Sistemas distribuídos: Simulação de topologia anel e estrela entre processos

Antônio Galvão dos Santos Freitas

Resumo: O presente trabalho tem como o intuito desenvolver duas aplicações, onde iremos trabalhar na primeira na implementação de uma topologia em anel utilizando os protocolos TCP e UDP na criação de sockets que simulam o funcionamento de processos comunicando entre si no envio de mensagens, já na segunda iremos implementar uma topologia em estrela em que também deve dar suporte ao enviado de mensagem entre os processos e a broadcast. Tudo isso será implementado usando os conhecimentos e definições dos conteúdos aprendidos em sala de aula, tais como threads, sockets TCP e UDP.

Palavras-chave: processos, TCP, UDP, Threads, topologia, simulação.

1. INTRODUÇÃO

Vivendo em um mundo globalizado, a necessidade de comunicação torna-se fundamental nas diversas partes do mundo, bem como, o fornecimento de serviços que necessitam de grande desempenho para desempenhar o seu papel de forma eficaz, dessa forma, é visto a necessidade de trabalhar ao máximo com os recursos disponíveis e sempre procurar formas mais eficientes de fazer as coisas, visto isso, surgiu a necessidade de criar arquiteturas ou sistemas que permitissem cumprir os requisitos das aplicações cada dia mais exigentes no mercado, logo, foi desenvolvidos diversas tecnologias para se trabalhar e aplicá-las nas mais diversas áreas em que a computação trabalha, e isso fez surgir os sistemas distribuídos a qual iremos trabalhar nesse trabalho usando as tecnologias, conceitos e definições dessa área da computação.

Neste trabalho será implementado duas topologias em formato anel e estrela para aplicar os conhecimentos adquiridos em sala de aula sobre sistemas distribuídos, bem como, para aplicar as tecnologias também aprendidas, nas quais são: Threads, Sockets usando o protocolo TCP e UDP. Tudo isso será feito com o intuito de simular a comunicação entre processos nas topologias anteriormente descritas.

2. METODOLOGIA

O presente trabalho foi desenvolvido no Eclipse versão 2020-09 que é bastante completo, contando com debug, suporte para outras linguagens e ferramentas diversas. Nela foi usado a versão java 8 do JDK 1.8, uma versão bastante estável e amplamente usada e aceita pela comunidade.

A elaboração dos algoritmos se deu a partir da explicação dos vídeos gravados pelo professor da disciplina e de pesquisa realizada a outros exemplos disponíveis na internet, bem como acesso a livros e exemplos comentados.

Em ambos os problemas foi identificado primeiro a topologia a ser implementada, montado a lógica de utilização dos sockets em TCP ou UDP, que serviriam como cliente e servidor para realizarem a simulação na comunicação entre os processos e arranjados os mesmos para que funcionassem como programas separados na abstração de processos separados. Vale salientar que a metodologia foi imposta nas duas questões implementadas.

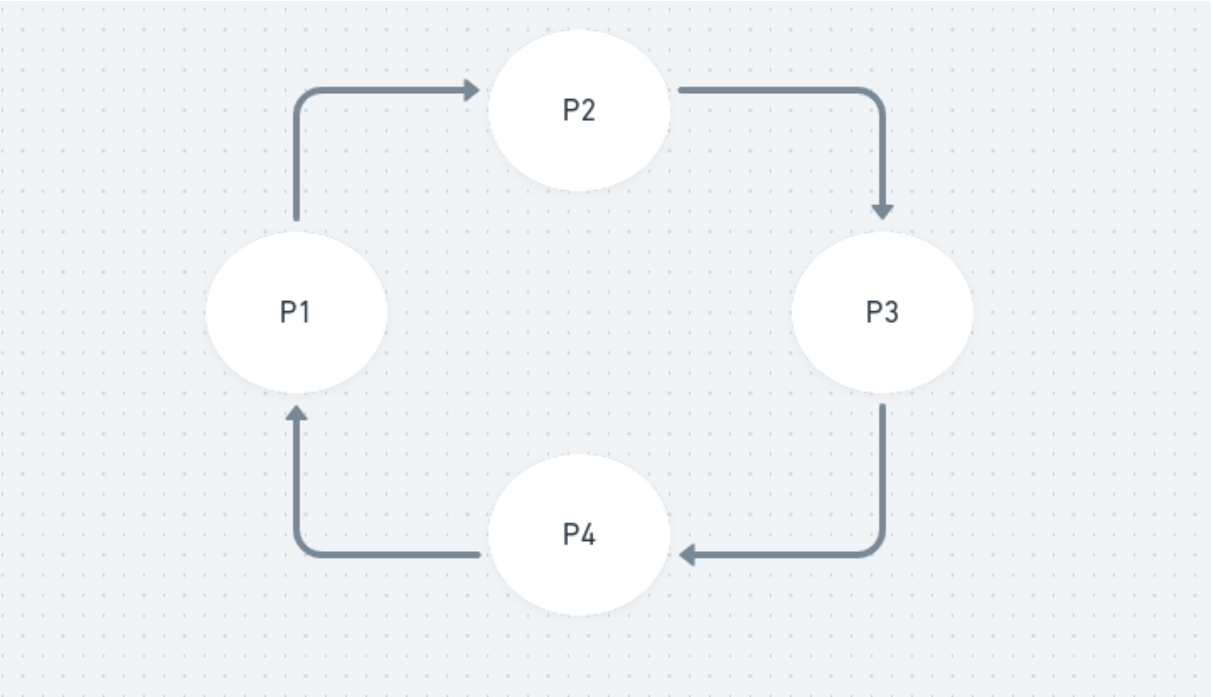
3. DESENVOLVIMENTO

Partindo da primeira implementação, deve ser criado uma simulação em topologia de anel para a comunicação de 4 processos, onde iremos ter o processo p1 conectado ao processo p2 e p4, p2 conectado a p1 e p3, p3 conectado a p2 e p4, e por fim p4 conectado a p3 e p1. Esses processos irão cumprir o papel de cliente e servidor ao mesmo tempo, onde por exemplo, p1 é cliente de p2, mas é servidor de p4, contudo, todos os processos podem se comunicar entre si enviando mensagens. Tanto na implementação com sockets TCP e UDP, a topologia e os requisitos serão os mesmos. Assim, ambas as implementações devem fazer segundo o trabalho:

- As mensagens podem ser enviadas por qualquer processo. A ideia é que elas circulem no anel de processos. Ou seja, se P1 enviar uma mensagem, ela deve passar por todos os proce ssos do anel até chegar novamente em P1.
- Basicamente, duas operações devem ser feitas no anel:
  - Se um processo enviar um número qualquer, cada processo ao receber a mensagem deve

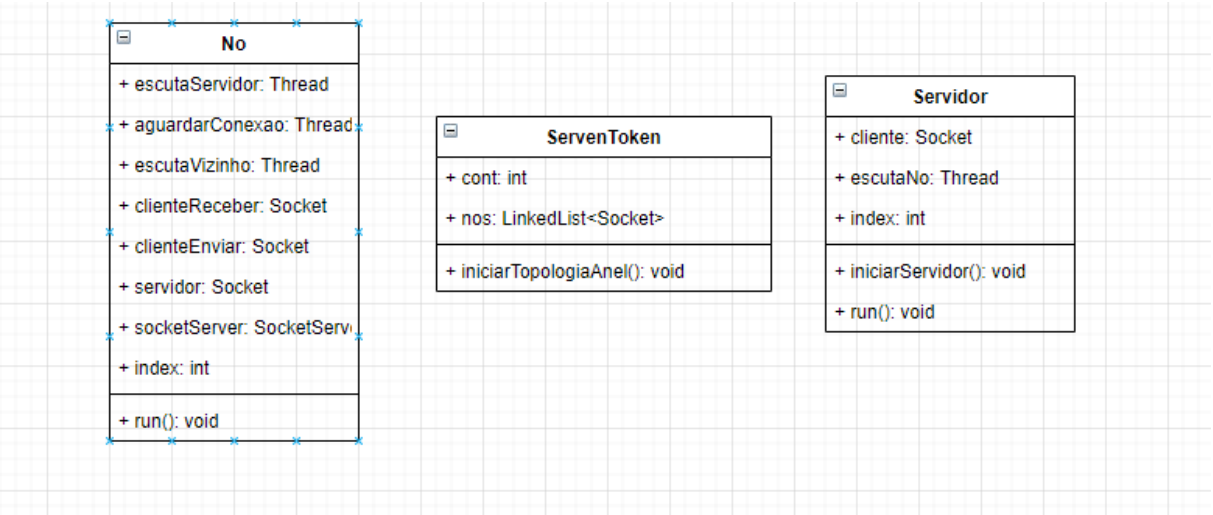
adicionar seu id ao número recebido. Ex.: P1 adiciona 1, P2 adiciona 2, etc.
◦ Se um processo enviar um caractere ou string, cada processo, ao receber a mensagem, deve concatenar seu id ao conteúdo recebido.

A seguir uma imagem mostrando como ficaria a topologia com os processos conectados, como foi descrita anteriormente:



fonte: autoria própria.

Abaixo segue como o código foi organizado para a implementação usando TCP e UDP da primeira questão:

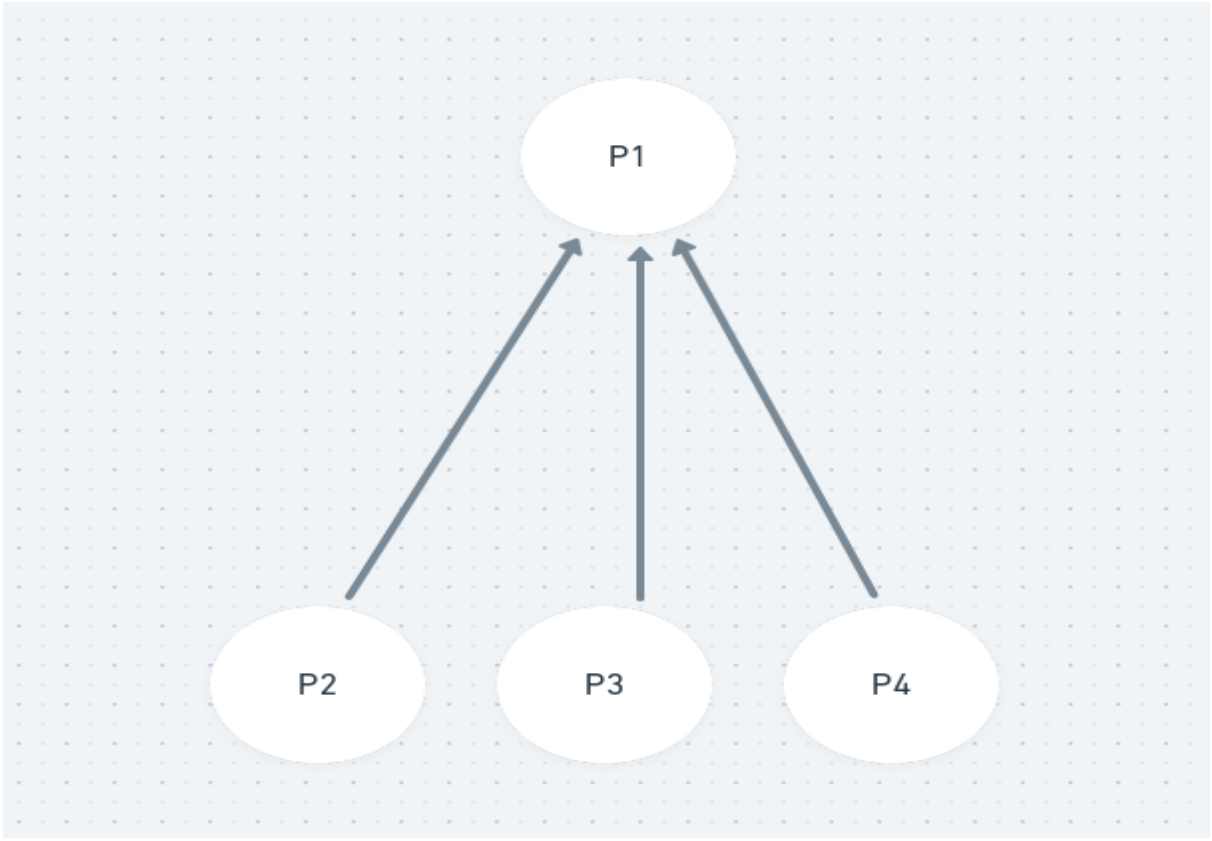


fonte: autoria própria.

Aqui será necessário executar o ServenToken primeiro e depois 4 vezes o Nó, pois irão cumprir o papel dos 4 processos durante a simulação. Vale salientar, que é necessário fechar a topologia com os 4 processos para que funcione, não irá conseguir enviar mensagem enquanto ela não estiver completa.

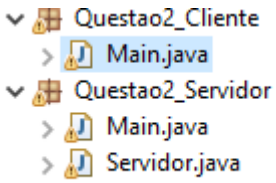
A segunda questão tem como intuito a implementação de uma topologia do tipo estrela, onde teremos 4 processos. Nessa topologia a comunicação entre os processos terá como elemento central o processo p1 e todos os outros estarão conectados entre si a partir dele, assim todas as mensagens trocadas entre os processos devem passar por p1. Essa implementação tem que dar suporte além ao unicast, envio de mensagem a um processo

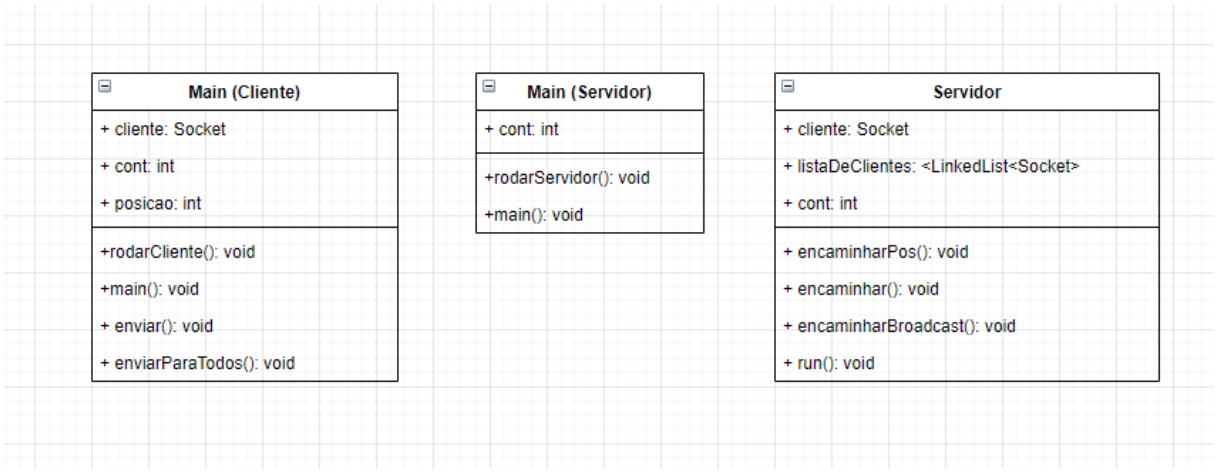
específico, como também ao broadcast, envio de mensagem para todos os processos na topologia ativos, abaixo segue como ficaria a topologia em estrela:



fonte: autoria própria.

Abaixo segue como o código foi organizado para a sua implementação, visto que foi separado em dois pacotes, onde um seria o servidor e o outro o cliente:

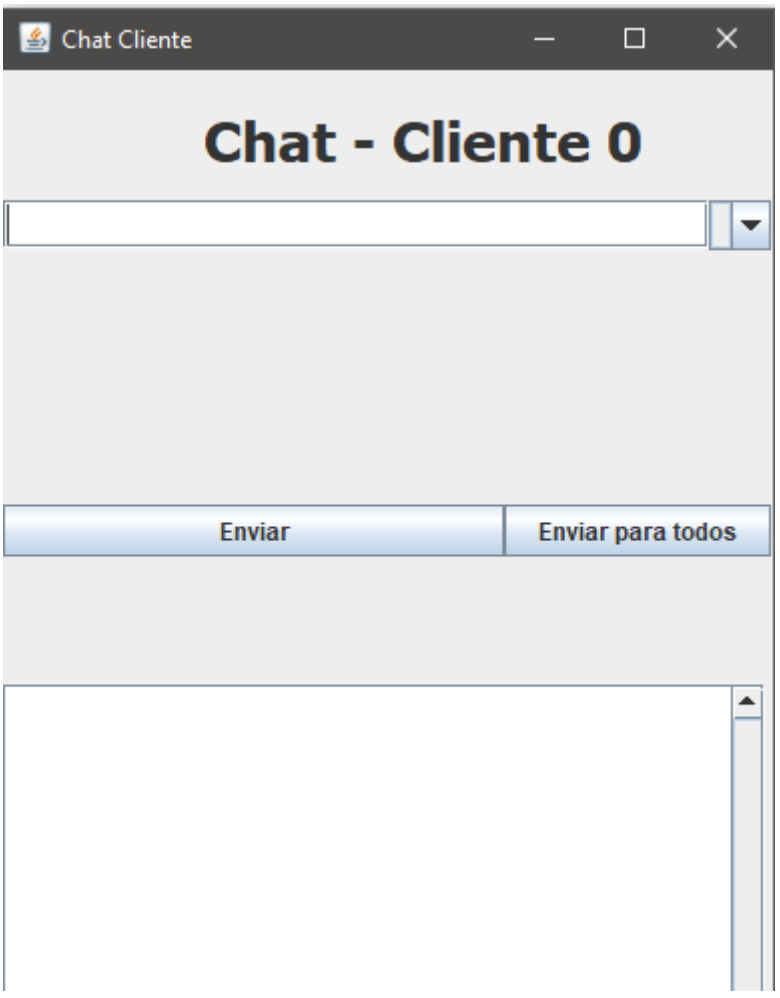




Aqui será necessário executar o Main (Servidor) primeiro e depois 4 vezes o Main (Cliente), pois irão

cumprir o papel dos 4 processos durante a simulação que irão se comunicar com o servidor para gerar a troca de mensagem entre os processos. Abaixo segue telas gráficas usadas para realizar o envio de mensagem e mostrar o status do servidor:

fonte: autoria própria.



fonte: autoria própria.

4. CONCLUSÕES

Ambas as questões implementadas estão funcionando corretamente e cumpriram os requisitos estabelecidos no trabalho. A aplicação se comportou de forma estável em todos os testes aplicados, sejam eles entre a comunicação entre os processos, quanto ao funcionamento tolerante aos erros.

Durante o desenvolvimento do trabalho senti certa dificuldade no trabalho com threads, visto que apesar de não ser um conceito novo adquirido no curso, sentir que a sua implementação com os protocolos na construção da simulação teve uma certa dificuldade devido a abstração de como tudo iria funcionar e pelo fato de ter alguns problemas técnicos, pois não tenho muita prática em trabalhar com java, tirando isso, o trabalho foi desenvolvido de forma bem tranquila, tanto que fiz o uso de interface gráfica para melhorar e deixar mais prática o funcionamento e uso da aplicação usando swing do java em contrapartida ao uso do javaFx queria daria um certo trabalho a mais.

Uma das lições aprendidas no trabalho foi a noção de como se trabalhar com threads e sua importância nos sistemas distribuídos, bem como, o uso de ferramentas que tomam uso dos protocolos de redes, tais como o TCP e UDP usados para implementar os sockets na comunicação entre os processos.

4. REFERÊNCIAS BIBLIOGRÁFICAS

Material disponibilizado pelo professor, slides e vídeo aulas, disciplina de sistemas distribuidos. Dr. Prof. Paulo Henrique, 2021.

Consultas a exemplos e implementações disponíveis no StackOverflow, 2021.