Definição do Trabalho da Disciplina

Este documento é muito importante: LEIAM ATÉ O FINAL!

O trabalho final da disciplina consiste na implementação de um mecanismo de processamento distribuído de tarefas sobre uma rede peer-to-peer.

Cada grupo deve implementar um peer que utiliza sockets e mensagens XML para comunicação. Existem duas funcionalidades principais a serem implementadas no trabalho.

1) Formação e manutenção da rede peer-to-peer
2) Processamento distribuído de tarefas

Formação e manutenção da rede peer-to-peer

Inicialmente o sistema deve permitir cadastrar um ou mais vizinhos (Nome do Peer, IP e Porta). O nome do Peer é sempre o nome do grupo que aquele peer representa (G1, G2, ...). Uma vez cadastrados os vizinhos, o sistema envia, para cada um dos vizinhos cadastrados, uma mensagem XML perguntando "quem vc conhece".

O peer, ao receber uma mensagem XML deste tipo, responde com os detalhes dos vizinhos que possui cadastrados.

Exemplo:
Peer G1 conhece Peer G2
Peer G2 conhece Peer G3 e G4

Peer G1 pergunta ao peer G2 quem ele conhece. Peer G2 adiciona G1 na sua lista de conhecidos (que chamaremos de lista online), e responde a G1 com os dados de G3 e G4 (ele não inclui G1 na resposta, afinal foi G1 quem mandou a pergunta). Peer G2 pergunta aos peers G3 e G4 quem eles conhecem (não pergunta ao G1 pois acabou de receber uma mensagem dele). G3 conhece apenas G2, que adicionou em seu cadastro no momento que recebeu a mensagem de G2. G4 idem. Ambos respondem com uma lista vazia.

Após 1 minuto, começa uma nova rodada de mensagens de "quem vc conhece". Nesta rodada todos perguntam novamente a todos os peers de sua lista "quem vc conhece". G3 e G4, em especial, agora conhecem G2, de quem receberam mensagem na rodada passada. Ao perguntar a G2, perguntam a lista com G1, G3 e G4.

Assim, a cada 1 minuto, novas rodadas acontecem. Para cada mensagem enviada, é necessário setar um timeout (vamos usar 30 segundos). Se não houver resposta, o peer que não respondeu passa a fazer parte de uma lista offline. Esse peer continuará a ser perguntado a cada 1 minuto. Quando responder, volta à lista online.

Cada peer deve mostrar, numa parte da tela de sua interface, os peers online e offline, e essa lista deve ser atualizada sempre que houver alguma modificação no status de algum peer conhecido, e sempre que um novo peer for descoberto.

Processamento Distribuído de Tarefas

O objetivo da rede peer to peer formada é compartilhar capacidade de processamento. As tarefas que processaremos neste trabalho são tarefas de ordenação de números. Um peer que precise processar uma tarefa de ordenação agirá sempre da seguinte forma:

1) Analisa se a sequencia de números tem menos de 50 números
2) Se sim, ordena a sequencia
3) Se não, divide a sequencia em duas (usando quick sort), separa o pivot e manda cada uma das metades para 2 peers de sua lista online
4) Ao receber as respostas, o peer junta as duas listas de resposta com o pivot para compor resultado desejado
5) Se um dos peers para o qual foi mandada uma tarefa ficar offline antes de mandar a resposta, é preciso que o peer que delegou a tarefa escolha outro peer e reenvie a tarefa

Notem que dependendo do tamanho da sequencia inicial, podem ocorrer várias divisões e várias delegações de tarefas. Essa estratégia de processamento é chamada de MapReduce (http://en.wikipedia.org/wiki/MapReduce).

A interface do sistema deve ter a opção de submeter uma tarefa, e de monitorar as tarefas em execução no momento (e as que estão aguardado resposta de outros peers). Note que para as tarefas que chegam solicitadas por outros peers, não será usada a interface de submissão de tarefas. O processamento deve ser feito automaticamente, e nesse caso a tarefa aparece apenas na interface de monitoramento. Na interface de monitoramento, é necessário que apareça o tempo gasto com cada tarefa, desde que ela foi submetida até que a resposta tenha sido recebida. É preciso também manter um histórico das tarefas já processadas, com os tempos que cada tarefa gastou.

Para as tarefas originadas no peer (submetidas via interface de submissão), o usuário deve poder escolher quantos números deseja que a tarefa tenha. Os números a serem ordenados devem ser gerados de forma randômica.

Uso de Padrões XML

A implementação do peer deve obrigatoriamente utilizar SAX ou DOM para ler os documentos XML. Além disso, deve usar um dos seguintes padrões: XPath ou XQuery ou XSLT. Estes podem ser usados para implementar alguma funcionalidade já exigida no trabalho, ou alguma funcionalidade extra.

Detalhes operacionais

Toda a comunicação das aplicações será feita através de arquivos XML que serão enviados pela rede via sockets (a professora fornecerá as classes Java que fazem a comunicação – vocês precisam se preocupar apenas com os arquivos XML (seu conteúdo e processamento)). Cada grupo

deve usar estas classes para fazer a comunicação. Se o grupo optar por utilizar outra linguagem de programação, fica automaticamente responsável pela implementação das classes de comunicação nessa linguagem.

Formatos das mensagens XML

1) Mensagem de Descoberta de Peer ("quem vc conhece? ")

<discover sender="G1" ip ="143.64.122.1" port="6754" destination="G2" message-id="01">
</discover>

sender = <id do grupo que está enviando a mensagem>
ip = <IP do grupo que está enviando a mensagem>
port = <porta do grupo que está enviando a mensagem>
destination = <id do grupo que vai receber a mensagem>
message-id = <id da mensagem>

O id da mensagem serve para controlar que resposta se refere a que mensagem. A cada nova mensagem, o id deve ser incrementado. Notem que só o id da mensagem não é um identificador único. É necessário também usar o identificador do grupo. Só assim teremos como identificar unicamente uma mensagem.

2) Mensagem de resposta de descoberta de peer

```
<response-discover sender="G2" destination="G1" message-id="02" response-to="01">
        <peer>
                <name>G3</name>
                <ip>10.20.155.21</ip>
                <port>1234</port>
        </peer>
        <peer>
                <name>G4</name>
                <ip>10.20.155.25</ip>
                <port>1254</port>
        </peer>
</response-discover>
```

sender = <id do grupo que está enviando a mensagem>
destination = <id do grupo que vai receber a mensagem>
message-id = <id da mensagem>
response-to = <id da mensagem que originou esta resposta>

3) Mensagem de solicitação de tarefa

```
<task sender="G1" destination="G2" message-id="05">
        <numbers count="6">
                <n>234</n>
                <n>125</n>
```

```
            <n>867</n>
            <n>12</n>
            <n>1</n>
            <n>50</n>
        </numbers>
</task>
```

sender = <id do grupo que está enviando a mensagem>
destination = <id do grupo que vai receber a mensagem>
message-id = <id da mensagem>
count= <número de valores a serem ordenados>

4) Mensagem de resultado de tarefa

```
<task-result sender="G2" destination="G1" message-id="07" response-to="05">
        <numbers>
            <n>1</n>
            <n>12</n>
            <n>50</n>
            <n>125</n>
            <n>234</n>
            <n>867</n>
        </numbers>
</task-result>
```

sender = <id do grupo que está enviando a mensagem>
destination = <id do grupo que vai receber a mensagem>
message-id = <id da mensagem>
response-to = <id da mensagem que originou esta resposta>

Cronograma

ATENÇÃO: ao final do trabalho, cada componente do grupo informará ao professor, de forma anônima, o percentual de participação de cada componente do grupo no trabalho. A nota final de cada componente do trabalho será modificada em função disso. Portanto, seja pró-ativo e não deixe o trabalho de lado! Sua nota depende disso!

14 e 16 de Setembro: Apresentar o andamento do trabalho. Nesta etapa os alunos já devem ter um plano de como farão o controle das informações (tempo de processamento de cada tarefa, monitoramento, lista de peers online e offline). Apresentar um diagrama com os módulos do sistema e um protótipo da interface.

28 de Outubro: Apresentar o andamento do trabalho. Nessa etapa o processamento das tarefas já deve estar funcionando. O sistema deve ser capaz de ler o documento XML de tarefa e processar a tarefa (ainda sem delegação a outros peers). A interface de monitoramento já deve ser capaz de mostrar o status das tarefas em andamento (nesta etapa serão sempre tarefas locais). Mostrar o arquivo de tarefa com os números gerados aleatoriamente, e o arquivo com a resposta da tarefa.

25 e 30 de Novembro: Apresentar o trabalho completo

- Demonstração prática do trabalho → essa apresentação não será feita no projetor. Todos os grupos colocarão seus trabalhos para funcionar ao mesmo tempo, e a professora passará de grupo em grupo para analisar o funcionamento do trabalho.
- Entrega do relatório final
- Entrega do código fonte

Relatório Final (Entrega dia 25 de Novembro)

O relatório final do trabalho deve conter o seguinte:

- Diagrama de classes
- Funcionamento interno da aplicação, com todas as decisões de projeto
- Manual de instalação e do usuário
- Dificuldades encontradas
- Sugestões de trabalhos futuros