



Universidade do Minho
Escola de Engenharia
Departamento de Informática

Mestrado integrado em Engenharia Informática
Paradigmas de Sistemas Distribuídos

Relatório do Trabalho Prático

23 janeiro 2018

Realizado por:

Carlos Silva – A75107
Catarina Cardoso – A75037
José Silva – A74601

Índice

Introdução.....	2
Decisões e Implementação	3
• Cliente.....	3
• Diretório.....	4
• Servidor Front-End.....	5
• Exchange.....	5
• Broker.....	5
• Esquema do Modelo	6
Conclusões	7

Introdução

Para este trabalho prático foi proposta a implementação de um modelo distribuído que fizesse o serviço de negociação de compra e venda de ações numa bolsa. Para isso, foi necessário incluir as diferentes entidades: cliente em Java, servidor front-end em Erlang, exchange em Java e diretório de ações em Java, que dispõe uma interface RESTful.

Foi também necessário incluir Protocol Buffers, para comunicações que envolvam o servidor front-end, ZeroMQ, para que os clientes recebam em tempo real notificações sobre o estado das suas ordens de venda/compra e informações sobre as suas subscrições, e Dropwizard, para a implementação da interface RESTful.

Decisões e Implementação

- **Cliente**

O cliente, ao iniciar-se, deve selecionar se pretende registar-se ou iniciar sessão (caso já se tenha registado previamente). O serviço de login, implementado no front-end, requer o que seja fornecido um username e uma password, concedidas pelo cliente. Após ter feito a autenticação no sistema, é criada uma socket ZMQ.SUB, que irá receber as notificações que subscrever. Um cliente autenticado tem várias opções:


1. **Logout:** O cliente pode desligar a sua sessão.
2. **Realizar ordem de venda:** Para o cliente registar este pedido deve indicar o nome da empresa, a quantidade de ações e o preço mínimo (exemplo de utilização: "buy sonae 1 1.5"). Depois é enviado um pedido GET ao diretório, que pesquisa pelo nome da empresa, e responde com o identificador da exchange. Com este identificador é realizado outro pedido GET ao diretório, para obter o host e a porta da exchange onde da exchange. Estes dados são remetidos para o servidor, que os tratará.
3. **Realizar ordem de compra:** Para o cliente registar este pedido deve indicar o nome da empresa, a quantidade de ações e o preço máximo (exemplo de utilização: "sell auchan 2 1.2"). Seguem-se as mesmas instruções que quando realizada uma ordem de venda.
4. **Subscrever até dez empresas:** O cliente pode subscrever várias empresas para receber notificações sobre estas (exemplo de utilização: "subscribe sonae auchan").
5. **Cancelar subscrição:** Para deixar de receber notificações sobre alguma empresa o cliente o cliente deve cancelar a subscrição. (exemplo de utilização: "unsubscribe sonae auchan").
6. **Listar empresas:** A qualquer momento o cliente pode comunicar com o diretório (a partir de um pedido GET) de modo a obter a listagem das empresas que são negociadas nas diferentes exchanges. (exemplo de utilização: "list"). Serão então imprimidas as informações sobre as empresas: nome e preços de abertura, fecho, mínimo, máximo do dia atual e do anterior, quando existam.

As ações 2. e 3. fazem uma subscrição ao broker com o formato "nome da empresa + username". Desta forma, receberão uma notificação no formato "nome da empresa + quantidade trocada + preço aplicado".

Para as ações 4. e 5. é feita um (ou vários) subscribe/unsubscribe "nome da empresa", respetivamente.

- **Diretório**

O cliente e a exchange dirigem-se ao diretório quando pretendem obter ou modificar, respetivamente, informações sobre exchanges ou companies. O diretório disponibiliza a seguinte interface RESTful, utilizando a framework Dropwizard:

company Operations about companies 

GET

/companies

Lists all companies

POST

/companies

Creates a new company

GET

/company/{name}

Lists info about given company

PUT


/company/{name}

Updates info about given company

DELETE

/company/{name}

Deletes a company

exchange Operations about exchanges 

GET

/exchanges

Lists all exchanges

POST

/exchanges

Inserts new exchange

GET

/exchange/{id}

Lists info about given exchange

PUT

/exchange/{id}

Updates info about given exchange

Para realizar os pedidos POST e PUT é necessário fornecer um JSON com a informação do objeto a adicionar (company ou exchange).

- **Servidor Front-End**

As comunicações que envolvam o servidor front-end são feitas através de Protocol Buffers, devido à interação Java-Erlang. O servidor suporta o sistema de autenticação de utilizadores e é a ponte entre os pedidos de compra e venda entre os clientes e as exchanges.

Quando um cliente se liga ao front-end é criado um novo actor, que trata os pedidos deste. Como já referido anteriormente, o cliente deve registar-se ou autenticar-se antes de poder realizar os outros pedidos. O servidor fica à espera de um pedido do cliente (que vem encoded: Protocol Buffers), trata esse pedido e envia uma mensagem encoded à exchange que tenha o host e a porta indicada pelo cliente. A exchange responde ao servidor, para este reencaminhar para o cliente, confirmando que é adicionada uma ordem de compra ou venda, mas também quando se encontra fora das horas de funcionamento.

- **Exchange**

Uma exchange é uma entidade que funciona entre as 9h00 e as 17h00. No início do dia é aberto um socket para receber pedidos do servidor (provenientes dos clientes). No fim do dia este socket é fechado, de modo a não receber mais pedidos, e são enviados para o diretório pedidos HTTP, para que sejam atualizados os valores alterados durante o decorrer do dia.

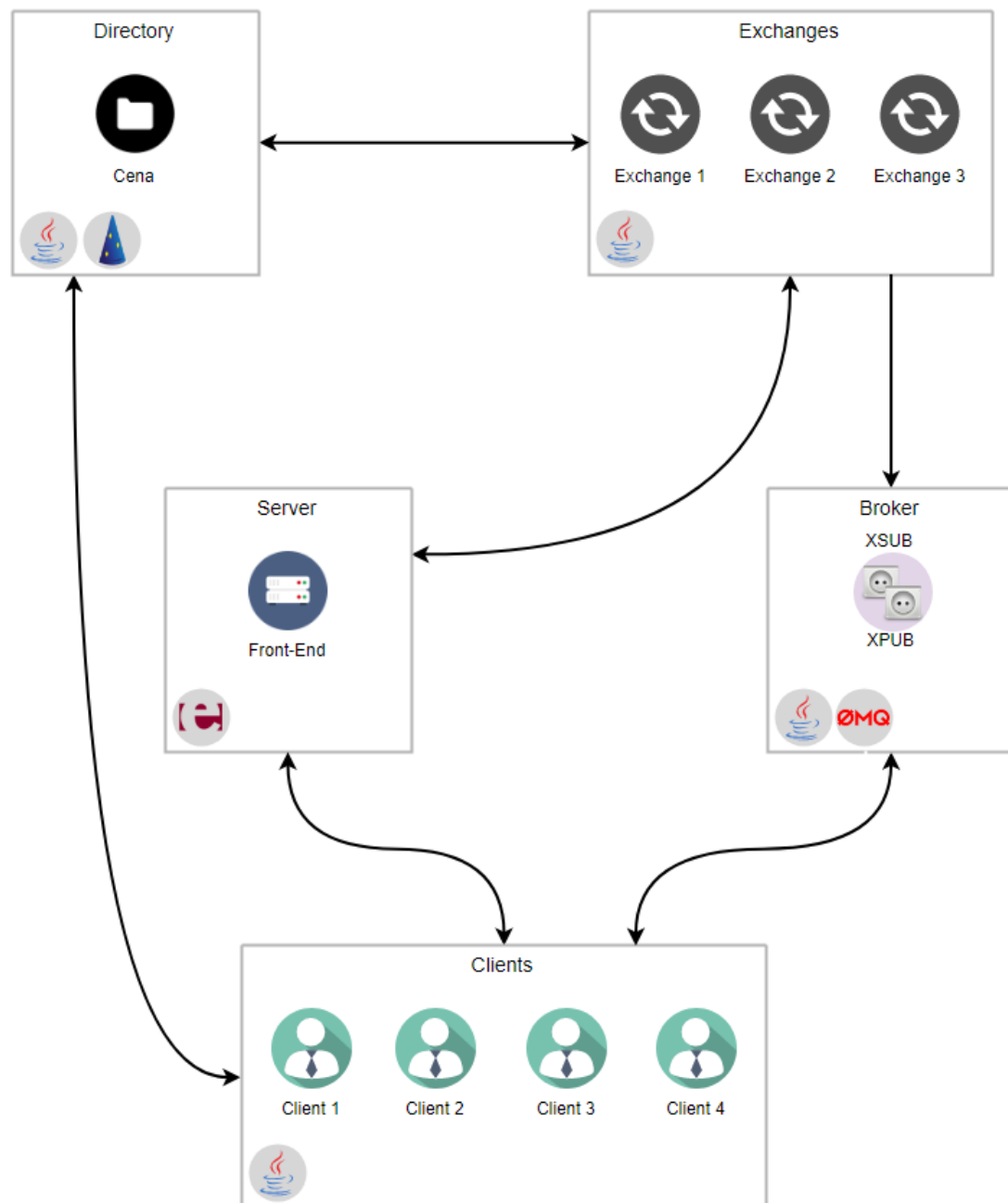
Nas classes Exchange e Share está a parte negocial do sistema. É aqui que são armazenados os mapeamentos dos pedidos de compra e venda, que são emparelhadas as correspondências entre pedidos (as transações) e calculados os preços e as quantidades de cada transação.

Quando é realizada uma transação, é enviada uma mensagem para o broker através de um socket ZMQ.PUB. Esta mensagem será entregue aos clientes envolvidos na transação no formato "nome da empresa + username + quantidade trocada + preço aplicado". Deste modo, para além da notificação ser recebida pelos clientes envolvidos, também os clientes que subscreveram a notificações da mesma empresa receberão a notícia. Importante notar que apenas os clientes que transacionaram a quantidade especificada quando adicionaram o seu pedido receberão a notificação (caso não tenham subscrito a essa empresa).

- **Broker**

Foi implementada a versão mais simples de broker, onde se utiliza um socket XSUB e outro XPUB, que são interligados através do proxy já implementado no ZeroMQ.

- **Esquema do Modelo**



Conclusões

O primeiro passo no desenvolvimento deste sistema distribuído foi a implementação do servidor front-end, da camada de negócio da exchange e do diretório. Devido à organização dos diferentes servidores foi possível o desenvolvimento paralelo de alguns blocos.

Com a implementação do cliente foi necessário adicionar novos “receives explícitos” no servidor front-end e estabelecer as comunicações cliente-servidor, servidor-exchange, exchange-servidor e servidor-cliente, usando Protocol Buffers. De seguida seguiu-se a ligação do diretório com os clientes e as exchanges. Por fim desenvolveu-se o broker usando ZeroMQ e foram tratados os pedidos de subscrição e cancelamento desta.

Como trabalho futuro seguir-se-ia uma implementação mais escalável ao nível do broker, deixando de haver apenas um, que se torna num gargarlo. A nível das exchanges, também se poderia permitir que uma empresa fosse negociada em mais que uma. Embora a API REST permita a adição de novas exchanges e companies, o sistema não é dinâmico e existem ficheiros de configuração para inicializa-lo.