Tecnologias de Distribuição e Integração

.NET Remoting: Relatório de Projeto

Grupo de trabalho:
Hugo Cardoso - 201105625 - ei11154@fe.up.pt
Vasco Gomes - 201106906 - ei11161@fe.up.pt

Faculdade de Engenharia da Universidade do Porto
Mestrado Integrado em Engenharia Informática e Computação
Rua Roberto Frias, sn, 4200-465 Porto, Portugal
Responsável: António Miguel Pontes Pimenta Monteiro

18 de Abril de 2015

Conteúdo

1 Introdução

Pretende-se desenvolver um sistema (Diginote Exchange System) de compra/venda e cotação de valores digitais chamados *diginotes*, baseado em .NET Remoting. Apesar de alguma semelhança das *diginotes* com *bitcoins*, para simplificar os seus aspetos de segurança, o mesmo sistema de compra/venda centraliza a existência das *diginotes* e quem é o seu dono atual.

Cada *diginote* é representada por um objeto, tem um número de série único, e um valor (também para simplificar, sempre igual a 1). As que existem estão registadas no sistema (através do seu número de série), assim como o dono atual. Os utilizadores registados no sistema (nome, nickname e password) deverão ser os atuais donos e quaisquer outros que pretendam adquirir *diginotes*.

O sistema permite apenas a venda e compra de *diginotes*, verificando a sua existência, e tomando nota do novo dono quando há uma transação. Em cada instante há uma cotação (para iniciar deverá ser igual a 1.00). Quem pretende comprar ou vender *diginotes* à cotação atual emite uma ordem de compra ou venda respetivamente, fazendo o sistema o emparelhamento entre compradores e vendedores, de acordo com regras específicas enunciadas em operações.

2 Arquitetura

2.1 Módulos

De forma a seguir um conjunto de boas práticas relativas à arquitetura, a aplicação desenvolvida foi estruturada em três módulos, ou projetos, individuais. A saber: *Server*, módulo central responsável por todas as operações e registos das mesmas; *Client*, que representa um utilizador normal do sistema; *Shared*, um conjunto de elementos comuns a ambos os módulos previamente referidos.

Serão, de seguida, apresentados os módulos em maior detalhe:

- **Server**: Representa o objeto remoto *Singleton* acessível por todos os clientes. Responsável pelo armazenamento de *diginotes* e registo de clientes e respetivas transações. O módulo *Server* é o único que efetivamente executa ações no sistema, através de pedidos de variados clientes.

  De forma a que um cliente tenha apenas acesso a uma determinada série de métodos deste módulo, o *Server* implementa uma interface disponível no módulo *Shared* que contém o conjunto de métodos que cada cliente pode executar remotamente.

  Para persistir toda a informação, optou-se pela utilização de uma base de dados em SQLite com as tabelas *diginote*, *exchange* (ordem de compra ou venda), *history* (transação efetuada entre duas *exchanges*) e *user*.

- **Client**: Representa um utilizador normal do sistema, e que comunica com o módulo *Server*. O *Client* disponibiliza uma interface gráfica para que um utilizador possa aceder remotamente aos métodos do *Server*.

  O *Client* subscreve os eventos de um objeto intermédio, através do módulo *Shared*, que, por sua vez, subscreve os eventos do objeto remoto. Deste modo, o *Server* conhece apenas a *metadata* do objeto remoto, e não a do cliente.

- **Shared**: Inclui todos os elementos comuns a ambos os módulos anteriores, evitando, assim, implementações e definições de estruturas iguais em módulos diferentes. Para além dos elementos referidos anteriormente, o módulo *Shared* inclui, também, as definições de todos os tipos de dados trocados entre *Client* e *Server*.

2.2 Estruturação de Classes

Cada um dos módulos criados é, por sua vez, estruturado em várias classes. Tentou-se, para *Client* e *Server*, separar toda a lógica da sua representação gráfica, dividindo-se, assim, os objetos de comunicação e os objetos gráficos em classes diferentes.

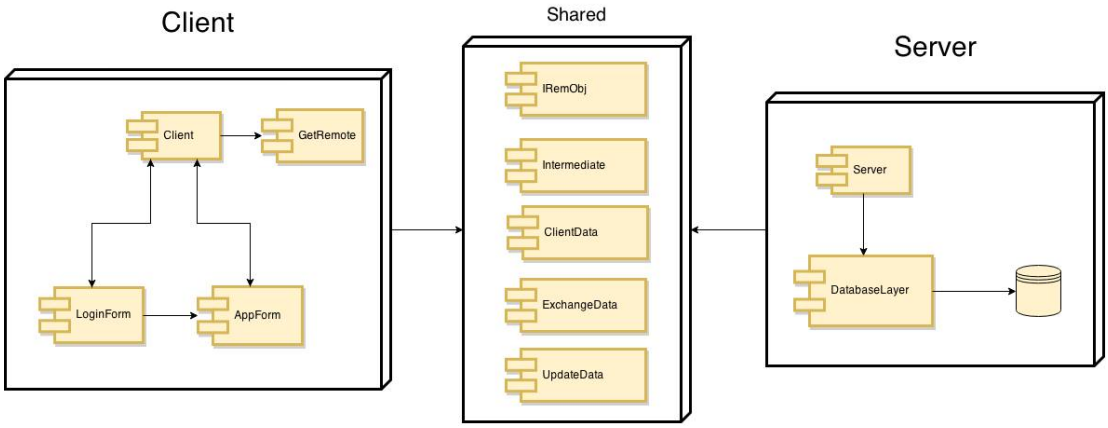


Figura 1: Diagrama de classes da aplicação desenvolvida.

Módulo Server

- *Server*: Objeto remoto a que todos os clientes acedem. Responsável por todo a lógica do módulo *Server*.
- *DatabaseLayer*: Responsável por todos os acessos diretos à base de dados, adicionando um maior nível de abstração ao *Server*.

Módulo Client

- *Client*: Responsável pela ligação ao objeto remoto e persistência de toda a informação de um cliente.
- *LoginForm*: Representa o ecrã inicial mostrado ao abrir a aplicação. Permite, também, o registo de novos clientes.
- *AppForm*: Ecrã principal da aplicação, com o qual um utilizador interage e realiza ações (através da classe *Client* que, por sua vez, comunica com o objeto remoto).
- *QuotationCreator*: Ecrã disponibilizado ao utilizador aquando de um pedido do servidor para a definição de uma nova cotação.
- *QuotationHandler*: Ecrã disponibilizado ao utilizador quando o servidor despoleta o evento correspondente à definição de uma nova cotação, para que o utilizador possa decidir o que fazer com as suas *exchanges* ainda não finalizadas.

Módulo Shared

- *IRemObj*: Interface implementada pelo objeto remoto, de forma a omitir a implementação de todos os métodos do mesmo a um cliente.
- *Intermediate*: Objeto intermédio criado por um cliente para subscrição dos eventos do servidor.

- *ClientData*: Classe que representa toda a informação relativa a um cliente, incluindo o seu saldo atual e todas as transações realizadas até ao momento. Utilizado para que o servidor possa transmitir a um cliente toda a sua informação atualizada após o início de sessão e sempre que é efetuada uma troca.
- *ExchangeData*: Classe que representa uma ordem de compra ou venda, incluindo o número de *diginotes* já compradas ou vendidas.
- *ExchangeType*: Enumerador que representa o tipo de uma *exchange*: compra ou venda.
- *UpdateData*: Classe composta por um objeto *ClientData* e um objeto *ExchangeData*, de forma a simplificar o envio de informação em alguns pedidos.

3 Desenvolvimento

3.1 Funcionalidades Implementadas

Todas as funcionalidades propostas no enunciado, e algumas extras, foram implementadas. De referir:

- Login/logout/registo no sistema – Os utilizadores deverão identificar-se perante o sistema (ou registar-se) antes de poderem efetuar qualquer outra operação.
- Cotação – Cada utilizador, quando logged in e através da sua aplicação quando ligada, deve saber sempre a cotação atual em tempo real.
- Ordem de venda – os detentores de *diginotes* podem emitir uma oferta de venda especificando a quantidade a vender. Se houver nesse momento ofertas de compra para essa quantidade, ou apenas para uma parte, a operação é concretizada à cotação atual na quantidade possível. No caso das ofertas de compra não serem suficientes o vendedor deverá indicar o valor de venda, para a quantidade não vendida, que só poderá ser igual ou menor que a cotação atual e passando a ser a nova cotação.
- Ordem de compra – os utilizadores que pretendem comprar *diginotes* podem emitir uma ordem de compra indicando a quantidade a comprar. Se houver oferta total ou parcial a correspondente transação é efetuada à cotação atual. Caso a oferta não seja suficiente o emitente da ordem terá de especificar um valor igual ou maior que a cotação atual, passando a ser a nova cotação.
- Em qualquer momento os emitentes de ordens de venda ou compra ainda não satisfeitas podem respetivamente descer ou subir o valor das suas ordens.
- Sempre que a cotação desce e haja outras ofertas de venda pendentes, estas ficam suspensas por um período de tempo (para testes vamos usar 1 minuto), podendo o emitente dessas ordens confirmar o novo valor ou retirar a ordem. Findo o período de suspensão e não havendo qualquer resposta do emitente esta mantém-se à nova cotação. Do mesmo modo, sempre que a cotação sobe e haja outras ofertas de compra não satisfeitas, estão são suspensas para confirmação ou retirada, presumindo-se confirmadas à nova cotação se não houver resposta no período de suspensão.
- Quando se procede a uma transação o sistema deverá sempre começar por satisfazer as ordens ativas mais antigas.
- Cada utilizador, quando logged in e através da sua aplicação quando ligada, tem acesso a um gráfico correspondente à cotação atual.

3.2 Modo de Funcionamento

De seguida são apresentadas algumas imagens que demonstram os principais fluxos de funcionamento da aplicação:

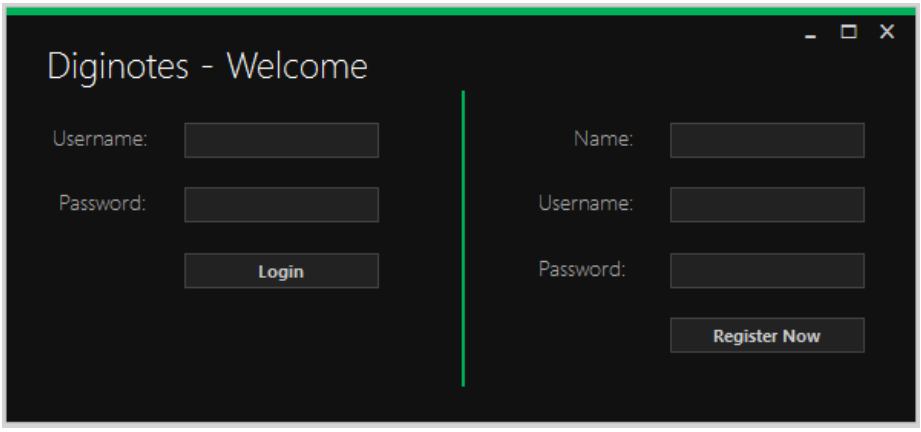


Figura 2: Ecrã inicialmente apresentado ao utilizador. Permite o login e registo de novos utilizadores.

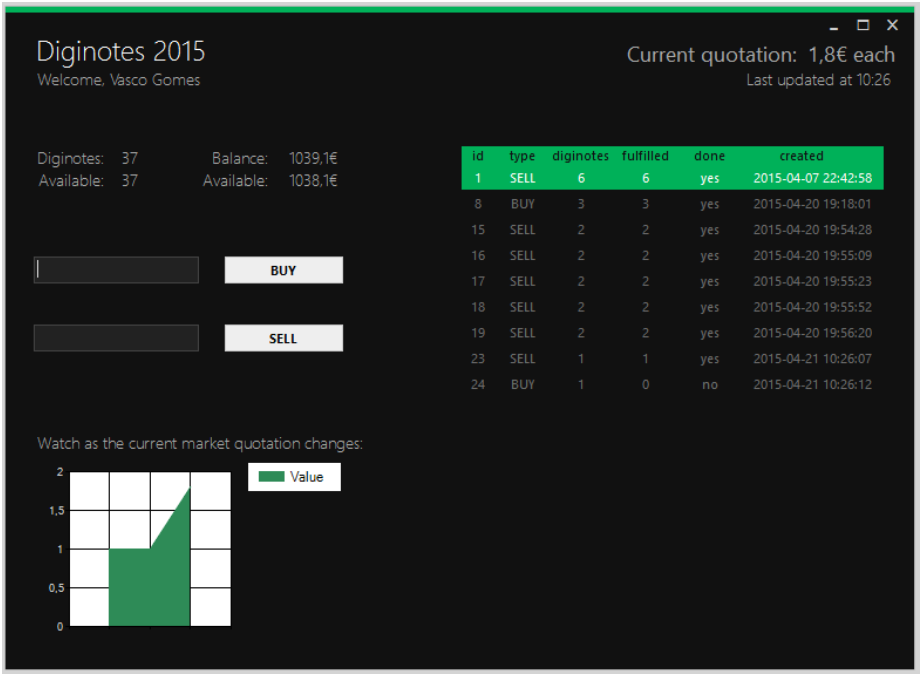
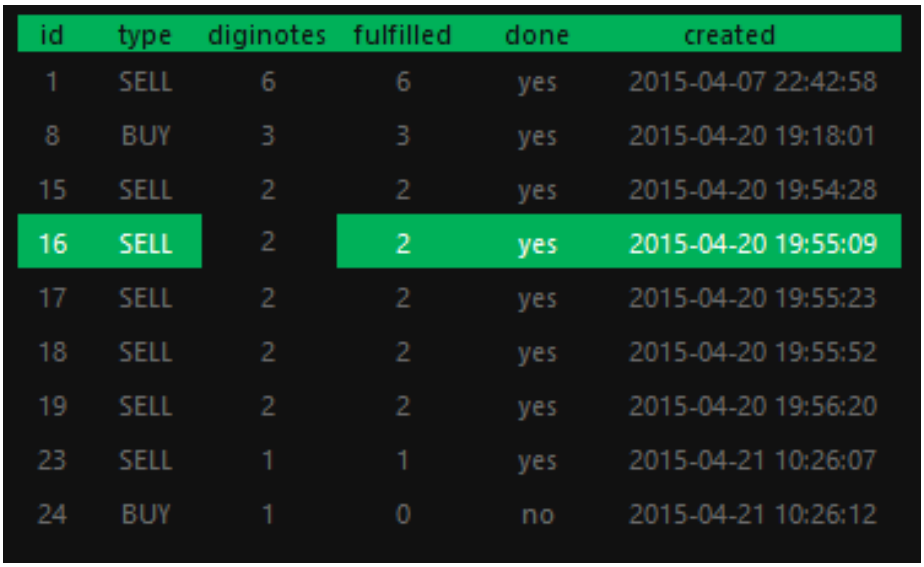


Figura 3: Ecrã principal da aplicação, permite ao utilizador executar todas as operações. É atualizado, em tempo real, de acordo com eventos acionados pelo objeto remoto.



id	type	diginotes	fulfilled	done	created
1	SELL	6	6	yes	2015-04-07 22:42:58
8	BUY	3	3	yes	2015-04-20 19:18:01
15	SELL	2	2	yes	2015-04-20 19:54:28
16	SELL	2	2	yes	2015-04-20 19:55:09
17	SELL	2	2	yes	2015-04-20 19:55:23
18	SELL	2	2	yes	2015-04-20 19:55:52
19	SELL	2	2	yes	2015-04-20 19:56:20
23	SELL	1	1	yes	2015-04-21 10:26:07
24	BUY	1	0	no	2015-04-21 10:26:12

Figura 4: Demonstração do modo de edição de uma ordem de compra ou venda. Após um duplo clique sobre a coluna de *diginotes*, a célula fica editável da forma acima mostrada.

Todos os elementos da interface são explicados, em detalhe, na secção de instruções disponibilizada abaixo.

4 Conclusão e Perspetivas Futuras

Analisando o trabalho desenvolvido, chega-se à conclusão que se cumpriram todos os objetivos proposto para este ponto de avaliação, sendo que todas as funcionalidades pedidas foram implementadas com sucesso.

Conclui-se que o conceito de *remoting*, apesar de interessante, se revelou, por vezes, um desafio. Julga-se que a maior dificuldade na realização do projeto passou pela falta que experiência com C#, a linguagem utilizada, que por vezes dificultou a implementação de eventos e respetiva sincronização. No entanto, no final do projeto, acredita-se que estas dificuldades tenham sido superadas.

Considerando o balanço entre planeamento, projeções, objetivos realizados e objetivos por alcançar, estima-se que 100% do trabalho a cumprir tenha sido concluído, e o projeto tenha sido terminado num ponto positivo.

5 Instruções para Utilização

Para correr a aplicação, o módulo *Server* (Server.exe) deve ser iniciado em primeiro lugar. De seguida, pode podem ser iniciadas múltiplas instâncias da aplicação *Client* (Client.exe).

Como referência e manual de instruções, são explicados todos os elementos da interface gráfica disponíveis a um utilizador:

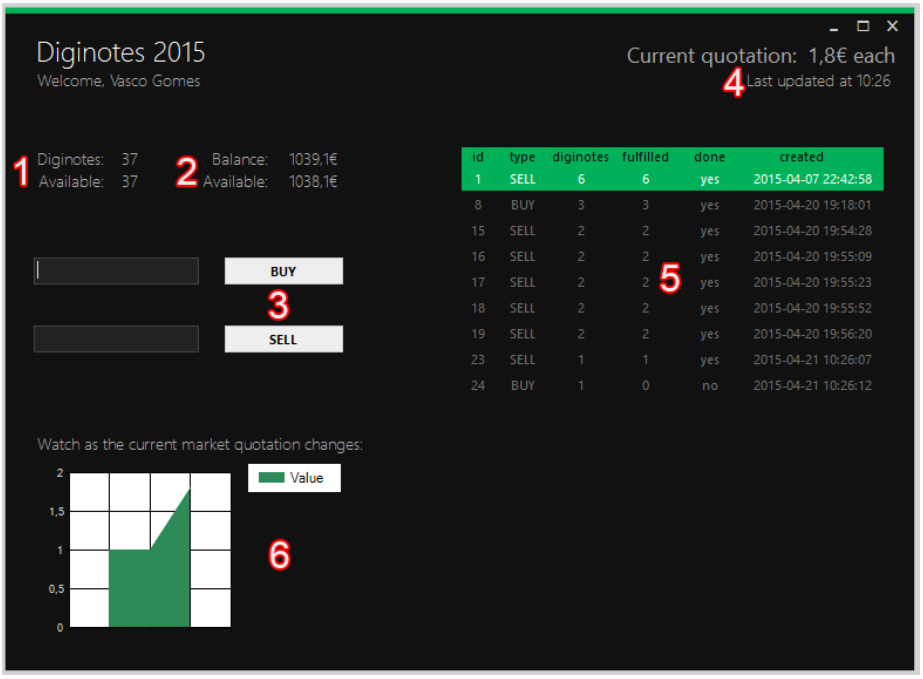


Figura 5: Principal ecrã da aplicação, permite ao utilizador executar todas as operações. É atualizado, em tempo real, de acordo com eventos acionados pelo objeto remoto.

- 1: *Diginotes* representam o número de *diginotes* que um utilizador possui atualmente. *Available* representa o número de *diginotes* que um utilizador pode vender. Por exemplo, após iniciar uma ordem de venda, o número de *diginotes* que um utilizador possui mantém até ao momento da transação de venda, mas o número de *diginotes* que este possui para vender diminui (estão "reservadas"para a ordem de venda efetuada).
- 2: *Balance* representa o saldo atual de um utilizador. *Available* funciona de modo semelhante ao mencionado acima, mas para ordens de compra.
- 3: Campos de emissão de ordens de compra ou vendo. Acionados através da tecla "Enter"ou pressionado o botão do respectivo campo.
- 4: Representa a cotação atual do sistema, em tempo real, e o último momento em que foi modificada.
- 5: Histórico de todas as ordens de compra ou venda previamente emitidas por um utilizador. Todas as ordens ainda não finalizadas (ou seja, em que

o número de *fulfilled* ainda não seja igual ao número de *diginotes*) podem ser editadas através de um clique duplo na coluna de *diginotes*.

- 6: Gráfico corresponde à cotação atual. É atualizado de minuto a minuto.

6 Referências

[1] Microsoft Developer Network: ".NET Remoting", 2015 [Online]. Disponível em https://msdn.microsoft.com/en-us/library/vstudio/72x4h507%28v=vs.100%29.aspx (Abril, 2015)

[2] António Miguel Pontes Pimenta Monteiro: "Distribution and Integration Technologies", 2015 [Online]. Disponível em http://paginas.fe.up.pt/ apm/T-DIN/ (Abril, 2015)