

# INF01151 - Sistemas Operacionais II - Trabalho Prático - Etapa 2

## 1 Especificação do Trabalho Prático

A proposta de trabalho prático é implementar um serviço distribuído de transferência de valores entre clientes, simulando um sistema de pagamento similar ao PIX. O objetivo do serviço é receber requisições de clientes diversos contendo um valor a ser transferido para outro cliente, processar cada requisição, atualizar os saldos dos clientes envolvidos e manter um histórico das transações. Cada requisição conterá um valor inteiro positivo, que será lido pelo cliente a partir da entrada padrão. O servidor deverá processar as transações de forma concorrente/paralela, usando threads (uma thread para processar cada requisição recebida).

A proposta deverá ser desenvolvida em duas etapas. A primeira etapa compreendeu funcionalidades implementadas usando conceitos como threads, processos, comunicação e sincronização. Na segunda etapa, a equipe deverá estender o serviço com algumas funcionalidades avançadas, onde destaca-se: esquema de replicação com eleição de líder. A aplicação deverá executar em ambientes Unix (Linux), mesmo que tenha sido desenvolvida em outras plataformas. O projeto deverá ser implementado em C/C++, usando a API User Datagram Protocol (UDP) sockets do Unix.

O principal critério de avaliação será a correta atualização dos saldos dos clientes e a consistência do histórico de transações, garantindo que o valor total do sistema permaneça constante. Um critério secundário de avaliação será o tempo de resposta para o servidor receber, processar e responder um grande volume de requisições (10.000+ transações) recebidas de clientes diversos.

## 2 Funcionalidades Avançadas

Na segunda parte deste projeto, a aplicação deverá ser estendida para atender os conceitos estudados na segunda metade da disciplina: replicação e eleição de líder.

### 2.1 Replicação Passiva

A estação servidora, caso venha a falhar, levará à indisponibilidade do serviço distribuído de transferência de valores. Este é um estado que não é desejável no serviço. Para aumentar a disponibilidade do serviço na ocorrência de falhas da estação servidora, uma nova estação servidora deverá assumir o seu papel e manter o serviço operacional. Note que essa mudança deve ser transparente para os participantes, os quais devem ser notificados de qual a nova estação servidora. Para garantir que os saldos dos clientes e o histórico de transações continuarão consistentes, você deverá utilizar um esquema de replicação, informando todas as modificações realizadas aos servidores secundários.

Mais especificamente, você deverá implementar um esquema de Replicação Passiva, onde o servidor é representado por uma instância primária de *replica manager* (RM), e uma ou mais instâncias de RM secundárias (ou *backup*). Seu esquema precisará garantir que:

1. Todos os clientes sempre utilizarão a mesma cópia primária (RM primário);
2. Após cada operação de transferência, o RM primário irá propagar o estado atualizado dos saldos e o histórico de transações aos RMs de *backup*;
3. A conservação do valor total do sistema seja mantida em todas as réplicas.

### 2.2 Eleição de Líder

Algoritmos de eleição de líder permitem escolher, dentro de um conjunto de processos distribuídos, qual processo deve desempenhar um papel particular (e.g., coordenador, alocador de recursos, verificador, etc.). Algoritmos de eleição de líder são muito usados como parte de outros algoritmos distribuídos, que exigem a escolha de um processo para desempenhar um papel específico. Na primeira parte do trabalho, assumiu-se a existência de uma única estação servidora. Agora, no entanto, como diferentes servidores potencialmente poderão assumir o papel de *manager*, o processo escolhido em questão deverá ser selecionado através do algoritmo de eleição de líder valentão.

Quando a estação servidora principal falhar, o algoritmo de eleição de líder deverá ser utilizado para determinar o próximo *manager*. Nesse caso, um dos servidores secundários deverá assumir essa função, mantendo um estado consistente do sistema. Para isso, implemente o algoritmo do *valentão* para eleger um novo RM primário após uma falha. Lembre-se de garantir que seu mecanismo atualize as informações sobre o novo líder nos clientes, permitindo que continuem a enviar requisições de transferência sem interrupção prolongada do serviço. A comunicação entre servidores e réplicas pode ser realizada tanto via *unicast* quanto *multicast*.

### 3 Descrição do Relatório

A equipe deverá produzir um relatório fornecendo os seguintes dados:

- Descrição do ambiente de testes: versão do sistema operacional e distribuição, configuração da máquina (processador(es) e memória) e compiladores utilizados (versões);
- Explicação e respectivas justificativas a respeito de:
  1. Funcionamento do algoritmo de eleição de líder implementado, justificando a escolha;
  2. A implementação da replicação passiva na aplicação, incluindo os desafios encontrados;
- Também incluía no relatório uma descrição dos problemas que a equipe encontrou durante a implementação e como estes foram resolvidos (ou não).

A nota será atribuída baseando-se nos seguintes critérios: (1) qualidade do relatório produzido conforme os itens acima, (2) correta implementação das funcionalidades requisitadas, (3) qualidade do programa em si (incluindo a implementação da interface exatamente como solicitado na especificação do trabalho, documentação do código, funcionalidades adicionais implementadas, etc.) e (4) qualidade da apresentação (o que inclui domínio sobre o trabalho realizado por cada integrante da equipe).

### 4 Métodos de Avaliação

O trabalho deve ser feito em grupos de **TRÊS ou QUATRO INTEGRANTES**, idealmente mas não obrigatoriamente mantendo-se a mesma equipe da Parte 1. As pessoas participantes da equipe devem estar claramente identificadas no relatório e na apresentação. A avaliação do trabalho será pela análise da implementação, do relatório produzido e da apresentação. A ausência de um(a) integrante da equipe no dia da apresentação implicará em conceito zero para a pessoa ausente (salvo por motivos excepcionais como por ex. de saúde, que deverão ser registrados via junta médica da UFRGS).

A apresentação dos trabalhos e demonstração prática em laboratório dos sistemas implementados será realizada presencialmente, conforme o cronograma da disciplina. O funcionamento correto e esperado dos sistemas implementados é um critério importante de avaliação, conforme mencionado no início deste documento.

Faz parte do pacote de entrega os códigos-fonte da implementação, um tutorial de como compilar e executar os códigos e o relatório em um arquivo ZIP. A implementação deve estar funcional para demonstração durante a apresentação pela equipe, em laboratório. A compilação deverá ser feita via scripts automatizados (por ex., Makefile), de modo a facilitar o processo de avaliação do projeto submetido.

### 5 Dúvidas, Questionamentos e Sugestões

Alterações nas especificações do trabalho poderão ser realizadas em concordância com o professor. Adições e melhorias, como interface gráfica ou métodos adicionais, poderão ser realizadas desde que a funcionalidade básica seja mantida.

Dúvidas, questionamentos e sugestões podem ser enviados por e-mail ou pelo fórum do Moodle.

Professor Eder Scheid: [ejscheid@inf.ufrgs.br](mailto:ejscheid@inf.ufrgs.br)

Mestrando João Nunes (Turma A): [jdmnunes@inf.ufrgs.br](mailto:jdmnunes@inf.ufrgs.br)

Mestrando Guilherme Soares (Turma B): [gmsoares@inf.ufrgs.br](mailto:gmsoares@inf.ufrgs.br)