FundStarter

Sistemas Distribuídos 2015/2016

Entrega: 1 de novembro de 2015, 20:00

1 Objetivos do projeto

No final do projeto, o aluno deverá ter:

- programado um sistema de *crowfunding*, seguindo uma arquitetura cliente-servidor;
- usado sockets TCP/IP para comunicação com o servidor, sendo que este será multi-threaded;
- criado uma segunda camada de comunicação usando a API de alto nível JavaRMI;
- garantido a disponibilidade do serviço, através de uma solução de fail-over usado comunicação UDP.

2 Overview

Crowdsourcing tem-se tornado num mecanismo popular de tomada de decisões online. Tem também servido como mecanismo financeiro para apoiar novas ideias, projetos e produtos. Sob o nome de crowdfunding, potenciais clientes podem pagar adiantado por produtos e ter acesso a itens promocionais exclusivos enquanto as empresas podem receber dinheiro mais cedo, o que lhes permite avançar com projetos de investimento alto com garantia de terem clientes. Crowdfunding pode também ser usado para projetos sociais, que não pretendem fazer lucro, mas sim ajudar a desenvolver aldeias, cidades, empresas familiares e individuais em países em vias de desenvolvimento.

O crowdfunding tem sido um mecanismo unidirecional, em que as decisões são tomadas pelos autores do projeto, e os apoiantes concordam ou não em financiar o projeto. Este projeto da disciplina de Sistemas Distribuídos consiste numa plataforma de crowdfunding em que os apoiantes podem votar em diferentes alternativas de realização de um projeto consoante o dinheiro que dão ao projeto.

3 Exemplo

A empresa FitJump está a desenhar uma corda de saltar que conta o número de saltos por minuto. Eles precisam de 10 000€ para produzir as primeiras 200 unidades. Eles abrem um projeto no FundStarter com as seguintes recompensas:

- 5€ - O nome do apoiante aparece na lista de apoiantes do projeto na caixa;
- 60€ - O apoiante recebe uma unidade de FitJumpingRope;
- 100€ - O apoiante recebe duas unidades de FitJumpingRope.

Quando um apoiante doa dinheiro para um projeto, ele pode votar em opções que o projeto lhe forneça. Por exemplo, a FitJumpingRope pode ser produzida tanto em azul como vermelho. Os apoiantes podem votar em qual das cores preferem, e apenas a cor com mais votos será produzida.

Finalmente, podem haver recompensas extra por total acumulado. Por exemplo, se a FitJump chegar aos 20.000€ ambas as cores poderão ser produzidas. E se chegar a 50.000€ a FitJumpingRope poderá incluir um módulo bluetooth para transmitir a informação ao telemóvel.

4 Requisitos funcionais

A aplicação deverá permitir ao utilizador fazer as seguintes acções:

- **Listar projetos actuais, com o seu progresso** - Deve ser possível de verificar para cada projeto se está perto do seu valor objectivo, ou não.
- **Listar projetos antigos** - Deve ser possível consultar uma lista de projetos cuja data-final já terminou, vendo se foram aceites ou não chegaram ao objectivo.
- **Consultar detalhes de um projeto** - Os detalhes incluem a data limite para investimento, os possíveis rewards, uma descrição do projeto e níveis extra.
- **Registar conta** - Deve ser possível criar uma nova conta para um novo utilizador. A cada conta, deverá ser atribuído um saldo inicial de 100€.
- **Login** - Deverá ser possível fazer login, passando a ter acesso às seguintes funcionalidades.
- **Consultar saldo** - Deverá ser possível a utilizadores autenticados consultar o crédito disponível na plataforma.
- **Consultar recompensas** - Deverá ser possível a utilizadores autenticados consultar as recompensas a que têm direito, de doações feitas a projetos em curso e terminados com sucesso.

- **Doar dinheiro ao projeto (Pledge)** - Os utilizadores poderão doar dinheiro correspondente a uma das recompensas, escolhendo uma das alternativas do projeto ao mesmo tempo. O dinheiro é retirado da conta do utilizador e fica retido pela plataforma. O voto na alternativa do projeto deverá ficar registado. (Exemplo: o utilizador João doa 60€ à FitJump, escolhe a opção vermelho e fica com a recompensa de uma unidade.)
- **Enviar mensagens para o projeto** - Os utilizadores registados poderão enviar mensagens para um projeto, de forma a esclarecer eventuais dúvidas.
- **Criar um projeto** - Cada utilizador pode começar um ou vários projetos. Um projeto tem um nome, descrição, data limite, valor pedido e recompensas (cada recompensa tem um valor e descrição).
- **Adicionar e Remover recompensas ao projeto** - O utilizador responsável pelo projeto poderá remover e adicionar as recompensas associadas ao projeto.
- **Cancelar projeto** - O utilizador responsável pelo projeto poderá cancelá-lo, impossibilitando outros utilizadores de o visualizar. O dinheiro retido das recompensas pela plataforma deverá ser devolvido aos apoiantes.
- **Responder a mensagens de apoiantes** - As mensagens enviadas pelos apoiantes merecem resposta, e cabe apenas ao responsável do projeto responder-lhes.
- **Fim do projeto** - Quando se chega à data-hora limite de um projeto, é verificado se o conjunto do dinheiro doado chega ao limite pedido. Se não chegar, o dinheiro é restituido aos apoiantes e o projeto automaticamente cancelado. Se o projeto tiver recolhido dinheiro suficiente, este é enviado para a conta do responsável pelo projeto. Os apoiantes devem também receber indicação das recompensas a que têm direito.

Para grupos de 3 elementos, as seguintes funcionalidades devem também ser implementadas:

- Os projetos podem ter vários administradores que podem alterar e cancelar o projeto, bem como responder a mensagens de utilizadores.
- As recompensas poderão ser atribuídas a um segundo utilizador, como oferta, em vez do utilizador que apoiou o projeto.
- Podem ser adicionados e removidos níveis de recompensas extra, caso o projeto ultrapasse certos montantes de financiamento.

5 Requisitos não-funcionais

- Os clientes não deverão notar qualquer problema de rede, exceto longas falhas de rede.
- Os dados devem ser persistentes, quer no cliente, quer no servidor.
- Mensages, recompensas e projetos não poderão ser perdidos, nem no cliente, nem no servidor.
- Apoiar um projeto deverá ser uma operação transacional. Se é retirado crédito da conta do apoiante, a recompensa deve ficar registada, bem como a voto para a decisão do projeto.
- A decisão de final de projeto deverá também ser transacional. Ao enviar dinheiro ao promotor do projeto, as rewards deverão ser entregues aos apoiantes e a decisão final tomada com base nos votos.
- Separação de Áreas: Comunicação, Lógica de Negócio e Dados deverão estar separados no código e o projeto deverá poder correr em diferentes máquinas.
- Qualquer interface é aceite. Não será recompensada qualquer interface gráfica, pelo que se sugere fortemente que os utilizadores invistam o seu tempo nas áreas de comunicação, protocolos, disponibilidade, robustez, fail-over e não a fazer uma interface bonita.
- As defesas serão feitas em diferentes máquinas.

6 Passos recomendados

6.1 Arquitetura

Em primeiro lugar, os grupos de alunos deverão focar-se em desenhar a arquitetura da solução. Deverão ter uma visão geral dos vários componentes (servidor, cliente, base de dados) e como é que eles comunicam entre si (TCP, UDP, RMI). Embora estes componentes possam correr na mesma máquina durante o desenvolvido, eles deverão poder ser colocados em diferentes máquinas. A arquitetura deverá ser validada com os docentes antes de avançarem para os próximos passos.

6.2 Estruturas de dados

Devem especificar como estão os dados organizados, tanto em memória como em disco. Devem espeificar classes que guardem Utilizadores, projetos, Recompensas, Votos, Mensagens. Deverão também escolher uma solução para persistência (ficheiros binários, ficheiros de texto, sqlite, mysql, postgres, oracle, mongodb, etc).

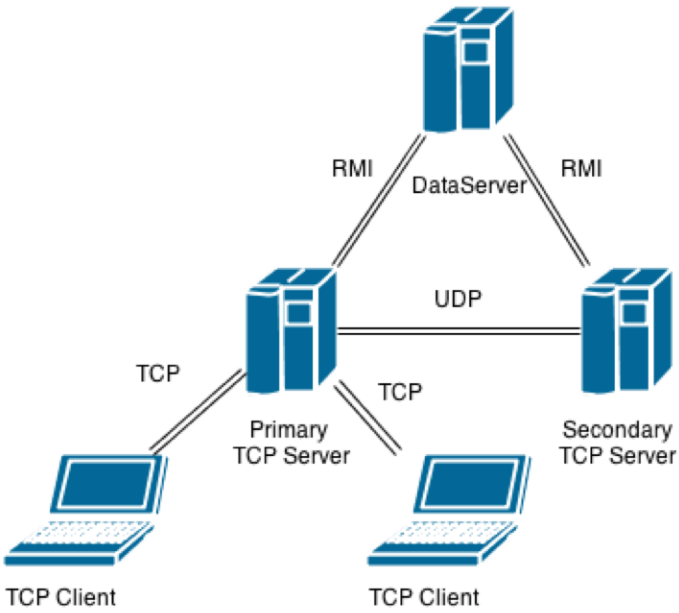

Fig. 1: Arquitetura do projeto

6.3 Cliente e servidor TCP

Devem criar duas aplicações. A primeira é um cliente TCP para os utilizadores utilizarem. Essa aplicação deve receber comandos do utilizador e enviá-los para o servidor e imprimir a resposta ao pedido. A outra aplicação é um servidor TCP que receberá os comandos, modificará os dados em memória e responderá ao cliente. O foco desta fase é a disponibilidade e fail-over da comunicação TCP, tratando todos os possíveis problemas da pilha de rede.

6.4 Cliente e servidor RMI

Devem criar um Servidor RMI que tratará de toda a comunicação com a base de dados escolhida. As operações estarão disponíveis através da rede. O Servidor TCP irá ser um cliente RMI e usará esse protocolo para fazer operações nos dados. A ideia é que a lógica de negócio e dados estejam separados da camada de comunicação. Será importante lidar com qualquer problema possível que aconteça com falhas de comunicação RMI da melhor maneira possível.

6.5 Failover sobre UDP

As máquinas também falham. Memória RAM corrompida, discos que falham, BIOS queimadas, etc... No vosso deployment, a mesma aplicação servidor TCP deverá estar a executar em duas máquinas em simultâneo. Uma delas será

a principal e a outra ficará em stand-by. Se a primária deixar de fornecer o serviço, a segundária passará a principal e trocam de papel. Se a máquina principal recuperar, deverá ficar como secundária.

6.6 Relatório

Devem alocar tempo para a escrita do relatório no final do projeto, tendo em conta os passos anteriores. Devem escrever o relatório de modo a que um novo colega se junte ao grupo e perceba a solução criada, as decisões técnicas efectuadas e possa introduzir novos componentes ou modificar os que já existem. O relatório deve incluir:

- Introdução
- Arquitetura da Solução
- Modelo de dados usado na aplicação.
- Protocolos usados para comunicação (Sobre TCP, RMI, UDP)
- Tratamento de erros em sockets e RMI
- Solução de Fail-over
- Implementação de Ordem Total
- Manual de instalação e configuração
- Descrição dos testes feitos à plataforma.

7 O que irão aprender

Este projeto presupõe que os alunos aprendam competências práticas relativas a:

- Programar sockets em Java.
- Tratamento de Excepções em Java.
- Desenvolvimento de Servidores multi-threaded.
- Java RMI
- Implementar uma solução de Fail-over
- Implementar um servidor com suporte a multi-protocolos.

8 Planos futuros para o projeto

No próximo projeto irão expandir esta solução, adicionando uma interface web usando HTML/JSP/Struts2 e irão integrar a aplicação com uma API REST de um serviço externo.

9 Entrega do projeto.

O projeto deverá ser entregue num ficheiro ZIP. Esse ficheiro deverá conter um ficheiro README com toda a informação necessária para instalar e executar o projeto sem a presença dos alunos. projetos sem este ficheiro, ou sem informações suficientes **não serão considerados**. projetos que não executem correctamente também não serão avaliados.

Dentro do ficheiro ZIP deverá também estar um PDF com o relatório. O relatório deve seguir a estrutura fornecida, dado que a avaliação irá incidir sobre cada um dos pontos.

Também no ficheiro ZIP deverá existirão exister três ficheiros JAR: client.jar, server.jar e dataserver.jar. O client.jar deverá ser executado para o cliente TCP, o server.jar deverá executar o servidor TCP e o dataserver.jar deverá executar o servidor RMI.

Finalmente, o ficheiro ZIP deverá ter também uma pasta com o código fonte do projeto.

O ficheiro ZIP deverá ser entregue na plataforma inforestudante até ao dia 1 de novembro de 2015, 20:00 (`http://inforestudante.uc.pt`).