INSTITUTO FEDERAL
Paraná
Campus Pinhais

Curso: Bacharelado em Ciência da Computação **Disciplina:** Projeto de Sistemas Distribuídos
Trabalho 2, v2.0 29 de outubro de 2026

1 Introdução

Seu trabalho consiste em implementar um algoritmo clássico de sistemas distribuídos integrado a uma aplicação prática **simplificada** do mundo real. O objetivo é compreender, na prática, os desafios de coordenação, concorrência, consistência, ordenação e tolerância a falhas em sistemas distribuídos.

A aplicação desenvolvida deve resolver explicitamente os problemas teóricos discutidos em sala de aula através da implementação dos algoritmos correspondentes:

- Como os nós coordenam ações ou elegem um líder sem dependência de um ponto único de falha?
- Como o sistema garante a ordenação de eventos ou a exclusão mútua distribuída?
- Como o sistema gerencia a replicação de dados e mantém a consistência entre as réplicas?
- Como o sistema detecta e reage a falhas de comunicação ou queda de nós em tempo de execução?

Note que nem todas as aplicações resolvem todos os problemas acima. Isso não é um problema. Faz parte do seu trabalho pesquisar, entender e relatar quais problemas são relevantes para a aplicação e por quê, e como são resolvidos.

Perceba também que você pode identificar outras perguntas interessantes no contexto de sistemas distribuídas. Inclua essas novidades no seu trabalho.

A descrição das soluções para os problemas devem ser técnicas e completas, dizendo quais algoritmos ou ideias são usadas, explicando as ideias e porquê elas realmente funcionam.

1.1 Sugestões de aplicações para pesquisa e os principais problemas envolvidos

- Apache Zookeeper (eleição de líder);
- Kubernetes (eleição de líder);
- Ceph (sistema de arquivos, replicação e consenso, exclusão mútua);
- Lustre (sistema de arquivos para computação de alto desempenho, replicação e consenso, exclusão mútua);
- Cassandra (banco de dados, ordenação de eventos);
- MongoDB (banco de dados, eleição de líder, replicação e consenso);
- PostgreSQL (banco de dados, replicação e consenso);
- Apache Kafka (filas de eventos distribuídas, eleição de líder, replicação e consenso);

- Matrix/Slack (chat distribuído, ordenação de eventos);

Este professor ressalta que, em geral, ferramentas de software livre possuirão mais discussões e detalhes sobre seu funcionamento interno, que podem enriquecer e facilitar o trabalho.

As ideias acima servem como ponto de partida para pesquisar cenários de uso de sistemas distribuídos e para pesquisar os algoritmos e estratégias envolvidos. Você **não deverá re-implementar** os sistemas acima como um todo. O que você deve fazer é planejar uma simulação e implementar os algoritmos de sistemas distribuídos que resolvem o problema do cenário escolhido, de maneira simulada.

Por exemplo, para o cenário/caso de uso escolhido, quais informações são relevantes para o sistema? Simule essas informações como variáveis, ou argumentos para o programa, e deixe o sistema trocar mensagens e resolver o problema. Mostre na tela as mensagens que estão sendo trocadas, para que possamos acompanhar o andamento dos algoritmos.

Vocês (estudantes) são encorajados a pesquisar outras ferramentas que sejam do seu interesse para este trabalho. Nesse caso, lembre-se de discutir a ideia com o professor, para garantir que sua proposta está dentro do escopo e dos objetivos do trabalho.

Você pode implementar os algoritmos e a sua aplicação na linguagem que preferir.

2 Entrega

Você deverá enviar para o email do professor, até 02/dezembro, 23h59, um arquivo *.zip* contendo uma pasta, que por sua vez contém:

- Os arquivos de código fonte da sua aplicação e dos algoritmos distribuídos;
- Um arquivo de texto chamado `README.md` contendo o nome completo dos integrantes do grupo e informações sobre como compilar e executar os programas;
- Um relatório chamado `relatorio.pdf` contendo o nome completo dos integrantes do grupo e o conteúdo descrito na seção 2.1.

O arquivo compactado descrito acima deve ser nomeado com as iniciais do nome de cada integrante separados por um hífen. Por exemplo: se o Fulano da Silva Sousa fez o trabalho com o João de Souza, o arquivo deve ser nomeado `fss-js.zip`. A pasta, dentro do arquivo compactado, deve ter o mesmo nome, a não ser pela extensão.

O trabalho pode ser feito em grupos de até 3 pessoas.

2.1 Relatório

A equipe deve entregar um relatório no formato `.pdf`, explicando como resolveu os desafios do trabalho. O relatório não deve conter o código completo! Só adicione código no relatório se precisar explicar/justificar algo muito específico da sua solução. Ainda assim, adicione apenas o trecho relevante.

O relatório deve explicar o problema escolhido, quais são os desafios envolvidos, como são resolvidos em aplicações reais e como a equipe resolveu esses problemas neste trabalho.

O relatório deve conter todas as informações necessárias para reproduzir a sua solução.

Inclua as referências de pesquisa e código que utilizar para seu trabalho no final de seu relatório.

2.2 Critérios de correção

Seu trabalho será corrigido considerando:

- Seus programas devem funcionar corretamente!
- Objetividade e clareza do relatório.
- Uso dos métodos e construções adequadas de programação e de algoritmos de sistemas distribuídos.
- Domínio do trabalho na defesa/acompanhamento.
- Adequação à esta especificação.

Não serão aceitos trabalhos entregues fora do prazo. Trabalhos sem relatório ou que não sejam defendidos/acompanhados em sala não serão considerados!

Plágio não será tolerado. Se isto acontecer, os trabalhos envolvidos serão desconsiderados e o caso será encaminhado para as Coordenações do campus.

O uso de inteligência artificial pode acarretar na redução de nota parcial ou totalmente, se o professor identificar que não há domínio do trabalho proposto por parte do estudante.

Histórico das Revisões:

- 29/out/2026 - v2.0: segunda versão, adicionando a parte de implementação prática.
- 17/nov/2025 - v1.0: primeira versão.