

# Uvod u mrežno programiranje

Slavica Tomović (slavicat@ucg.ac.me)

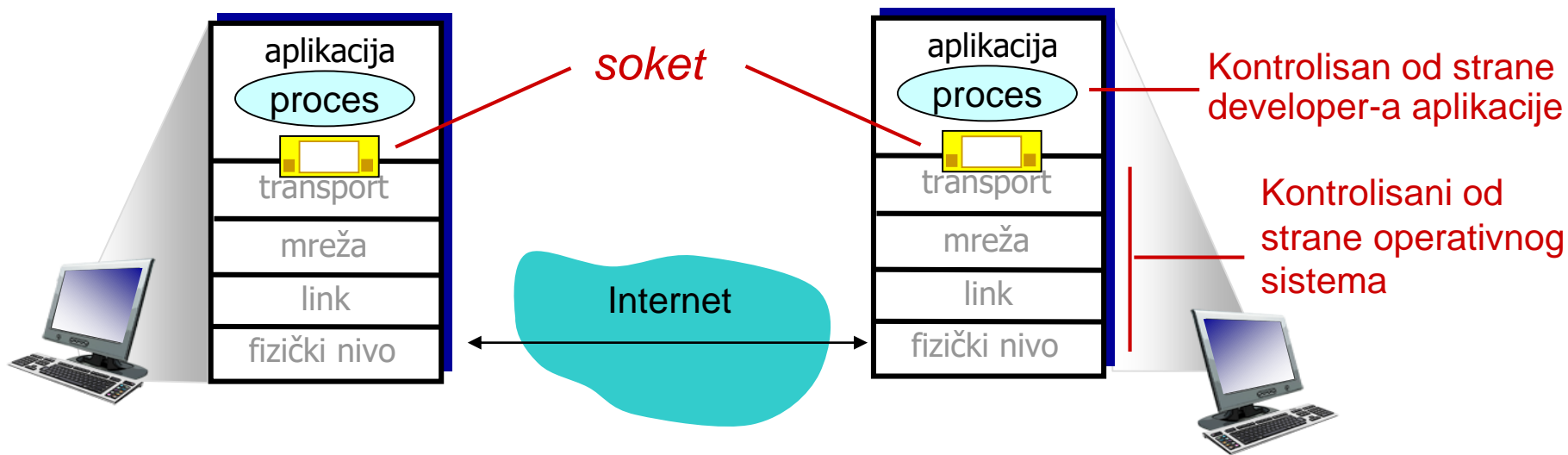
Elektrotehnički fakultet, Podgorica

Univerzitet Crne Gore

# Programiranje soketa

**cilj:** naučiti kako se razvijaju klijent/server aplikacije koje komuniciraju preko soketa

**soket:** krajnje tačke end-to-end transportne konekcije između dva aplikacijska procesa



# Programiranje soketa

*Dva tipa soketa za dva tipa transportnih servisa:*

**UDP:** nepouzdan prenos

**TCP:** pouzdan prenos, *stream* bajtova

*Primjer Aplikacije:*

1. Klijent čita liniju podataka unesenih putem tastature i šalje je serveru.
2. Server prihvata podatke i kovertuje karaktere u velika slova (*uppercase*).
3. Server šalje modifikovane podatke klijentu.
4. Klijent prihvata modifikovane podatke i štampa ih na ekranu.

# Programiranje UDP soketa

UDP: nema “konekcije” između klijenta i servera

- ❖ nema kontrolne komunikacije prije slanja podataka
- ❖ pošiljalac eksplicitno dodaje destinacionu IP adresu i broj porta svakom paketu
- ❖ primalac ekstrahuje IP adresu i broj porta pošiljaoca iz primljenog paketa

UDP: poslati podaci se mogu izgubiti ili primiti neredosledno

Sa aspekta aplikacije:

- ❖ UDP pruža *nepouzdan* prenos grupe datagrama između klijenta i servera

# Klijent-server interakcija: UDP

## server (koristi serverIP)

kreiraj soket, port= x:  
`serverSocket =  
DatagramSocket(x)`

↓  
učitaj datagram iz  
`serverSocket`

↓  
napiši odgovor za  
`serverSocket`  
navodeći IP adresu  
i broj porta klijenta

## klijent

kreiraj soket:

`clientSocket =  
DatagramSocket()`

↓  
kreiraj datagram sa IP adresom  
i brojem porta servera (port=x);  
pošalji datagram sa `clientSocket`

↓  
učitaj datagram iz `clientSocket`

↓  
zatvori  
`clientSocket`



# Primjer aplikacije: UDP klijent

## *Python UDPClient*

Python biblioteka  
za sokete

→ `import socket`

`serverName = 'alias ili IP adresa servera'`

`serverPort = 12000`

kreira UDP soket za  
server

→ `clientSocket = socket.socket(socket.AF_INET,  
socket.SOCK_DGRAM)`

učitavanje unosa sa  
tastature

→ `message = raw_input('Unesite rečenicu malim slovima:')`

dodavanje imena server i  
destinacionog porta u poruku  
koja se šalje soketom

→ `clientSocket.sendto(message,(serverName, serverPort))`  
`modifiedMessage, serverAddress =`

učitavanje modifikovanih  
karaktera u string

`clientSocket.recvfrom(2048)`

štampa primljeni string i  
zatvara soket

`print modifiedMessage`  
`clientSocket.close()`

# Primjer aplikacije: UDP server

## *Python UDPServer*

```
from socket import *
```

```
serverPort = 12000
```

kreira UDP soket

pridružuje soket lokalnom  
broju porta 12000

```
serverSocket = socket(AF_INET, SOCK_DGRAM)
```

```
serverSocket.bind(("", serverPort))
```

```
print "Server je spreman za prijem podataka"
```

beskonačna  
petlja

učitava podatke iz UDP  
soketa u varijablu *message*  
i ekstrahuje klijentovu IP  
adresu i broj porta

```
while 1:
```

```
    message, clientAddress = serverSocket.recvfrom(2048)
```

```
    modifiedMessage = message.upper()
```

```
    serverSocket.sendto(modifiedMessage, clientAddress)
```

Šalje string sa velikim  
slovima klijentu



# Programiranje TCP soketa

klijent mora kontaktirati  
server prije slanja  
podataka

- server mora biti već pokrenut
- server mora imati soket za prihvatanje klijentskih konekcija

klijent se povezuje sa  
serverom:

kreiranjem TCP soketa,  
navodeći IP adresu i broj  
porta serverskog procesa

server prihvata konekciju:  
*kreiranjem novog za  
konekciju rezervisanog  
soketa*

server može simultatno  
komunicirati sa više klijenata

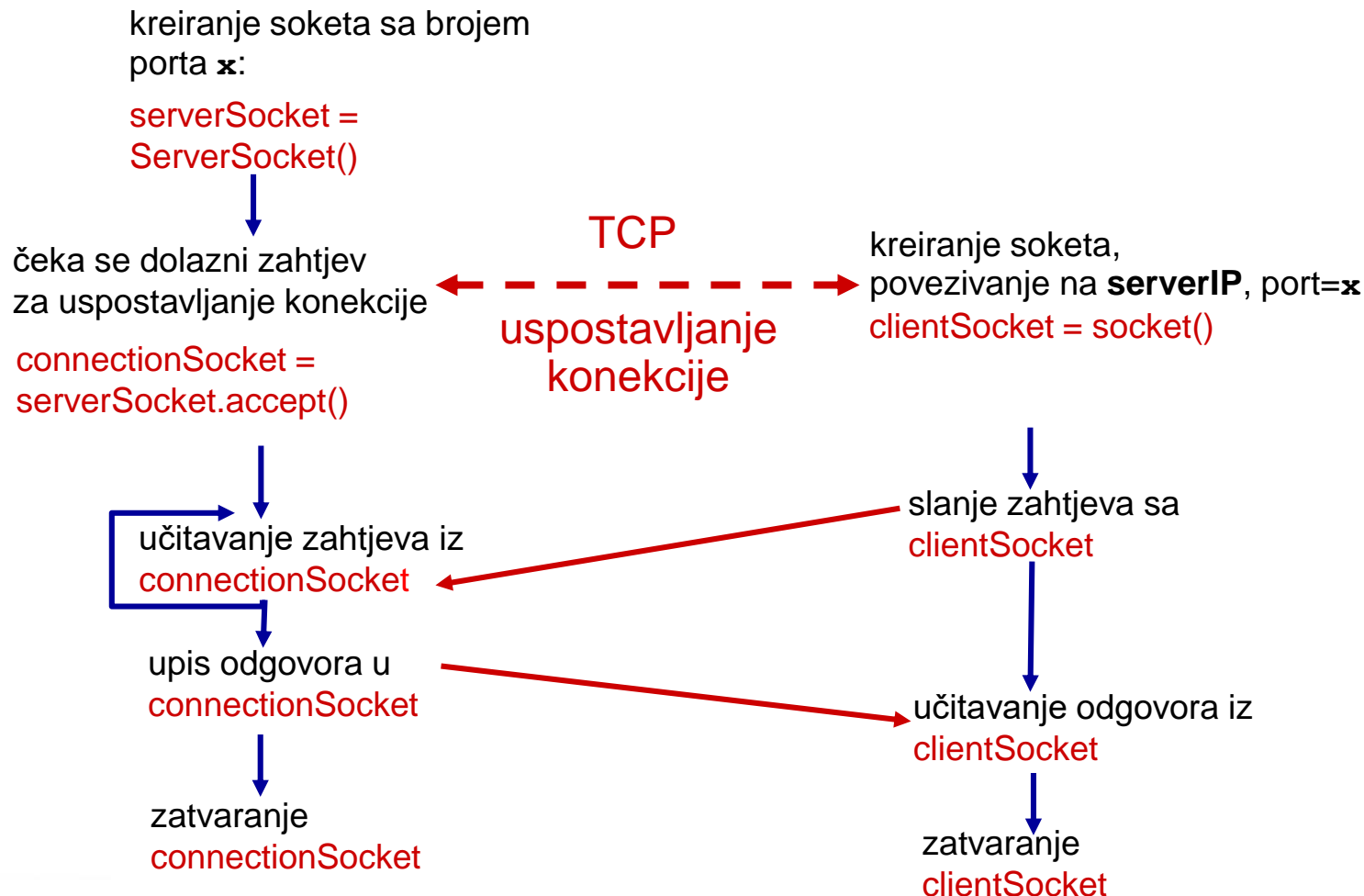
Sa aspekta aplikacije:  
TCP pruža pouzdan, redosledan  
prenos toka bajtova (“pipe”)  
između klijenta i servera



# Klijent-server interakcija: TCP

server (koristi serverIP)

klijent



# Primjer aplikacije: TCP klijent

## *Python TCPClient*

```
import socket
serverName = 'alias ili IP adresa servera'
serverPort = 12000
clientSocket = socket.socket(socket.AF_INET,
                              socket.SOCK_STREAM)
clientSocket.connect((serverName, serverPort))
message = raw_input('Unesite rečenicu malim slovima:')
clientSocket.send(message)
modifiedMessage = clientSocket.recv(1024)
print 'Od servera:', modifiedMessage
clientSocket.close()
```

kreirati TCP socket za  
serverIP, udaljeni port  
12000



nema potrebe za  
navođenjem IP adrese  
servera i destinacionog  
broja porta



# Primjer aplikacije: TCP server

## *Python TCPServer*

kreiranje TCP soketa za prijem zahtjeva	→	from socket import *
		serverPort = 12000
server osluškuje dolazne TCP zahtjeve	→	serverSocket = socket(AF_INET, SOCK_STREAM)
		serverSocket.bind(('', serverPort))
		serverSocket.listen(1)
beskonačna petlja	→	print 'Server je spreman za prijem podataka'
		while 1:
server čeka na accept(), kreira se novi soket preko kojeg prihvata poruku	→	connectionSocket, addr = serverSocket.accept()
		sentence = connectionSocket.recv(1024)
učitavanje bajtova iz soketa (ali ne i adresa kao kod UDP-a)	→	capitalizedSentence = sentence.upper()
		connectionSocket.send(capitalizedSentence)
zatvaranje konekcije ali ne i soketa za prijem TCP zahtjeva	→	connectionSocket.close()

# Dodatna dokumentacija

- <http://python.org/>
  - dokumentacija, tutorijali ...
- Knjige:
  - *Learning Python*, Mark Lutz
  - *Python Essential Reference*, David Beazley
  - *Python Cookbook*, Martelli, Ravenscroft and Ascher
  - (online <http://code.activestate.com/recipes/langs/python/>)
  - <http://wiki.python.org/moin/PythonBooks>

