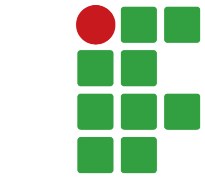

INSTITUTO
FEDERAL
Santa Catarina

Sistema Distribuído para Quebra de Senhas

Relatório Técnico Projeto Prático I

Marcelo Bittencourt do Nascimento Filho

Novembro, 2019

Histórico de revisões

Data	Versão	Descrição
23/10/2019	1.1	Definições do projeto
10/11/2019	1.0	Versão inicial
12/11/2019	1.2	Versão final

Sumário

1 Introdução

O presente relatório visa especificar e detalhar todos os processos realizados no desenvolvimento do Projeto Prático I que tem como objetivo a implementação de um sistema que fará a distribuição de tarefas para a quebra de senhas utilizando o *software John, The Ripper*. O sistema consistirá em um processo Mestre (também denominado como servidor ao longo do projeto) que terá a função de enviar arquivos e tarefas para processos Trabalhadores distintos (denominados também de processos clientes) que irão realizar a execução de atividades para a quebra de senhas. Essas tarefas estão relacionadas com a quantidade de caracteres a serem quebrados de acordo com as regras do *John*, sendo elas senhas com zero a cinco caracteres, seis caracteres ou sete caracteres.

É válido lembrar, que o Mestre poderá requerer e enviar algumas informações para seus clientes como por exemplo saber o estado dos mesmos (os possíveis estados dos clientes são de *espera* e *trabalhando*) e ordenar que as atividades sendo realizadas nos clientes sejam interrompidas. Um ponto importante para o bom funcionamento do sistema é permitir que o mesmo seja desenvolvido para uma comunicação bidirecional, ou seja, Mestre e Trabalhadores podem enviar e receber mensagens sem depender de respostas um do outro.

Na segunda seção deste documento serão detalhados os passos iniciais para a execução dos programas assim como as configurações necessárias para a comunicação entre os processos e por fim a explicação da estrutura do sistema.

2 | Tecnologias utilizadas

Diante dos problemas iniciais propostos pelo projeto, visando a comunicação bidirecional com independência de resposta por ambas as partes do sistema (Cliente e Servidor) foi definida como tecnologia principal de implementação o uso de *Sockets* para estabelecer esta conexão estável e confiável entre os processos. Porém, ao decorrer do desenvolvimento do projeto foi necessário realizar algumas técnicas para se conseguir o real objetivo, como por exemplo estabelecer pares de fluxos diferentes em cada processo onde um será responsável por receber mensagens e outro exclusivamente para enviar mensagens.

Tendo como princípio, a tecnologia *Socket* necessita de algumas informações para se conseguir realizar uma conexão. Primeiramente, os processos Trabalhadores necessitam conhecer o IP do seu Mestre assim como uma porta em que o mesmo está disponibilizando serviço. No projeto em questão, o usuário necessita saber o IP do Mestre e fornecer como argumento de linha de comando ao executar os arquivos dos Trabalhadores.

Realizando este processo, o cliente irá estabelecer uma conexão na porta 1234 (porta padrão para a primeira execução) e após isso, automaticamente os processos receberão do Mestre uma mensagem contendo outra porta para outra conexão, desta forma, todos os clientes irão ter duas conexões com servidor garantindo a bidirecionalidade. A figura 2.1 demonstra estes fluxos de conexão realizados entre os processos Mestre e Trabalhadores enfatizando a direção das mensagens na comunicação.

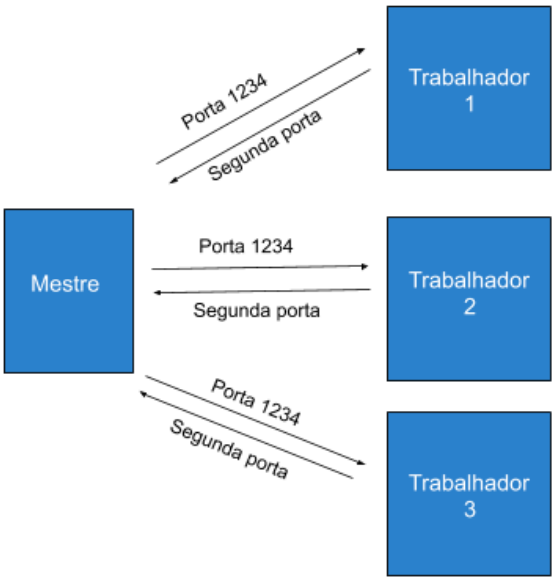

Figura 2.1: Representação das conexões estabelecidas pelos processos. *fonte: autoria própria.*

3 Estrutura do projeto

3.1 Projeto Mestre

O projeto relacionado ao processo Mestre é composto por quatro classes, sendo possível observar na figura 3.1 o diagrama UML referente à elas. A classe Mestre ficará responsável por realizar a conexão com o Trabalhador na porta 1234 e, ao passar por esta etapa o Mestre adiciona os fluxos da conexão em listas da classe ListaConexoes assim como dispara *Threads* para a execução da interface do usuário e para realizar uma nova conexão na porta escolhida pelo sistema.

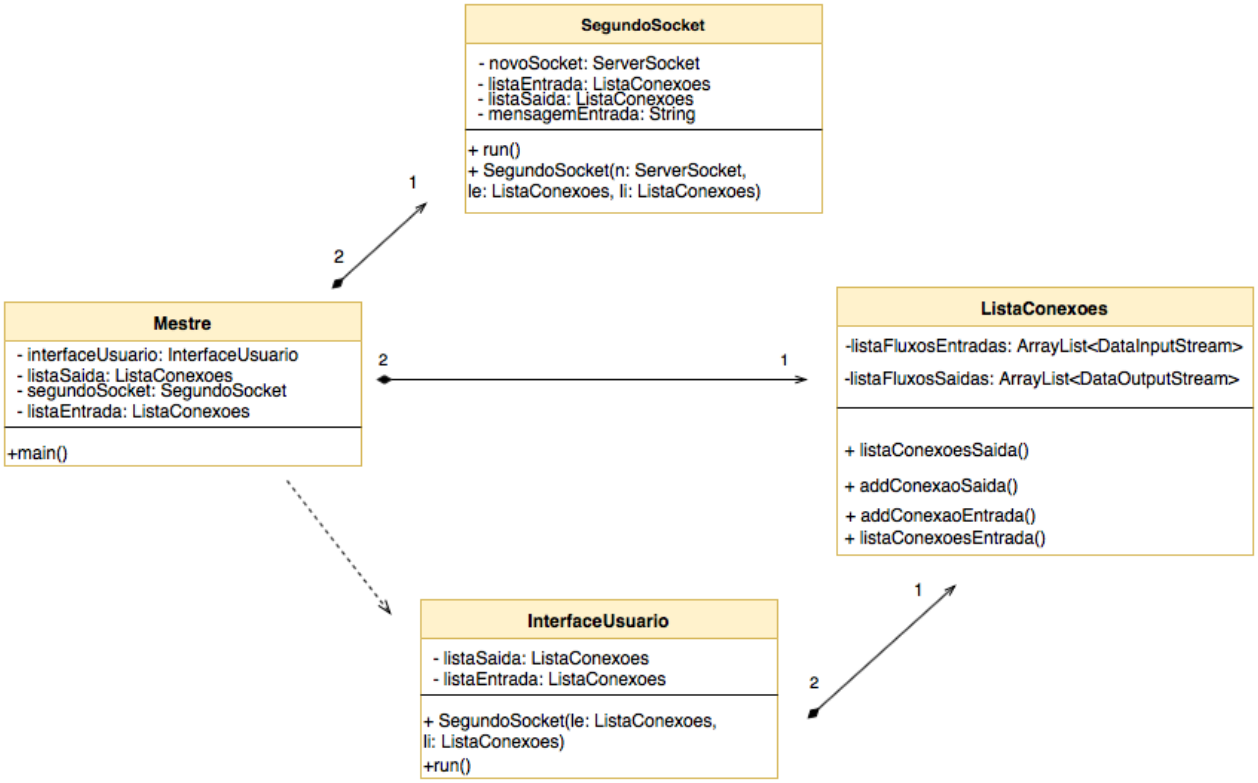

Figura 3.1: Diagrama UML projeto Mestre. *fonte: autoria própria.*

A classe InterfaceUsuario será responsável por fornecer ao usuário todas as possíveis opções que o mesmo poderá realizar com os processos Trabalhadores. Todo o processo de navegação pela interface se encontra no arquivo Cenario.md localizado aqui.

3.2 Projeto Trabalhador

O projeto referente aos Trabalhadores ficou constituído de cinco classes, estas ilustradas no diagrama UML da figura 3.2. Nesta figura é possível visualizaar que a classe Trabalhador terá a função de estabelecer a primeira conexão com o Mestre e logo após dispara uma *Thread* para a classe PrimeiroSocket. Esta classe ficará responsável por receber as mensagens do Mestre e tratá-las para realizar as devidas funcões, assim como estabelecer uma nova conexão e disparando ela para uma *Thread* que executará a classe SegundoSocket, responsável por enviar mensagens ao Mestre.

A clase ListaFluxos possui a função de armazenar o fluxo de saída para comunicação com o Mestre para todos as outras classe terem essa informação. A classe processoJohn realiza os processos relacionados com o serviço de quebra de senhas, ou seja, é nela que ocorrerá o disparo para execução dos comandos do aplicativo *John, the ripper.*

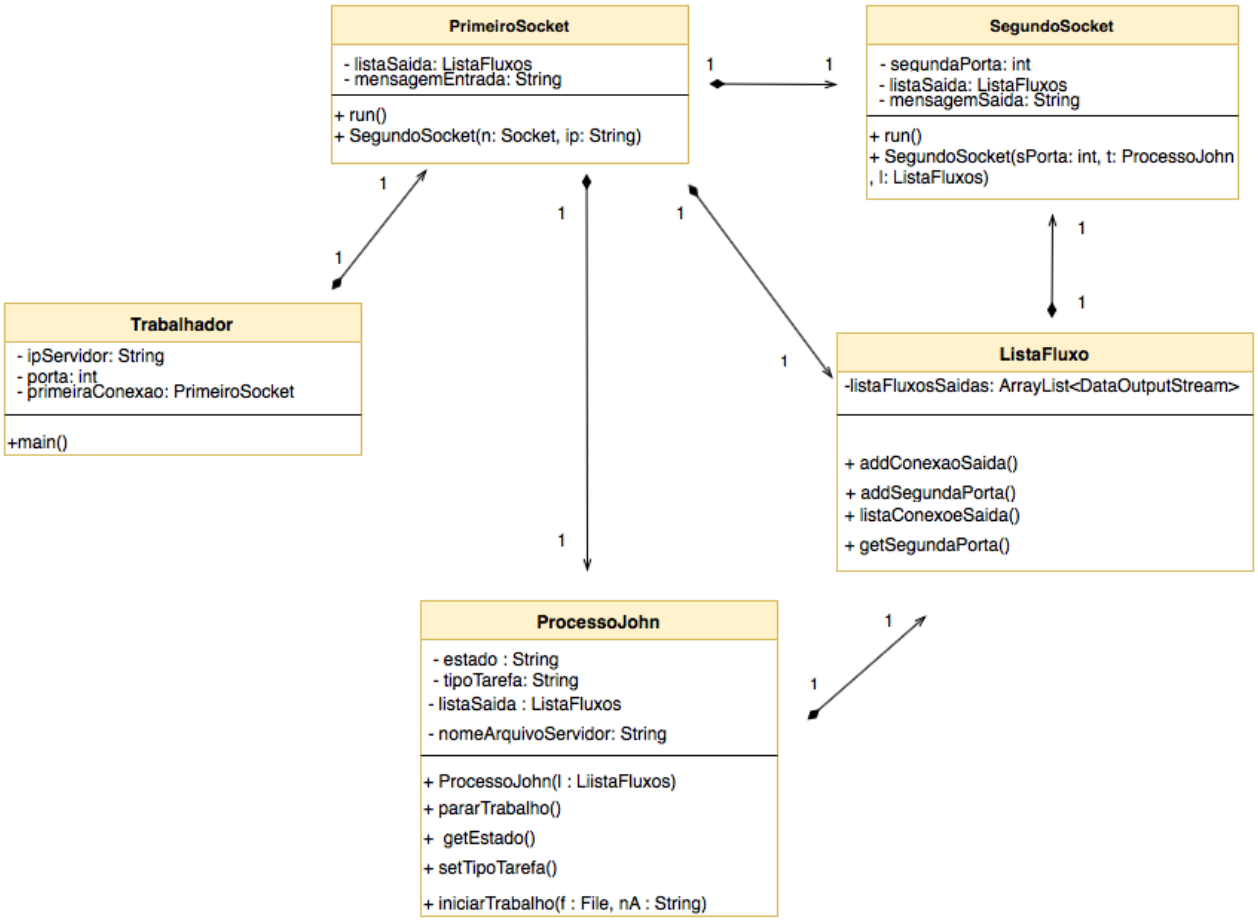

Figura 3.2: Diagrama UML projeto Trabalhador. *fonte: autoria própria.*