Exa844 – Programação para Redes
Professor: João B. Rocha-Junior

Atividade 2: *P2P File Sharing*

Descrição:
Existem diversos aplicações para baixar arquivos diretamente de outros computadores sem passar por um servidor central (ex: kazaa, bittorrent, gnutella, napster). Estas aplicações se baseiam na comunicação direta entre computadores chamados de *peers*. Para se conectar a rede é necessário saber o endereço (ip:port) de ao menos um *peer* ativo. Uma das principais funcionalidade dessa aplicação é localizar arquivos de diversos formatos, analisando o seu conteúdo ou apenas o título. Nesta atividade você vai desenvolver um programa capaz de se comunicar com outros programas e localizar arquivos armazenados localmente nestas máquinas. Segue as principais *user stories* da sua aplicação:

User stories

User Story	Título	Breve Descrição
1	Listar hosts local	Os hosts armazenados localmente são listados na tela
2	Listar hosts remotos	Os hosts armazenados em um (ip:port) são listados na tela
3	Atualizar hosts locais	Atualiza a lista de hosts armazenada localmente. A atualização é a união entre os hosts locais e os hosts obtidos de um determinado peer (ip:port)
4	Buscar arquivos localmente	Entra com um conjunto de palavras-chave e retorna os arquivos armazenados em um determinado diretório que atendem ao critério de busca
5	Buscar arquivos remoto	Entra com um conjunto de palavras-chave e retorna os arquivos armazenados localmente que atendem ao critério de busca
6	Atualizar arquivos locais	Atualiza os arquivos locais (diretório) a partir de um conjunto de arquivos "baixados" de um host remoto

O seu programa deve manter uma lista de hosts (ip:porta) conhecidos. Esta lista deve ser armazenada em um arquivo xml que segue o formato especificado em *Hosts.dtd.*

Hosts.dtd
```
<!DOCTYPE hosts [
<!ELEMENT hosts (host*)>
     <!ELEMENT host (ip?,port?)>
        <!ELEMENT ip (#PCDATA)>
        <!ELEMENT port (#PCDATA)>
]>
```

A comunicação entre os programas segue o protocolo especificado em *p2ps2.dtd* e consiste em enviar uma mensagem de solicitação (*request*) e processar a resposta (*response*). Por exemplo, para solicitar a lista de *hosts* conhecidos de um determinado *peer*, o seu programa deve enviar para o *peer* a seguinte xml:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE p2pse SYSTEM "p2pse.dtd">
<p2pse>
  <getHosts/>
</p2pse>
```

Ao receber uma mensagem solicitando a lista de hosts, uma aplicação deve retornar um arquivo xml com a lista de hosts conhecidos. Por exemplo:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE p2pse SYSTEM "p2pse.dtd">
<p2pse>
  <getHostsResponse>
     <host>
       <ip>10.67.5.1</ip>
       <port>3121</port>
     </host>
     <host>
       <ip>10.67.5.7</ip>
       <port>1024</port>
     </host>
  </getHostsResponse>
</p2pse>
```

p2pse.dtd

```
<!DOCTYPE p2pse [
  <!ELEMENT p2pse (getHosts?, getHostsResponse?, searchFile?,
             searchFileResponse?, getFiles?, getFilesResponse?)>
  <!ELEMENT getHosts EMPTY>
  <!ELEMENT getHostsResponse (host*)>
  <!ELEMENT host (ip?,port?)>
  <!ELEMENT ip (#PCDATA)>
  <!ELEMENT port (#PCDATA)>
  <!ELEMENT searchFile (keywords?)>
  <!ELEMENT keywords (#PCDATA)>
  <!ELEMENT searchFileResponse (file*)>
  <!ELEMENT file (fileName?, fileSize?)>
  <!ELEMENT fileName (#PCDATA)>
  <!ELEMENT fileSize (#PCDATA)>
  <!ELEMENT getFiles (fileName*)>
  <!ELEMENT getFilesResponse (fileData*)>
  <!ELEMENT fileData (fileName?, data?)>
  <!ELEMENT data (#PCDATA)>
  <!ATTLIST data encode CDATA #REQUIRED>
]>
```

A sua aplicação deve ser capaz de solicitar mensagem e tratar mensagens solicitadas por outros programas.

Produto:

Aplicação deve atender a todas as *user stories* listadas, seguindo o protocolo xml apresentado. A aplicação também deve ter uma interface de usuário bem simples (ex: linha de comando), permitindo executar as *user stories*. O programa deve ser enviado por email, até a data especificada no cronograma do curso. No email, você deve indicar os autores e alguma explicação sobre como executar a aplicação, caso necessário.