Projeto Computação Distribuída

Distributed CD Tester

Trabalho Realizado:

Ivan Horoshko 120603

Duarte Branco 119253

deti universidade de aveiro
departamento de electrónica,
telecomunicações e informática

Conteúdo

Arquitetura

A arquitetura usada neste projeto foi uma arquitetura P2P (peer-to-peer) distribuída, onde cada nó atua simultaneamente como distribuidor de tarefas e como executor dos módulos de teste. A comunicação entre nós foi feita através de endpoints RESTful de uma aplicação Flask, sendo os endpoints usados os seguintes:

/evaluation - Focado na iniciação de um pedido de avaliação e de consultar as mesmas;

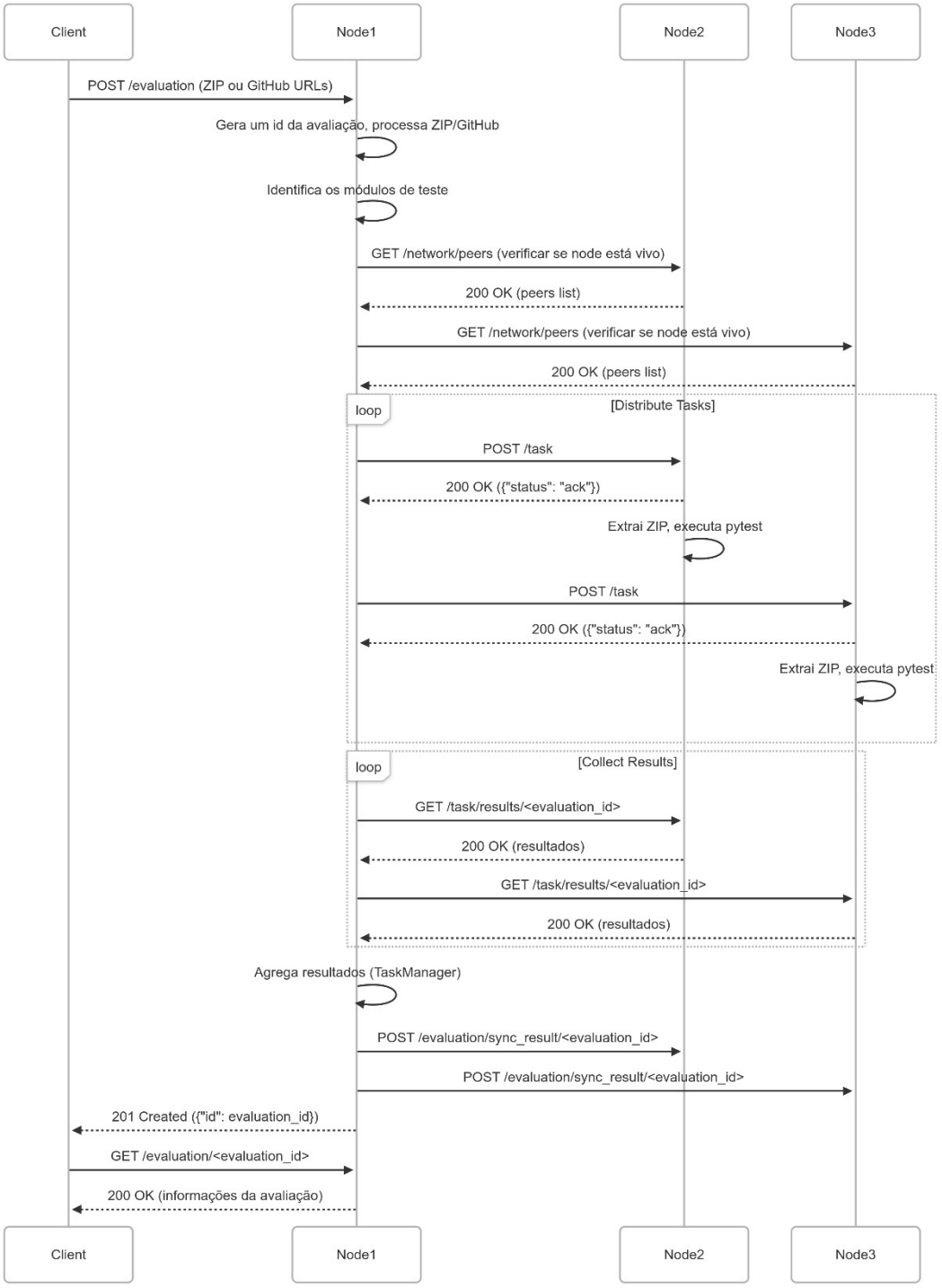
/worker – Focado na associação de uma tarefa a um node e para receber os resultados das tarefas mandadas;

/stats – Focado nas estatísticas do sistema distribuído;

/network – Focado na rede P2P como os nós conhecidos por cada nó e na descoberta de novos nós.

Troca de Mensagens durante um pedido de avaliação

Consideramos o seguinte exemplo onde temos no sistema distribuído com 3 nós:



O cliente inicia o pedido de avaliação através do endpoint *POST /evaluation* para qualquer nó do sistema, mandando um ficheiro ZIP dos projetos a avaliar ou através de link dos repositórios dos projetos.

O nó ao receber o pedido gera um id único e começa por processar o pedido. Para distribuir o nó tem de verificar se os nós da rede P2P estão vivos para distribuir os módulos entre eles, através do endpoint *GET /network/peers*. Contando com os nós que se encontram vivos começa a distribuição de tarefas através *POST /task*.

Cada nó com as tarefas executa a função *_process_stored_modules* por thread, que cria um ambiente virtual, instalando as dependências dos projetos e executa o pytest e no final do processamento guarda os resultados até serem pedidos.

O nó que fez a distribuição pega nos resultados de cada um através *GET /task/results/<evaluation-id>* e agrega os resultados para depois no final enviar o id da avaliação ao cliente. Depois da agregação, o nó inicial sincroniza com os nós disponíveis na rede através de *POST /evaluation/sync_result/<evaluation-id>,* pois o cliente pode perguntar a qualquer nó sobre as informações da avaliação.

Por sua vez o cliente usando o endpoint *GET /evaluation/<id>* consegue consultar as informações referente a avaliação em questão.

Distribuição de tarefas

No nosso sistema cada nó da rede P2P é capaz de ser um worker e de fazer atribuição das tarefas para os outros nós da rede. A distribuição dos módulos por nós de uma avaliação é feita através do seguinte cálculo:

$$Nº\ máx.\ de\ módulos\ por\ nó = \left\lceil \frac{Nº\ de\ módulos}{Nº\ de\ nodes} \right\rceil$$

Considerando um exemplo em que uma avaliação tem 5 módulos e no sistema só temos 2 nós, o número máximo de módulos por nó vai ser igual a 3, ficando um nó com 3 módulos e o outro com 2 módulos.

O nó que faz a atribuição das tarefas, mantêm uma lista dos módulos que foram testados e executa a distribuição até não haver mais módulos ainda por avaliar, em cada ciclo de distribuição o nó verifica os módulos por testar e os nós vivos na rede P2P, faz um novo cálculo e faz a distribuição dos módulos.

A lista de módulos é sempre atualizada sempre que receber uma resposta positiva do endpoint *GET /task/results/<evaluation-id>* caso contrário, os módulos que foram enviados ao nó em questão não são guardados na lista.