SISD - Sistema Integrado Simulado de Distribuição

Objetivo Geral:

Desenvolver um sistema distribuído completo que simule uma plataforma de consulta e monitoramento de sensores climáticos remotos, utilizando os principais conceitos e técnicas de sistemas distribuídos.

Descrição Geral:

O sistema será composto por:

- Múltiplos sensores (servidores) simulando dados climáticos (temperatura, umidade, pressão etc.).
- Um cliente principal que coleta dados dos sensores e exibe informações ao usuário.
- Comunicação via Sockets, gRPC, Multicast e técnicas de replicação, sincronização, eleição, tolerância a falhas e segurança.
- A estrutura deve ser capaz de tolerar falhas, sincronizar o tempo dos sensores, e garantir consistência eventual nas respostas.

Módulos do Projeto e Correspondência com Conteúdos

1. Fundamentos e Arquitetura do Sistema

- Introdução aos Sistemas Distribuídos - Conceitos e Objetivos
- Aspectos de Projeto e Modelos de Sistemas Distribuídos
- Modelo Cliente-Servidor
- Explorando Modelos com Exemplos (documentação de arquitetura)

Atividade: Documentar a arquitetura geral do sistema proposto, os modelos utilizados e as decisões de projeto.

2. Comunicação entre Componentes

- Implementação Básica de Cliente-Servidor com Sockets
- Sockets e Middlewares em Sistemas Distribuídos
- Middlewares com RPC/gRPC
- Invocação Remota de Métodos (RMI)
- Implementação de RMI em Java ou Python

Atividade: Implementar a comunicação entre cliente e sensores (servidores) com Sockets e gRPC. Criar interfaces para chamadas remotas com RMI.

3. Sincronização e Estado Global

- Algoritmos Distribuídos - Clocks e Sincronização de Tempo
- Implementação de Clocks Lógicos (Lamport)
- Estado Global e Captura de Estado
- Implementação de Captura de Estado Global (Snapshot)

Atividade: Adicionar timestamps Lamport aos eventos de leitura. Criar mecanismo de snapshot global para salvar o estado da rede.

4. Gerência de Falhas e Eleições

- Algoritmos de Eleição - Bully e Anel
- Simulação de Algoritmo de Eleição (Bully)
- Detecção de Falhas em Sistemas Distribuídos
- Implementação de Detecção de Falhas (Heartbeat)

Atividade: Implementar eleição do coordenador dos sensores (nó mestre). Monitorar a falha de sensores com heartbeat.

5. Comunicação em Grupo

- Comunicação em Grupo e Multicast
- Implementação de Comunicação em Grupo com Multicast

Atividade: Implementar mecanismo para envio de alertas climáticos por multicast aos sensores e clientes.

6. Computação em Nuvem

- Computação em Nuvem - Introdução e Paradigmas
- Implementação Básica com Computação em Nuvem

A*tividade:* Hospedar parte do sistema (coletor ou banco de dados) em uma instância cloud (pode ser simulada com Docker/VM).

7. Tolerância a Falhas e Concorrência

- Tolerância a Falhas - Replicação de Dados e Controle de Concorrência
- Replicação de Dados e Consistência Eventual
- Controle de Concorrência e Exclusão Mútua Distribuída
- Implementação de Exclusão Mútua Distribuída

Atividade: Implementar replicação de sensores e controle de concorrência ao registrar novos dados.

8. Recuperação e Segurança

- Recuperação de Falhas - Checkpoints e Rollback
- Implementação de Checkpoints para Recuperação
- Segurança em Sistemas Distribuídos - Autenticação e Criptografia
- Implementação de Autenticação e Criptografia Simples

Atividade: Implementar mecanismo de salvamento de estado periódico (checkpoints) e autenticação básica entre cliente e sensores usando criptografia simples (RSA/AES).

Resumo dos Componentes do Projeto

1. Fundamentos e Arquitetura

- Documentação do modelo de sistema distribuído proposto
- Modelos utilizados: cliente-servidor, multicamadas, RMI, gRPC

2. Comunicação entre Componentes

- Implementação com Sockets (TCP)
- Middleware com gRPC
- Invocação Remota (RMI com Java/Python)

3. Sincronização e Estado Global

- Clocks lógicos (Lamport)
- Captura de estado (Snapshot global)

4. Eleição e Detecção de Falhas

- Algoritmo de eleição (Bully ou Anel)
- Detecção de falhas com heartbeat

5. Comunicação em Grupo

- Multicast entre servidores e clientes

6. Computação em Nuvem

- Implementação parcial em nuvem (simulada ou real)

7. Tolerância a Falhas e Concorrência

- Replicação de dados e consistência eventual
- Exclusão mútua distribuída

8. Recuperação e Segurança

- Checkpoints e rollback
- Autenticação e criptografia simples

Tecnologias Sugeridas:

- Linguagem: Python ou Java (com uso de threads, RMI, sockets etc.)
- Frameworks: grpc, socket, threading, pickle, cryptography, multiprocessing
- Docker para simulação de múltiplas instâncias
- Possível uso de nuvem: AWS Free Tier, Google Cloud ou simulada

Entrega Final:

- Código-fonte comentado (organizado por módulos).
- Documentação técnica (arquitetura, modelos, justificativas).
- Apresentação com demonstração funcional (vídeo).
- Relatório final com análise crítica e sugestões de melhoria.

Critérios de Avaliação (Peso 4)

Critério	Pontuação Máxima
1. Arquitetura e documentação técnica do projeto	0,5
2. Implementação de comunicação básica (Sockets e RMI)	0,5
3. Implementação com Middleware (gRPC)	0,5
4. Sincronização e Captura de Estado (Lamport e Snapshot)	0,5
5. Eleição e Detecção de Falhas (Bully e Heartbeat)	0,5
6. Comunicação em Grupo (Multicast)	0,3
7. Tolerância a Falhas e Concorrência (Replicação, Mutex)	0,4
8. Segurança e Recuperação (Criptografia, Checkpoint)	0,3
9. Apresentação oral ou em vídeo do projeto	0,3
Total:	4,0 pontos

Observações:

- O projeto pode ser feito em grupos de até 3 alunos.
- A avaliação será baseada em critérios técnicos e na demonstração funcional do sistema.
- Cada grupo deve entregar um repositório com o código, um relatório técnico, e uma apresentação (vídeo).

Data de Entrega: 16/04/2025

Respostas contendo todos os critérios a serem avaliados, devem ser enviados exclusivamente via email: felipe_silva@ifba.edu.br com a devida identificação do aluno, disciplina e turma.