Reti di Calcolatori I

Prof. Roberto Canonico
Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione

Corso di Laurea in Ingegneria Informatica

A.A. 2018-2019

Esempi di programmi client/server in Python

**I lucidi presentati al corso sono uno strumento didattico
che NON sostituisce i testi indicati nel programma del corso**

Università degli Studi di Napoli
Federico II

Nota di copyright per le slide COMICS

Nota di Copyright

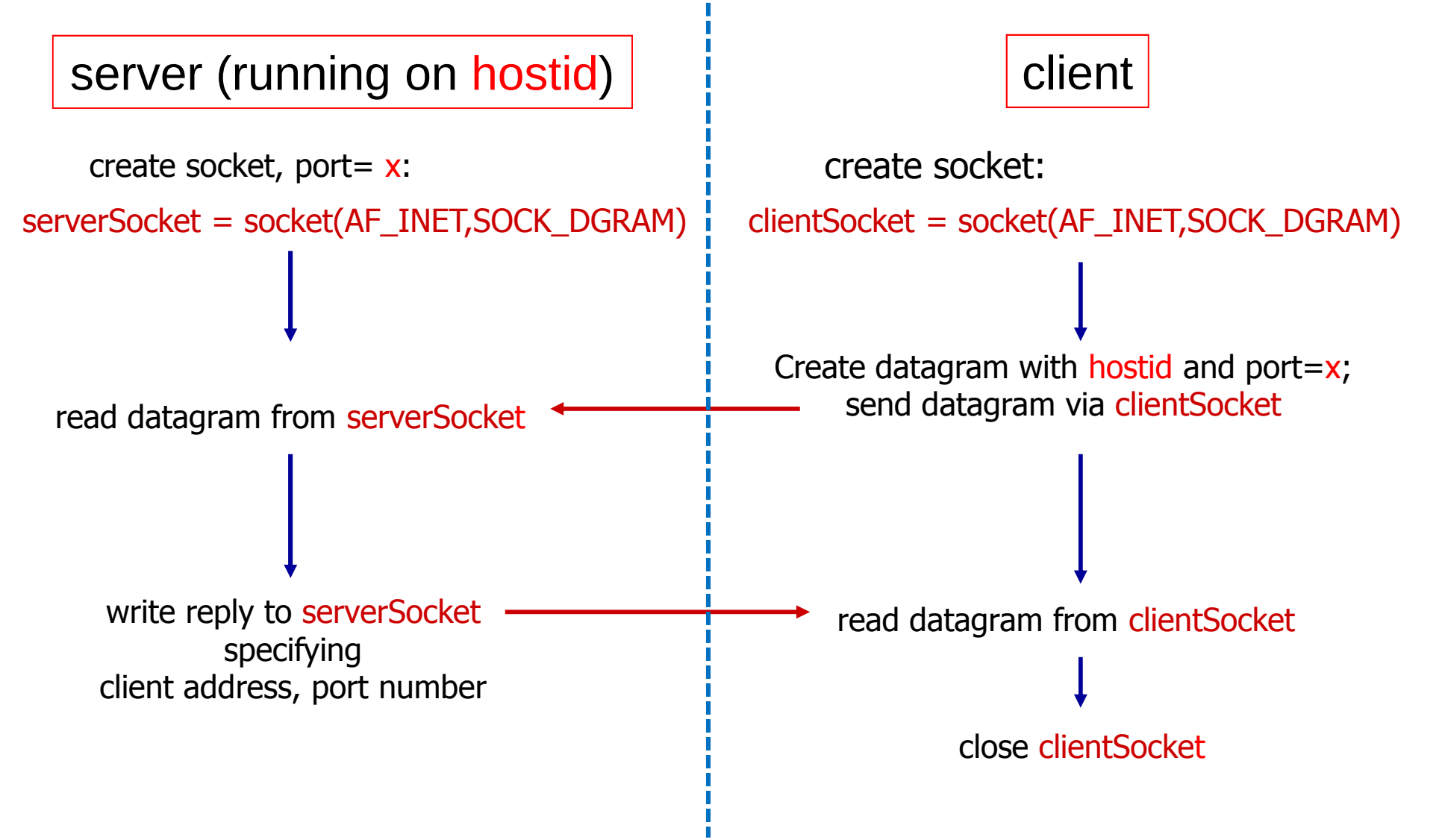
Autori:
Simon Pietro Romano, Antonio Pescapè, Stefano Avallone,
Marcello Esposito, Roberto Canonico, Giorgio Ventre

Università degli Studi di Napoli
Federico II

Disclaimer

- Gli esempi sono presi dalla sesta edizione del testo: Kurose-Ross - **Reti di calcolatori e internet**
- Non si intende presentare qui il linguaggio Python, ma evidenziare analogie e differenze tra la programmazione delle applicazioni client/server in Python rispetto a C

Client/server socket interaction: UDP



UDP client in Python

```
# -*- coding: utf-8 -*-
# include Python's socket library
from socket import *
serverName = 'localhost'
serverPort = 12000

# create UDP socket
clientSocket = socket(AF_INET, SOCK_DGRAM)

# get user keyboard input
message = raw_input('Input lowercase sentence:')

# Attach server name, port to message; send into socket
clientSocket.sendto(message, (serverName, serverPort))

# read reply message from socket into modifiedMessage string
modifiedMessage, serverAddress = clientSocket.recvfrom(2048)

# Print out received modifiedMessage string
print modifiedMessage
# Close socket
clientSocket.close()
```

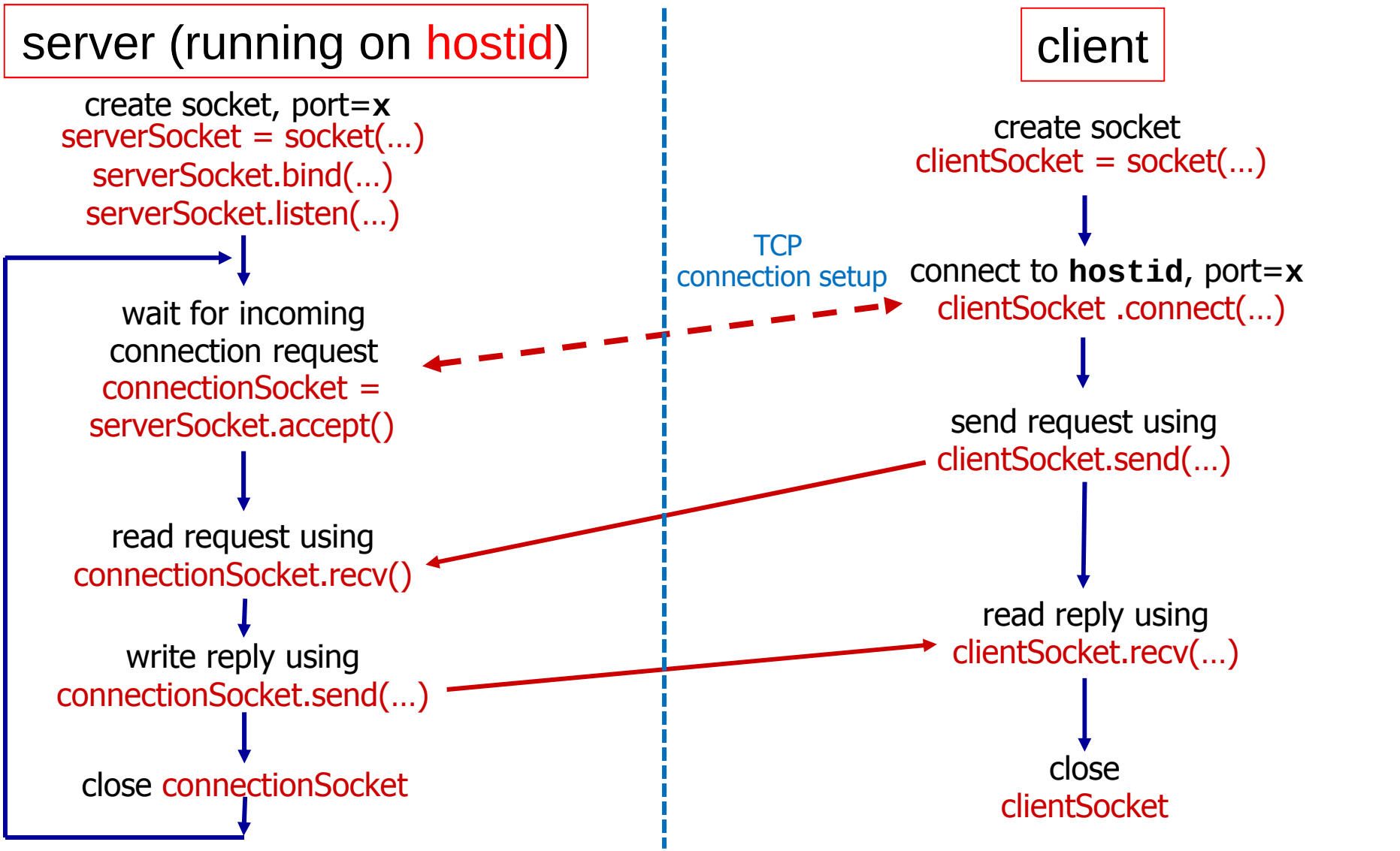
UDP server in Python

```
# -*- coding: utf-8 -*-
# include Python's socket library
from socket import *
serverPort = 12000
# create UDP socket
serverSocket = socket(AF_INET, SOCK_DGRAM)

# bind socket to local port number 12000
serverSocket.bind(('', serverPort))
print "The server is ready to receive"

# Loop forever
while 1:
    # Read from UDP socket into message
    # getting client IP and port
    message, clientAddress = serverSocket.recvfrom(2048)
    # Convert message to upper case
    modifiedMessage = message.upper()
    # Send back modified string to client
    serverSocket.sendto(modifiedMessage, clientAddress)
```

Client/server socket interaction: TCP



TCP client in Python

```
# -*- coding: utf-8 -*-
# include Python's socket library
from socket import *
serverName = 'localhost'
serverPort = 12000

# create TCP socket
clientSocket = socket(AF_INET, SOCK_STREAM)
# connect socket to remote server at (serverName, serverPort)
clientSocket.connect((serverName, serverPort))
# get user keyboard input
sentence = raw_input('Input lowercase sentence:')
# Send sentence into socket, no need to specify server IP and port
clientSocket.send(sentence)

# read reply message from socket into modifiedMessage string
modifiedSentence = clientSocket.recv(1024)

# Print out received modifiedMessage string
print modifiedSentence
# Close socket
clientSocket.close()
```

TCP server in Python

```
# -*- coding: utf-8 -*-
# include Python's socket library
from socket import *
serverPort = 12000
# create TCP socket
serverSocket = socket(AF_INET, SOCK_STREAM)
# bind socket to local port number 12000
serverSocket.bind(('', serverPort))
# put socket in passive mode
serverSocket.listen(1)
print "The server is ready to receive"
# Loop forever
while 1:
    # server waits for incoming connections on accept()
    # for incoming requests, new socket created on return
    connectionSocket, addr = serverSocket.accept()
    # receive sentence on newly established connectionSocket
    sentence = connectionSocket.recv(1024)
    # convert message to upper case
    modifiedSentence = sentence.upper()
    # send back modified string to client
    connectionSocket.send(modifiedSentence)
```