INSTITUTO FEDERAL
Santa Catarina

Projeto prático 2

Replicação mestre escravo

Curso: Engenharia de Telecomunicações
Disciplina: STD29006 – Sistemas Distribuídos
Professor: Emerson Ribeiro de Mello

Aluna
Sarom Torres

8 de dezembro de 2019

1 Introdução

Este relatório tem por objetivo apresentar a implementação de uma API REST para um sistema mestre/escravo que utiliza o protocolo 2PC para realizar a atualização de informações de contas bancárias em diferentes sistemas. O projeto foi desenvolvido na disciplina de Sistemas Distribuídos (STD29006) do curso de engenharia de telecomunicações do IFSC-SJ.

Na Seção 2 será feita uma breve apresentação do cenário desenvolvido, a Subseção 2.1 apresentará o papel de uma aplicação coordenador e na Subseção 2.2 explicará o papel de uma aplicação réplica, na Seção 3 serão comentadas as tecnologias utilizadas, e por fim na Seção 4 haverá uma lista das funcionalidades implementadas na API.

2 Apresentação

O cenário de implementação é composto por três máquinas que utilizam a mesma API REST para simular o protocolo 2PC. Sendo assim uma das máquinas desempenha o papel de coordenador enquanto as outras duas desempenham o papel de réplicas. Qualquer uma das aplicações pode ser coordenador ou réplica, sendo que a diferença entre elas é que o coordenador possui a lista de todas as réplicas.

2.1 Aplicação coordenador

A aplicação chamada de coordenador é responsável fazer a interação com o usuário e guardar em seu espaço de memória temporária uma lista de todas as réplicas que fazem parte do sistema. Além disso cabe a ela enviar às replicas ações de atualização de contas bancárias e aguardar a resposta de confirmação ou não da possibilidade de atualizar os dados. Assim que recebe a confirmação ela envia uma decisão a todas as réplicas de armazenar ou não os dados de forma permanente e devolve a resposta ao cliente.

2.2 Aplicação réplica

A aplicação chamada de réplica é responsável por manter cópias das contas bancárias atualizadas a partir das ações enviadas pelo coordenador. Ao receber uma ação a réplica confirma ou não a possibilidade de atualização das contas e guarda os dados em memória temporária. A partir disso aguarda a confirmação do coordenador e ao receber a confirmação persiste ou não as atualizações em memória. A aplicação réplica confirma a possibilidade de atualização em apenas 70% das requisições.

3 Tecnologias utilizadas

A API REST foi desenvolvida em Python3 e sua documentação foi realizada de acordo com a especificação API BluePrint.

4 Funcionalidades da API

Abaixo serão listadas as funcionalidades que foram implementadas na API:

1. A lista de contas é armazenada em um arquivo.
2. Retorna um JSON contendo a lista de contas com seus respectivos saldos atualizados. Esse recurso pode ser acessado tanto em réplicas quanto no coordenador.
3. A API carrega uma lista de réplicas e define o processo que a chamou como coordenador.
4. É possível apagar a lista de réplicas do coordenador, sendo que é removido seu status de coordenador automaticamente.
5. Apenas uma aplicação pode carregar a lista de réplicas por vez, sendo necessário apagar a lista de uma aplicação para inicializá-la em outra.
6. É possível obter a lista de réplicas do coordenador.
7. O processo coordenador recebe uma ação e repassa para as réplicas.
8. O processo réplica responde a 70% das requisições com *yes* e 30% com *no*.
9. Tanto o processo coordenador quanto o processo réplica armazenam a ação em área de memória temporária.
10. Após o coordenador enviar a decisão de confirmação os dados são persistidos em memória. Caso a decisão seja de cancelamento os dados são apagados da memória temporária.
11. Caso o identificador da ação não exista ele retorna o código HTTP 404 Not Found.
12. É possível obter um JSON do histórico das ações processadas com o identificador da ação e seus status de *success* ou *fail*.
13. É possível carregar a semente para geração de números pseudo aleatórios.
14. Ao final do processo de commit os três processos possuem os mesmo dados armazenados de forma permanente.