it.

4.2 Operation

A client-server system works according to the following scheme:

- The client isuess a request to the server with his IP address and the receiving gate number (host) can determinate on which local application can be assigned the message, that defines a particular service of the server.
- The server receive and answers through the address of the client terminal and his gate.

4.2.1 Concurrent and iteratives servers

- Concurrent: it's used to satisfy more requests of service aside of more clients through typical ways of multithreading and manage hardware sources and machine software.
- Iterative: it's used to satisfy only a request of service one a time aside of a client with a typical way that is to waiting the management process.

4.2.2 Port number

- The gates are numbers of bit used for indentify a particular connection of transport between those active moments on a calculator.
- The gates are notes like gates TCP and UDP in a range of 0-1023. The numbers of the gates signed up are in the range 1024-49151. The numbers of the gates in the range 49151-65535 are private gates or dynamics one and are not used in a particular application.
- More process can use the transport level TCP or UDP.
- When a client wants to comunicate with a server for identify a single server.
- Are necessary two levels of addressing:
  - Every machine on the network needs to have an address that checks uniquely.
  - Every applications on every machine(multitasking) needs to have an address that checks uniquely.

4.2.3 Identify a connection

- A gate assigns a service, but in case of multitasking could be that more server process are actives for the same service and needs to be a correct server process, this happens when the information of the server or the client for addressing packets is used.
- TCP and UDP are using 4 informations for identify a connection:
  - IP address of the server.
  - Number of the gate of the side service server.
  - IP address of the client.
  - Number of the gate of the side service client.

4.2.4 Endpoints e concurrent Server

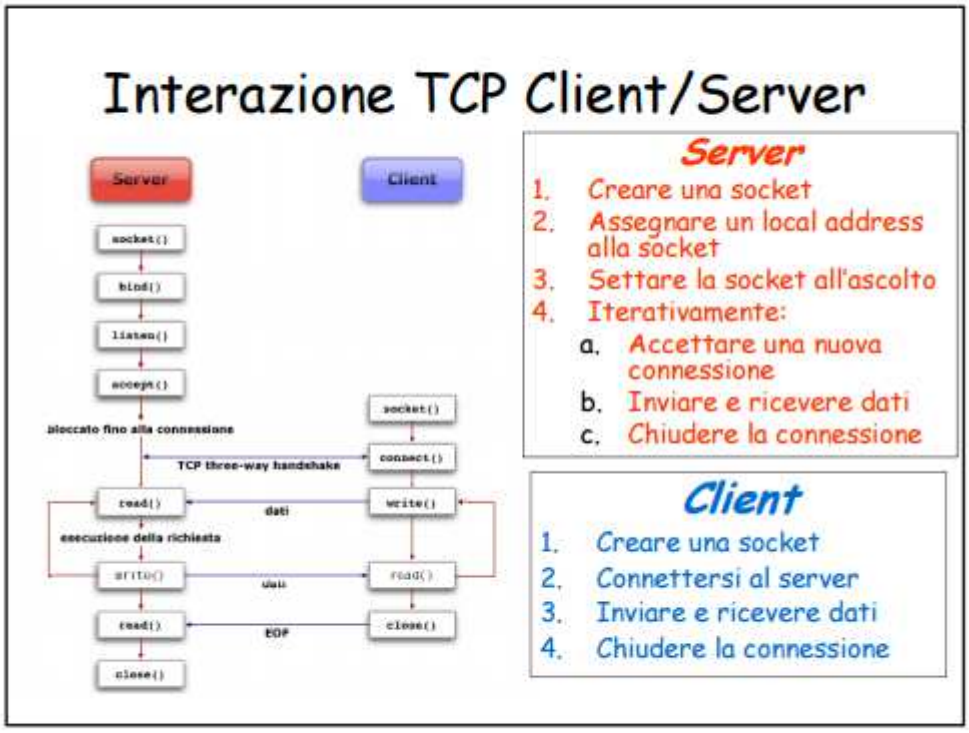
- The endpoint is the pair of IP address and gate and is in the IP protocol.
- Is considered a concurrent server with a server that generates a new children process to every client request.
- A new client makes a request to the same server.

4.2.5 Iterative application-transport level

- To be able to comunicate, 2 applications must interact with the respective operating systems by asking to transmit data through the network (the packet has a header that contains flow control, control, congestion, packet order in the message, handshaking function).
- The software of the application is outside the operating system, meanwhile that one that manage the transport protocol is inside.
- TCP and UDP are transport protocol of the applications client-server.

5 Use client-server with Socket

[5]



6 Makefile

6.1 Makefile birth

[6] As all programmers know, creating an executable program from more source files consists in compiling separately the source files one by one and then unite them in the final executable file. When there are many source files, the compilation could be boring and enervating. For this we will simply create:

- a file who contains the compilator to use.
- a file that explains the way to use it.
- a file who says which files to compile.

- the executable file (program or library) and other usefull things(as the directory where can be found the necessary libraries and the included files). All of this is contained in the Makefile.

```
PROGS=error my_io my_signal str_srv_sum str_cli_sum somma_client somma_server
CC = gcc
CCN = gcc -std=c99

all:$(PROGS)

error:  error.c
	${CC} -c error.c

my_io : my_io.c
	${CC} -c my_io.c

my_signal: my_signal.c
	${CC} -c my_signal.c

str_srv_sum: str_srv_sum.c
	${CC} -c str_srv_sum.c

str_cli_sum: str_cli_sum.c
	${CCN} -c str_cli_sum.c

somma_client:somma_client.o str_cli_sum.o my_io.o error.o
	${CC} -o $@ $^

somma_server:somma_server.o str_srv_sum.o my_signal.o my_io.o error.o
	${CC} -o $@ $^

clean:
	rm -f *.o
	rm -f *.*~
	rm -f *~
shish:
	rm -f *.o
	rm -f *.*~
	rm -f *~
	rm somma_client
	rm somma_server
```

6.2 The make command

The make command allows to develop programs of big sizes keeping traces of which portions of the whole program have been modified. But only those portions will be compilated. Then is proposed the example of the make command used in the exercise for the compilation of various parts of the server.

6.3 Other commands

These commands got created to delete all useless files. clean: In the following case, the "make shish" command, has been created to delete binary files.

```
shish:
    rm -f *.o
    rm -f *.*~
    rm -f *~
    rm somma_client
    rm somma_server
```

With this command, we delete directly the written files ("rm somma_client" and "rm somma_server") after the "make clean" command, without the obligation of selecting them one by one.

```
clean:
    rm -f *.o
    rm -f *.*~
    rm -f *~
```

7 Signals

The server creates a child process to manage the connection. When the child process ends the connection with the client sends a signal to father process.

The child process trasforms itself into a ZOMBIE process and the father process will end it by using the function wait() or waitpid(), otherwise the ZOMBIE process will end when also the father process will be ended(or killed).

References

[1] Wikipidia.

[2] it.ccm.net/contents/42-tcp-ip .

[3] Wikipedia e www.dacrema.com/Informatica/Socket.html.

[4] Wikipedia,Appunti di Informatica: Architettura Client/Server e Pear to Pear- Indirizzi IP – Indirizzamento statico e dinamico ,“Dipartimento di Ingegneria dell’Informazione ” - Università degli Studi di Siena (Gianluca Daino),www.di.uniba.it.

[5] Materiale fornito dal Prof. Iaccarino.

[6] Materiale didattico fornito dal Prof Iaccarino.