INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA BAHIA CAMPUS - SANTO ANTÔNIO DE JESUS - BAHIA
CURSO DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISITEMAS

MARCELO DE JESUS

RONALDO CORREIA

Serviço de Mensagens Distribuídas com Consistência Eventual e Autenticação Básica

SANTO ANTONIO DE JESUS-
BA 2025

MARCELO DE JESUS

RONALDO CORREIA

Serviço de Mensagens Distribuídas com Consistência Eventual e Autenticação Básica

Relatório técnico da atividade de Serviço de Mensagens Distribuídas com Consistência Eventual e Autenticação Básica, elaborado como requisito parcial de avaliação para a disciplina de Sistemas Distribuídos, ministrada pelo Prof. Felipe Silva.

SANTO ANTONIO DE JESUS –

BA 2025

Sumário

Introdução

Este relatório tem como finalidade apresentar de forma teórica como foi realizado o desenvolvimento de um sistema distribuído simples de publicação e leitura de mensagens, dando possibilidade de visualizar na prática conceitos de consistência eventual, replicação assíncrona e autenticação básica. O sistema pode ser composto por vários nós independentes que mantêm uma cópia local de um mural compartilhado, comunicando-se entre si por meio de sockets TCP. Através dessa atividade realizada, foi notório a percepção dos desafios de sistemas distribuídos, como por exemplo tolerâncias a falhas, reconciliação de dados e controle de acesso, onde cada nó pode entrar em um modo de falha, deixando de receber mensagens, mas que a qualquer momento pode retornar, havendo assim uma recuperação e uma sincronização, para que nenhuma informação seja perdida.

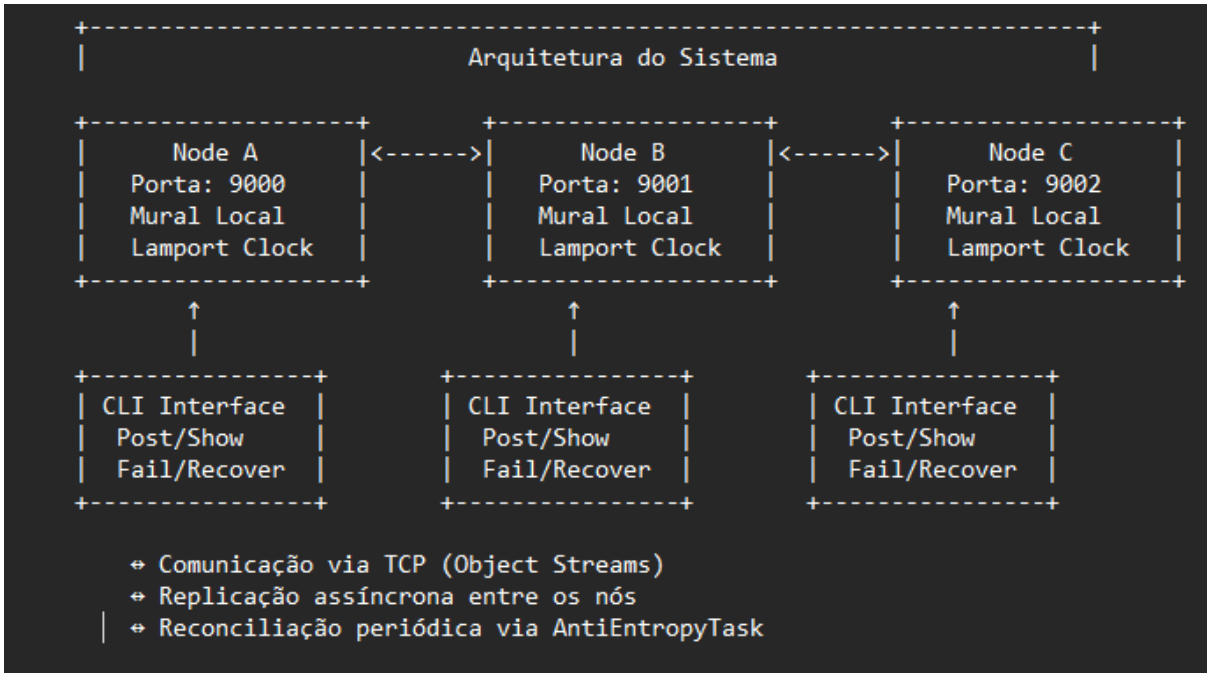
Arquitetura adotada

O sistema foi desenvolvido com base em uma arquitetura distribuída simples, composta por múltiplos nós que se comunicam entre si para manter um mural de mensagens replicado. A arquitetura foi projetada para garantir **tolerância a falhas**, **replicação assíncrona** e **consistência eventual**, conforme os requisitos da atividade.

Componentes Principais

- **Nós Distribuídos**: Cada nó é um processo independente que mantém uma cópia local do mural. Os nós são identificados por um nodeId e operam em portas distintas.
- **Cliente CLI**: Interface de linha de comando que permite ao usuário postar mensagens, visualizar o mural, simular falhas e recuperar nós.
- **Serviço de Mensagens (**MessageService**)**: Responsável por criar mensagens e iniciar a replicação.
- **Servidor de Replicação (**ReplicationServer**)**: Escuta conexões TCP e recebe mensagens de outros nós.
- **Gerenciador de Replicação (**ReplicationManager**)**: Realiza a reconciliação de mensagens recebidas.
- **Tarefa de Anti-Entropia (**AntiEntropyTask**)**: Executa periodicamente a sincronização entre os nós.

- **Simulador de Falhas (**FailureSimulator**)**: Permite colocar um nó em modo de falha e restaurá-lo.



Estratégia de replicação e consistência eventual

Estrutura da Mensagem

- Cada mensagem postada por um usuário é encapsulada em um objeto Message, contendo:
- **ID único** gerado via UUID
- **Timestamp físico** (System.currentTimeMillis)
- **Relógio lógico de Lamport**, para ordenação causal
- **Origem do nó** que criou a mensagem
- **Usuário e conteúdo textual**

Replicação Assíncrona

- Após a criação, a mensagem é armazenada localmente com NodeState.addMessage() e enviada aos peers por meio do ReplicationClient. A replicação ocorre de forma **assíncrona**, ou seja, o nó não aguarda confirmação imediata dos demais, permitindo maior resiliência e desempenho.

Reconciliação via Anti-Entropy

- Para garantir a **consistência eventual**, o sistema executa uma tarefa periódica chamada AntiEntropyTask, que:
- Roda a cada 5 segundos
- Envia todas as mensagens locais para os peers
- Recebe mensagens dos peers em resposta
- Executa reconciliação com ReplicationManager.reconcile(), adicionando mensagens ainda não recebidas
- Esse processo é tolerante a falhas e garante que, com o tempo, todos os nós compartilhem o mesmo conjunto de mensagens.

 Tolerância a Falhas

Cada nó pode entrar em **modo de falha** (failMode = true), simulando indisponibilidade. Durante esse período, o nó não recebe mensagens. Ao se recuperar (failMode = false), ele é automaticamente sincronizado com os demais por meio da tarefa de anti-entropy, recebendo todas as mensagens perdidas.

Garantia de Consistência Eventual

- Mesmo com atrasos, desconexões ou falhas temporárias, o sistema garante que:
- Nenhuma mensagem é perdida
- Todos os nós convergem para o mesmo estado
- A ordenação lógica é preservada via relógio de Lamport

Explicação da autenticação implementada

O sistema desenvolvido tem como um modelo simples de autenticação, baseado em usuário e senha, com o objetivo de não permitir que mensagens sejam postadas sem que haja autenticação de usuário, no entanto é público a visualização de mensagens mesmos por usuários não logados. A estrutura do serviço de autenticação foi definida e desenvolvida na classe AuthService, responsável por gerenciar cadastros de usuários, logins, verificação de autenticação, logout e integração com os serviços de mensagens. Segue abaixo o detalhamento de cada parte de sua estrutura:

- Cadastro de Usuários:
  O método register, permite criar novos usuários, armazenando suas credenciais em um mapa de usuário. Antes do registro, a existência do usuário é verificada pelo método userExists para evitar duplicações.
- Login:
  O método login, verifica se as credenciais fornecidas correspondem a um usuário registrado. Caso o login seja bem-sucedido, o usuário é adicionado ao conjunto de usuários logados (loggedUsers), permitindo que ele poste mensagens no mural.
- Verificação de Autenticação:
  A função isAuthenticated, retorna um valor booleano indicando se o usuário já está logado, garantindo que apenas usuários autenticados possam postar mensagens.
- Logout:
  O método logout, remove o usuário do conjunto loggedUsers, revogando seu acesso para postagem até que ele realize um novo login.
- Integração com o Serviço de Mensagens

O serviço de mensagens (MessageService) é integrado ao AuthService. Antes de postar uma mensagem, é realizada a verificação de autenticação:

```
// Cria e envia mensagem para o mural e peers
public void postMessage(String user, String text) {
    if (!auth.isAuthenticated(user)) {
        System.out.println(x:"❌ Usuário não logado! Mensagem não enviada.");
        return;
    }
```

Tendo em vista todos os pontos abordados acima, caso o usuário não esteja autenticado, a mensagem não é enviada, garantindo que apenas os usuários que estejam logados possam se comunicar através do mural. Dessa forma o sistema possui uma parcela de segurança para o controle de acesso, garantindo que a leitura do mural seja pública, porém a postagem seja restrita aos usuários cadastrados. Exemplos de funcionamento:

Usuário ainda não tinha cadastro:

```
login
Usuário: marcelo
Senha: ? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
123
? Usuário ou senha inválidos!
```

Usuário efetuado registro

```
register
Novo usuário: marcelo
Senha: ? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
123
? Usuário registrado com sucesso!
Comando (register/login/post/logout/show/fail/recover):
```

Usuário efetuando login após cadastro

```
Comando (register/login/post/logout/show/fail/recover): login? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
Usuário: marcelo? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
Senha: 123iação com 127.0.0.1:9002: recebidas 0 msgs.
? Login bem-sucedido!
Comando (register/login/post/logout/show/fail/recover): ? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
```

Evidências (prints/logs) da simulação de falha e recuperação

```
C:\Windows\system32\cmd.e
ReplicationServer rodando na porta 9001
A rodando na porta 9001 com peers=127.0.0.1:9002,127.0.0.1:9003
Comando (fail/recover/show/post): ? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
Erro ao conectar com 127.0.0.1:9003
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
Erro ao conectar com 127.0.0.1:9003
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
fail
?? Nodo em modo de falha!
Comando (fail/recover/show/post): recover
? Nodo recuperado!
Comando (fail/recover/show/post): ? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
post
Usuário: ? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
ronaldo
Mensagem: oi
? Mensagem enviada!
Comando (fail/recover/show/post): ? Reconciliação com 127.0.0.1:9002: recebidas 1 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 1 msgs.
sh? Reconciliação com 127.0.0.1:9002: recebidas 1 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 1 msgs.
ow
Mensagens armazenadas:
ronaldo: oi [1]
Comando (fail/recover/show/post): ? Reconciliação com 127.0.0.1:9002: recebidas 1 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 1 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 1 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 1 msgs.
```

```
C:\Windows\system32\cmd.e
ReplicationServer rodando na porta 9002
Erro ao conectar com 127.0.0.1:9001
B rodando na porta 9002 com peers=127.0.0.1:9001,127.0.0.1:9003
Erro ao conectar com 127.0.0.1:9003
Comando (fail/recover/show/post): ? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
Erro ao conectar com 127.0.0.1:9003
? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
Erro ao conectar com 127.0.0.1:9001
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
Erro ao conectar com 127.0.0.1:9001
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 1 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 1 msgs.
```

```
C:\Windows\system32\cmd.e
ReplicationServer rodando na porta 9003
C rodando na porta 9003 com peers=127.0.0.1:9001,127.0.0.1:9002
Comando (fail/recover/show/post): ? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
Erro ao conectar com 127.0.0.1:9001
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
Erro ao conectar com 127.0.0.1:9001
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 0 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 1 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 1 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 1 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 1 msgs.
```

```
Usuário: marcelociliação com 127.0.0.1:9002: recebidas 1 msgs.
Mensagem: ola tubem liação com 127.0.0.1:9002: recebidas 1 msgs.
? Mensagem enviada! 127.0.0.1:9003: recebidas 1 msgs.
Comando (register/login/post/logout/show/fail/recover): ? Reconciliação com 127.0.0.1:9002: recebidas
2 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 2 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 2 msgs.
? Reconciliação com 127.0.0.1:9003: recebidas 2 msgs.
```

```
? Reconciliação com 127.0.0.1:9001: recebidas 1 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 1 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 1 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 1 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 1 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 1 msgs.
? Reconciliação com 127.0.0.1:9001: recebidas 2 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 2 msgs.
show
Mensagens armazenadas:
marcelo: ola tubem  [3]
marcelo: oi [1]
Comando (register/login/post/logout/show/fail/recover): ? Reconciliação com 127.0.0.1:9001: recebidas
2 msgs.
? Reconciliação com 127.0.0.1:9002: recebidas 2 msgs.
```

CONCLUSÃO

O desenvolvimento desta aplicação permitiu a aplicação prática de conceitos fundamentais de sistemas distribuídos, tais como replicação de dados, consistência eventual e tolerância a falhas. A implementação demonstrou que é possível manter um mural de mensagens consistente entre múltiplos nós, mesmo em cenários de falhas temporárias, por meio da sincronização periódica (anti-entropy) e reconciliação das mensagens.

A autenticação implementada garante controle de acesso simples e efetivo, restringindo a postagem de mensagens a usuários registrados e logados, enquanto permite a leitura pública do mural. A interface de linha de comando mostrou-se funcional e intuitiva, permitindo simular falhas, recuperar nós, cadastrar novos usuários e gerenciar sessões.

Portanto, o trabalho atendeu aos objetivos propostos, integrando de forma eficiente os aspectos de backend, replicação distribuída e autenticação, e fornecendo evidências claras da operação correta do sistema. Além disso, foi de muita importância para o aprendizado sobre desafios reais enfrentados na construção de sistemas distribuídos resilientes.

LINK REPOSITORIO

- Ronaldo-Correia/ATV8-Servico-de-Mensagens-Distribuidas: Serviço de Mensagens Distribuídas com Consistência Eventual e Autenticação Básica: Ssistema distribuído simples de publicação e leitura de mensagens entre múltiplos nós, explorando consistência eventual em cenários de falha de comunicação e controle básico de autenticação.

REFERÊNCIAS

- **TANENBAUM, Andrew S.; VAN STEEN, Maarten.** *Distributed Systems: Principles and Paradigms*. 2nd edition. Prentice Hall, 2007.
- **Oracle.** “Java Secure Coding: Authentication and Password Storage.” Disponível em: https://www.oracle.com/java/technologies/javase/seccode.html. Acesso em: 07 set. 2025.
- **Red Hat Developers.** “Implementing Authentication in Java Applications.” Disponível em: https://developers.redhat.com/articles/2020/04/23/implementing-authentication-java. Acesso em: 07 set. 2025.