

Trabalho Prático 1 – PPD Groups

Neste primeiro trabalho da disciplina de Programação Paralela e Distribuída, você terá que implementar uma aplicação cliente/servidor, o **PPD Groups**. O objetivo deste sistema é oferecer um serviço semelhante ao *Google Groups*, onde grupos de usuários se comunicam através de mensagens.

O sistema deverá ser composto por um servidor que centralizará todas as mensagens trocadas entre os clientes. Para isto, cada cliente terá no servidor um arquivo contendo as mensagens não lidas e já lidas por este cliente. Estes arquivos deverão seguir o seguinte padrão: `<id_usuario>_naolidas.txt` e `<id_usuario>_lidas.txt`. A Tabela 1 mostra um exemplo da organização destes arquivos para três usuários:

Tabela 1: Organização dos arquivos de mensagens.

| Usuário | Arquivo de mensagens lidas | Arquivo de mensagens não lidas |
|----------|----------------------------|--------------------------------|
| joao | joao_lidas.txt | joao_naolidas.txt |
| maria | maria_lidas.txt | maria_naolidas.txt |
| colorado | colorado_lidas.txt | colorado_naolidas.txt |

Adicionalmente, o servidor deverá conter um arquivo especial que armazenará todas as mensagens globais, ou seja, aquelas mensagens enviadas para todos os usuários (e não para um usuário específico). Este arquivo deverá ter o nome *globais.txt*. **Importante:** há necessidade de um controle sobre quais mensagens globais já foram lidas por cada cliente.

O servidor, componente que centralizará todo o controle do sistema, oferecerá diversos serviços que serão invocados pelo cliente e enviados a ele através de chamadas de procedimento remotas (RPC). Cada cliente terá os seguintes comandos disponíveis:

- **\connect <id_usuario>:** informa ao servidor que o usuário `<id_usuario>` está conectado. Automaticamente deverá ser informado ao cliente se existem mensagens novas para o mesmo e quantas são.
- **\send <mensagem> <id_usuario>:** envia a `<mensagem>` para o usuário `<id_usuario>`. Em outras palavras, este comando faz com que o servidor insira no arquivo `<id_usuario>_naolidas.txt` o texto inserido no parâmetro `<mensagem>`.
- **\sendall <mensagem>:** envia a `<mensagem>` para todos os usuários, ou seja, informa ao servidor que este deverá escrever a `mensagem` no arquivo `globais.txt`.
- **\sendgroup <mensagem> <id_usuario_1> ... <id_usuario_n>:** envia a `<mensagem>` para os usuários especificados na linha de comando. Em outras palavras, este comando faz com que o servidor insira nos arquivos de mensagens não lidas de cada usuário especificado na linha de comando o texto inserido no parâmetro `<mensagem>`.
- **\readnew:** imprime somente as mensagens que ainda não foram lidas para este cliente (separando em individuais e globais). A partir deste momento, mensagens individuais já lidas vão para o arquivo `<id_usuario>_lidas.txt` e as globais devem ser assinaladas de alguma forma como já lidas (Obs.: não é permitida a criação de um novo arquivo para conter as mensagens globais já lidas de um cliente.)
- **\readold:** imprime somente as mensagens que já foram lidas por este cliente (separando em individuais e globais).

- **\readglob:** imprime todas as mensagens do arquivo global, assim como indica para cada mensagem quais usuários já as leram. Ao contrário da função \readnew, não altera o status das mensagens não lidas. Este comando pode ser executado por qualquer usuário conectado e imprime **todas** as mensagens do arquivo globais.txt, inclusive aquelas que não pertencem ao usuário que fez a requisição.
- **\deleteold:** apaga todas as mensagens enviadas para o usuário que já foram lidas inclusive as globais (mas nesse caso não as retira do servidor, pois outros clientes ainda podem querer acessá-las).
- **\deletglob:** apaga todas mensagens globais armazenadas no arquivo *globais.txt*.
- **\deleteall:** apaga todas as mensagens armazenadas em arquivos do servidor independentemente de usuário.

Todos os arquivos contendo as mensagens deverão permanecer no servidor. Não será permitida a criação de arquivos locais nos clientes. Toda a lógica e controle das mensagens deverão ser feitos no servidor. Nenhum arquivo além dos especificados deverá ser criado adicionalmente.

Observações

1. Para este trabalho você deverá utilizar RPC (usando a linguagem de programação C/C++). O sistema operacional fortemente recomendado é o Linux.
2. Não é permitido o uso de *threads*.
3. Deve haver transparência de localização dos processos envolvidos (servidor e clientes). Os processos podem funcionar tanto local como remotamente.

Avaliação

1. 10% Interface (log no servidor, clareza dos comandos e das informações dos clientes)
2. 20% Funcionamento comprovado da comunicação de pelo menos um cliente com o servidor (**comando connect**)
3. 60% Funcionamento dos demais comandos previamente listados
4. 10% Documentação* (estrutura da implementação, execução, outras questões relevantes)

*A parte da nota relativa à documentação só será considerada caso o funcionamento do programa contemple pelo menos a comunicação RPC entre os processos (item 2). A documentação deve **obrigatoriamente** incluir a descrição de como deve ser executado o programa do grupo e uma explicação sobre a implementação.

A entrega do trabalho será feita por todos os grupos até o dia 20/10. Além disso, cada grupo deverá fazer uma apresentação em sala de aula laboratório (no dia da entrega). Nesse dia, todos os participantes do grupo deverão estar presentes (participantes ausentes ficarão sem a nota do trabalho). O grupo deverá entregar a documentação e enviar para o e-mail do professor o código fonte do programa após a apresentação. **Trabalhos com atraso serão aceitos mediante um desconto de 2.0 pontos na nota final por dia de aula de atraso.**

Este trabalho deverá ser feito em grupo de 3 componentes.