Universidade Federal de Viçosa – Campus Florestal
Bacharelado em Ciência da Computação

Trabalho Prático/PARTE3 – Sistemas Distribuídos – ENTREGA: 06/05/2025

Professora: Thais Regina de M. B. Silva
Semestre: 2025/01 – **Valor:** 15,0
Formato de entrega: PDF + código

O objetivo das duas partes finais deste trabalho é mostrar aos alunos as diferenças, benefícios, desvantagens e dificuldades do desenvolvimento de sistemas distribuídos utilizando diferentes visões de elementos arquitetônicos. Em outras palavras, será uma oportunidade de construir uma experiência sobre como é o desenvolvimento de SDs com e sem o uso de *middlewares*.

Para atingirmos este objetivo, cada grupo fará duas implementações do sistema proposto. A primeira (Parte 3) será feita utilizando simplesmente a API de Sockets, correspondendo assim à visão de processos para os elementos arquitetônicos. A segunda implementação (Parte 4) utilizará o recurso de um *middleware* RMI, através do qual os elementos arquitetônicos serão vistos como objetos.

Assim, nesta, que é a parte 3 do trabalho, toda a implementação deverá ser feita com uso apenas da API de Sockets. Cada elemento do sistema deverá ser implementado como um processo e a comunicação deverá ser feita por fluxo TCP. Observe que, ao usar a API de Sockets, a comunicação é feita entre processos, utilizando troca de mensagens, as quais devem ser inteiramente construídas e gerenciadas pelo programador. Em outras palavras, vocês serão responsáveis por definir a estrutura das mensagens a serem trocadas e deverão programar todo o processo de construção e leitura das mesmas.

Threads deverão ser utilizadas nesta implementação de forma bastante simples: o processo servidor deverá utilizar uma *thread* para recepcionar as requisições recebidas e outra(s) para processá-las. O uso de interface gráfica (simples, sem exageros) é obrigatório, e portanto também terá que lidar com threads para apresentar essa interface gráfica e realizar as comunicações. Para a implementação deverá ser utilizada obrigatoriamente a linguagem Python (e somente ela), para a interface gráfica também é obrigatório o uso do PyQt6 (não serão aceitos outros frameworks) e é também sugerido o uso do Docker para facilitar a execução dos servidores e simulação de diferentes acessos.

É importante destacar que todas as bibliotecas, dependências, instruções de instalação e comandos necessários para a execução do projeto devem estar claramente descritos na documentação que acompanha este trabalho. Não é objetivo deste projeto que a professora ou a mentora precisem pesquisar ou recorrer a tentativa e erro para conseguir executá-lo. A documentação deve ser completa, clara e objetiva, permitindo que qualquer pessoa com os requisitos básicos consiga reproduzir o funcionamento do sistema apenas seguindo o que está descrito.

Além disso, é importante que a entrega seja preparada para funcionar corretamente em um sistema Ubuntu 21.04, sem apresentar erros de execução, como bibliotecas faltando, versões incompatíveis, permissões incorretas ou quaisquer outros

problemas que comprometam a execução do projeto. Portanto, testar previamente o funcionamento completo em um ambiente compatível é parte essencial da entrega.

Requisitos

1) O SD será testado no sistema operacional Linux na distribuição Ubuntu 21.04. Sugiro fortemente o uso deste SO no desenvolvimento do trabalho, visto que não serão feitas adaptações nos códigos entregues para que os mesmos possam executar corretamente.
2) O SD deve possuir base de dados implementada via SQLite com arquivo, devendo o mesmo já conter tudo o que for necessário para que o sistema seja utilizado com sucesso.
3) Utilização de arquivos makefile ou Docker para facilitar a execução do SD.
4) Deve ser usado o Python 3, de preferência uma versão recente (3.10 ou superior).

Entrega

O que deve constar nesta entrega:

- Documentação concisa e completa, descrevendo o projeto do sistema distribuído desenvolvido com uso da API de Sockets.

  - Descreva todos os componentes desenvolvidos para o seu trabalho. Deixe bem claras todas as escolhas de implementação, tanto para parte do negócio, como para a comunicação entre os processos. É importante descrever a implementação, responsabilidades, funcionalidades e permissões dos servidores e do cliente definidos no sistema.

  - Descreva, com suficientes detalhes, todas as mensagens criadas e como elas são trocadas entre os processos

  - Descreva como se deu o uso de threads no SD (onde utilizaram, como foi feita essa implementação e se houve necessidade de algum mecanismo de sincronização)

  - Coloque uma conclusão em que vocês relatam as principais dificuldades percebidas com a programação via Sockets, indicando se e como as mesmas puderam ser superadas ou contornadas

  - A qualidade e o capricho com a documentação também serão considerados na avaliação.

- Arquivo README contendo o passo a passo detalhado para a correta execução do SD desenvolvido. Esta informação também poderá estar no texto da documentação.
- Todo código fonte produzido.

Cada dupla submeterá, via PVANET Moodle, um único arquivo compactado (.zip ou .tar.gz) contendo tudo o que foi pedido (documentação em formato PDF e código-fonte do sistema). O nome do arquivo deve seguir o padrão: **<nomes-e-matriculas>-TP1P3-SD.<zip/tar.gz>**.

Exemplo: **alice1234-bob1235-carol1236-TP1P3-SD.zip**

Tutorial

Para auxiliar no desenvolvimento, a monitora preparou um tutorial com um sistema exemplo simples, mas que cumpre todos os requisitos mínimos esperados para este trabalho. Esse exemplo demonstra de forma clara a comunicação entre uma interface gráfica (conforme especificado no enunciado), um cliente e um servidor, utilizando sockets, threads e Docker. Embora seja um sistema básico, ele serve como referência funcional para a estrutura do projeto e para a integração dos componentes principais. Recomenda-se fortemente que os alunos consultem esse material antes ou durante o desenvolvimento. Em caso de dúvidas, procurem a monitora.

Link do tutorial: Tutorial TP3