ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

A3 – SISTEMAS DISTRIBUIDOS E MOBILE

SALVADOR-BA

2023

ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

BRUNO DE SOUZA PEDROSA - 1272116295

CLERISTON PEREIRA - 1272116908

DANILO RAMOS COUTO - 1272116639

INGRYD SILVA MAGALHÃES - 1272116233

MICHEL CRUZ - 1272117133

Trabalho referente à A3 da UC Sistemas distribuídos e mobile, solicitado pelos professores Eduardo Xavier e Wellington Lacerda

SALVADOR- BA

2023

RELATÓRIO PROJETO A3

INTRODUÇÃO

O código em Python apresentado é parte de um aplicativo de gerenciamento de vendas que utiliza o protocolo UDP para comunicação entre clientes e servidor. O objetivo do aplicativo é permitir que vendedores enviem informações de vendas e que gerentes façam consultas sobre as vendas registradas.

O aplicativo utiliza uma arquitetura cliente-servidor. A arquitetura cliente-servidor é baseada em uma comunicação entre os dois componentes através de uma rede, como a Internet, onde o cliente é responsável por solicitar um recurso e o servidor fica a cargo de enviar uma resposta a esta solicitação.

O código também implementa um mecanismo de eleição para seleção de um servidor temporário caso o servidor principal esteja indisponível. Esse mecanismo faz parte da estratégia para que o serviço seja ofertado com o mínimo de perdas na comunicação. Quando o servidor deixa de funcionar, um cliente assume a função para manter o serviço ativo.

Outra estratégia utilizada para manter o padrão na comunicação é a utilização de menu para as escolhas de interação por parte do cliente. Este módulo é inserido e executado no código clientUDP.py, e gera um código passado para o servidor que por sua vez faz a interpretação do código e executa o serviço solicitado.

Requisitos de Software:

SO: Windows 10

Obrigatório utilizar o editor de código-fonte Visual Studio Code

Linguagem: Python

Bibliotecas: Socket, JSON, random, Sqlite3, datetime (importando date)

Instalação e Execução:

Certifique-se de ter o Python 3.11.4 instalado em seu sistema e configurado o path para o Python em seu computador.

Execute o arquivo principal (servidorUDP.py) no prompt de comando.

O último cliente a ser utilizado deve ser iniciado através do terminal no Visual Studio Code, utilizando a opção Debug Python File, permitindo ao usuário digitar os comandos necessários.

Em todos os testes de eleição realizados, o último cliente a se conectar ao servidor (cliente eleito) também foi o último cliente a iniciar a eleição. (A eleição é iniciada instantaneamente após a primeira tentativa de contato com o servidor indisponível).

Detalhamento do algoritmo:

O servidor principal:

O código do servidor apresenta uma função chamada ini_server_UDP que é responsável por iniciar o servidor UDP e processar as solicitações dos clientes. O servidor pode ser iniciado tanto como servidor principal quanto como servidor temporário, dependendo do valor do parâmetro from_client que determina quem chamou a função. Esta função também recebe parâmetro vazio de HOST e uma PORT padrão 9000. O HOST do servidor será atribuído através da função gethostbyname(), responsável por identificar o ip da máquina, não sendo necessário a consulta manual por parte do usuário. Após a identificação do HOST, o servidor chama uma função para armazenar seu HOST e PORT em um JSON exclusivo para servidor. Este arquivo servirá de base para que os clientes consultem os dados do servidor ativo e se conectem a ele.

O sistema conta com um arquivo JSON para armazenar os dados dos clientes. Este arquivo já conta com portas pré-determinadas iniciando da 9001 até a 9100.

Quando o cliente se conecta, envia para o servidor o seu endereço ip. O servidor armazena o endereço em um arquivo JSON, identifica a porta referente a este cliente e envia esta informação como resposta ao cliente.

Cria um socket UDP e o associa ao endereço e porta especificados.

Inicia uma conexão com o banco de dados SQLite3.

Aguarda as solicitações dos clientes.

Processa as solicitações de login, mensagens de consulta e entradas de venda.

Realiza as consultas no banco de dados conforme as solicitações dos gerentes.

Insere as entradas de venda no banco de dados.

De acordo com a solicitação, envia as respostas aos clientes solicitantes.

O cliente:

O cliente possui as opções de interação para gerente e vendedor. Todas as interações do usuário com o cliente se dão através de um menu contendo valores já programados. A única exceção é a inserção de valor da venda, onde é informado um padrão aceito para a inserção.

O cliente inicia com um HOST e PORT vazios e entra em uma condição que o força a se conectar ao arquivo JSON para consultar os dados do servidor. Após esta consulta o cliente inicia um socket e envia para o servidor a mensagem “LOGIN” que serve de gatilho para a inserção de suas credenciais no JSON dos clientes. Em seguida o cliente recebe do servidor o valor de uma PORT. E a partir deste momento é chamado o menu de interação do cliente. Ao fim de sua interação, o cliente passa um código no formato V+1+1 ou G+5, indicando ser gerente ou vendedor através da Letra inicial e números indicando as opções escolhidas, separados pelo sinal de adição. Caso haja erro na comunicação com o servidor, ele exibe em tela o erro e inicia o processo de eleição

A eleição:

Para realizar a eleição optamos por uma variação do algoritmo de anel. Nosso algoritmo não estabelece as comunicações entre os clientes transmitindo o seu ID. Isso não se fez necessário porque o servidor inseriu cada cliente conectado dentro de um array contido no arquivo JSON. Quando o cliente inicia o processo de eleição, ele faz uma varredura por esse array e atribui o título de eleito ao último cliente conectado. Nosso código utiliza a posição do cliente como um ID, de forma que sempre o valor mais alto ganhe a eleição

O processo de eleição está sendo executado pela função eleicao(HOST,PORT). O cliente ao solicitar uma funcionalidade do servidor e perceber que ele não está respondendo, captura essa exceção e executa a função eleicao, passando como parâmetro seu host e port. Inicialmente ele cria um servidor de eleição com todos os padrões citados anteriormente. Em seguida faz a leitura dos clientes conectados e as portas atribuídas a eles armazenados em clients.json. Logo após é capturado o cliente conectado na última posição (“ID” mais alto) como cliente eleito. Depois é feita uma comparação entre as portas do cliente eleito e do cliente que iniciou a eleição, se os valores forem iguais, o cliente assume a posição de servidor temporário, caso contrário ele fica preso aguardando que o novo servidor o libere. Todos os clientes conectados devem iniciar uma eleição.

O servidor temporário:

O servidor temporário é iniciado e logo em seguida envia uma mensagem para todos os clientes conectados fazendo assim a liberação dos que estavam presos na etapa de eleição. Em seguida ele assume as mesmas funções de comunicação exercidas pelo servidor principal