Programação com Sockets

Ricardo Couto Antunes da Rocha
rcarocha@inf.ufg.br

Parâmetros de protocolos

- Objetivo do protocolo
- Protocolo de transporte: TCP ou UDP
- Porta (servidora!): padronizada
- Características gerais: arquitetura P2P/CS, com estado/sem estado (sessão), textual, proprietário/aberto, escopo de uso, etc.

Protocolo da Camada de Aplicação

- Define
  - Tipos de mensagens trocadas, como de requisição e resposta
  - Sintaxe dos vários tipos de mensagens, tais como campos da mensagem e como os campos são delineados
  - Semântica dos campos
  - Regras para identificar como e quando um processo envia mensagens e responde a mensagens

Modelos de Protocolo de Aplicação

- Request-Reply
  - Mais simples modelo de protocolo
  - Conversa é composta de uma requisição (**request**), enviada em uma única mensagem, que é respondida (**reply**) em uma única mensagem.
  - Após uma resposta, o protocolo volta ao estágio inicial, esperando uma nova requisição.
  - *Tipicamente*, o protocolo não mantém estado.
  - Exemplo: HTTP, RPC, RMI

Modelos de Protocolos de Aplicação

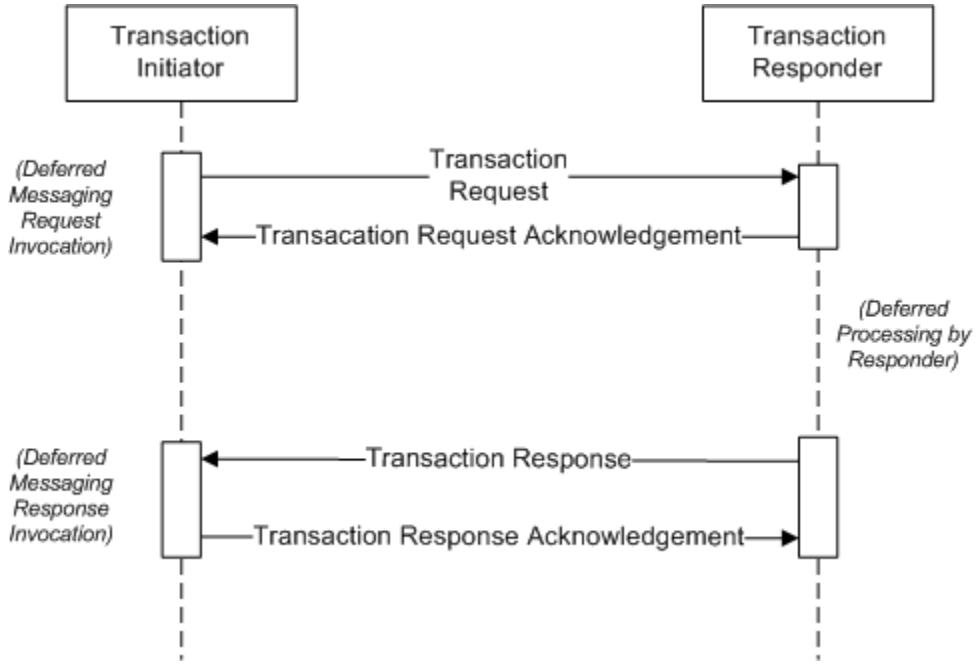
- Protocolo Interativo
  - Protocolo com estado, cujo funcionamento é baseado em sucessivas mensagens Request-Reply.

Modelos de Protocolos de Aplicação

- Fluxo/Streams
  - Requer o envio de um fluxo contínuo (streaming) de mensagens entre os processos
  - Protocolo com estado.
  - Exemplo típico: Protocolos para envio de dados multimídia (video, som), como RTP
  - *Tipicamente*, exige um canal de comunicação em paralelo, por onde são enviadas mensagens de controle sobre o fluxo (parar, reiniciar, mudar estado conexão)

Modelos de Protocolos de Aplicação

- Mensagens Assíncronas
  - Utilizado quando tempo de processamento da mensagem leverá um tempo indeterminado.
  - Após o envio da mensagem, o cliente/requisitor fica passivo.



Exemplo

An Overview of the File Transfer Protocol. Mike Gleason, NcFTP Software. http://www.ncftp.com/libncftp/doc/ftp_overview.html

Clinton: (Dials the phone number for the mail service)

Service: "Hello, this is the Acme Mail Service. How may I help you today?"

Clinton: "Hello, this is Carl Clinton. I would like to access mailbox number MB1234."

Service: "OK, Mr. Clinton, I need to verify that you may access mailbox MB1234. What is your password?"

Clinton: "My password is QXJ4Z2AF."

Service: "Thank you Mr. Clinton, you may proceed."

Clinton: "For now, I'm only interested in looking at the bills and invoices, so look at the folder marked "bills" in my mailbox."

Service: "OK."

Clinton: "Please prepare to have your assistant call my secretary at +1 402 555 1234."

Service: "OK."

Clinton: "Now call my secretary and tell him the names of all the items in the bills folder of my mailbox. Tell me when you have finished."

Server: "My assistant is calling your secretary now."

Server: "My assistant has sent the names of the items."

Clinton: (Receives the list from his secretary and notices a bill from Yoyodyne Systems.)

"Please prepare to have your assistant send to my fax machine +1 402 555 7777."

Service: "OK."

Clinton: "Now fax a copy of the bill from Yoyodyne Systems."

Server: "My assistant is calling your fax machine now."

Server: "My assistant has finished faxing the item."

Clinton: "Thank you, that is all. Good bye."

Server: "Goodbye."

Protocolo (1/2)

Client:		Connects to the FTP service at port 21 on the IP address 172.16.62.36.
Server:	`220 Hello, this is the Acme Mail Service.`	
Client:	`USER MB1234`	
Server:	`331 Password required to access user account MB1234.`	
Client:	`PASS QXJ4Z2AF`	Note that this password is not encrypted. The FTP is susceptible to eavesdropping!
Server:	`230 Logged in.`	
Client:	`CWD Bills`	Change directory to "`Bills.`"
Server:	`250 "/home/MB1234/Bills" is new working directory.`	
Client:	`PORT 192,168,1,2,7,138`	The client wants the server to send to port number 1930 on IP address 192.168.1.2. In this case, 192.168.1.2 is the IP address of the client machine.
Server:	`200 PORT command successful.`	
Client:	`LIST`	Send the list of files in "`Bills.`"
Server:	`150 Opening ASCII mode data connection for /bin/ls.`	The server now connects out from its port 20 on 172.16.62.36 to port 1930 on 192.168.1.2.

An Overview of the File Transfer Protocol. Mike Gleason, NcFTP Software. `http://www.ncftp.com/libncftp/doc/ftp_overview.html`

Protocolo (2/2)

Server:	`200 PORT command successful.`	
Client:	`LIST`	Send the list of files in `"Bills."`
Server:	`150 Opening ASCII mode data connection for /bin/ls.`	The server now connects out from its port 20 on 172.16.62.36 to port 1930 on 192.168.1.2.
Server:	`226 Listing completed.`	That succeeded, so the data is now sent over the established data connection.
Client:	`PORT 192,168,1,2,7,139`	The client wants the server to send to port number 1931 on the client machine.
Server:	`200 PORT command successful.`	
Client:	`RETR Yoyodyne.TXT`	Download `"Yoyodyne.TXT."`
Server:	`150 Opening ASCII mode data connection for Yoyodyne.TXT.`	The server now connects out from its port 20 on 172.16.62.36 to port 1931 on 192.168.1.2.
Server:	`226 Transfer completed.`	That succeeded, so the data is now sent over the established data connection.
Client:	`QUIT`	
Server:	`221 Goodbye.`	

An Overview of the File Transfer Protocol. Mike Gleason, NcFTP Software. `http://www.ncftp.com/libncftp/doc/ftp_overview.html`

Exemplo: Programa em Execução

```
ksh$ /usr/bin/ftp
ftp> open ftp.acmemail.example.com
Connected to ftp.acmemail.example.com (172.16.62.36).
220 Hello, this is the Acme Mail Service.
Name (ftp.acmemail.example.com:root): MB1234
331 Password required to access user account MB1234.
Password: QXJ4Z2AF
230 Logged in.
ftp> cd Bills
250 "/home/MB1234/Bills" is new working directory.
ftp> ls
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
-rw-r--r--   1 ftpuser  ftpusers     14886 Dec  3 15:22 Acmemail.TXT
-rw-r--r--   1 ftpuser  ftpusers    317000 Dec  4 17:40 Yoyodyne.TXT
226 Listing completed.
ftp> get Yoyodyne.TXT
local: Yoyodyne.TXT remote: Yoyodyne.TXT
200 PORT command successful.
150 Opening ASCII mode data connection for Yoyodyne.TXT.
226 Transfer completed.
317000 bytes received in 0.0262 secs (1.2e+04 Kbytes/sec)
ftp> quit
221 Goodbye.
```

An Overview of the File Transfer Protocol. Mike Gleason, NcFTP Software.
http://www.ncftp.com/libncftp/doc/ftp_overview.html

Representação externa de dados

- Formato independente de linguagem, SO etc
- Utilizada para comunicação dos dados de requisições e respostas entre clientes e servidores
- Formato serializado
- *Marshalling*: conversão entre a representação interna e externa

Representação externa de dados

- CORBA Common Data Representation
- Serialização de objetos em Python/Java
- XML (Extensible Markup Language)
- JSON
- *protocol buffers*

Serialização de Objetos

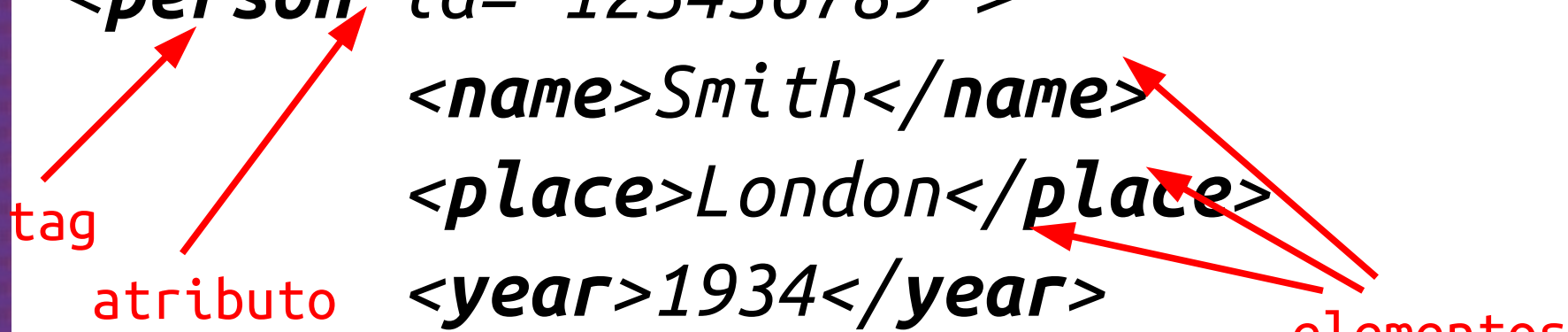
```
import pickle

data1 = {'a': [1, 2.0, 3, 4+6j],
         'b': ('string', u'Unicode string'),
         'c': None}

selfref_list = [1, 2, 3]
selfref_list.append(selfref_list)

output = open('data.pkl', 'wb')

# Pickle dictionary using protocol 0.
pickle.dump(data1, output)

# Pickle the list using the highest protocol
available.
pickle.dump(selfref_list, output, -1)

output.close()
```

XML (Extensible Markup Language)

- Define a estrutura lógica de documentos
- Usos de interesse aqui:
  - definir a interface de serviços Web
  - prover a representação externa de dados na comunicação entre clientes e serviços
- Representação textual: independente de plataforma
- Representação hierárquica
- Extensível: novos tags podem ser definidos
- Auto-descritiva: tags, esquemas e namespaces
- Representação textual ao invés de binário aumenta significativamente o tamanho da mensagem

Definição da estrutura *Pessoa* em XML

```
<person id="123456789">
    <name>Smith</name>
    <place>London</place>
    <year>1934</year>
    <!-- a comment -->
</person >
```



Namespaces

- Permitem definir contextos para os marcadores (tags) utilizados em um documento
- Referenciados através de URLs
- Evitam choques de nomes de tags em contextos diferentes

```
<person pers:id="123456789"
          xmlns:pers="http://www.cdk4.net/person">
   <pers:name> Smith </pers:name>
   <pers:place> London </pers:place >
   <pers:year> 1934 </pers:year>
</person>
```

Esquemas XML

- Definem:
  - os elementos e atributos que podem aparecer em documentos XML (i.e., um vocabulário)
  - como os elementos são aninhados
  - a ordem, o número e o tipo dos elementos
- Usados para validar documentos XML
- Um mesmo esquema pode ser compartilhado por vários documentos

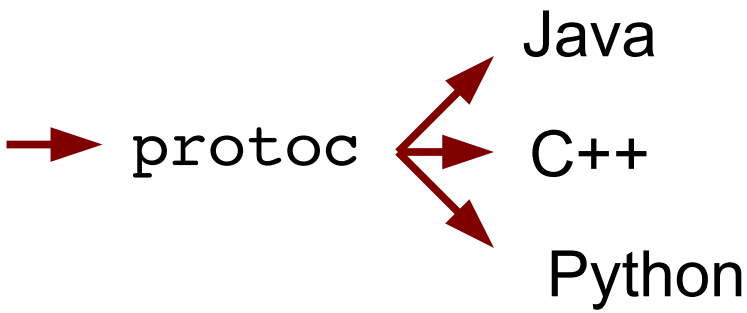
Um esquema XML para a estrutura *Pessoa*

```
<xsd:schema  xmlns:xsd =URL of XML schema definitions >
 <xsd:element name="person" type="personType" />
  <xsd:complexType name="personType">
    <xsd:sequence>
     <xsd:element name="name" type="xs:string"/>
      <xsd:element name="place" type="xs:string"/>
       <xsd:element name="year"
                          type="xs:positiveInteger"/>
    </xsd:sequence>
   <xsd:attribute name="id" type="xs:positiveInteger"/>
  </xsd:complexType>
</xsd:schema>
```

Protocol Buffers

- Projeto gestado no Google para implementação de protocolos de comunicação
  - Representação independente de dados
- Representação arquivo `.proto`

```
message Person {
  required int32 id = 1;
  required string name = 2;
  optional string email = 3;
}
```



JSON

- Notação também baseada em ASCII e mais enxuta e legível que XML
- Independente de linguagem
- Exemplos e comparação com XML
  - http://json.org/example.html