INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Área Departamental de Engenharia de Electrónica e Telecomunicações e de Computadores

Licenciatura/Mestrado de Engenharia Informática e Computadores

Unidade Curricular de Sistemas Distribuídos

(2º semestre letivo 2014/2015)

Serviço de Jogo

Grupo Nº 2

Autores: 36907 Carlos Serra

38205 Adriano Sousa

38222 Marta Nascimento

Junho de 2015

Resumo

Neste trabalho foi desenvolvido um jogo que consiste na procura de um tesouro, a ser jogado por múltiplos jogadores e representado numa matriz preenchida aleatoriamente. O jogo tem as seguintes regras:

* Cada jogador joga indicando uma coordenada da matriz;
* Um jogador pode ganhar, acumulando, mais vidas ao escolher uma posição marcada com vida; não está em curso;
* Um jogador pode desistir do jogo;
* Um jogador morre se escolher uma posição marcada com morte e já não tiver vidas, se tiver, perde uma vida;
* O jogo termina quando um jogador indicar a posição da matriz que contém o tesouro.

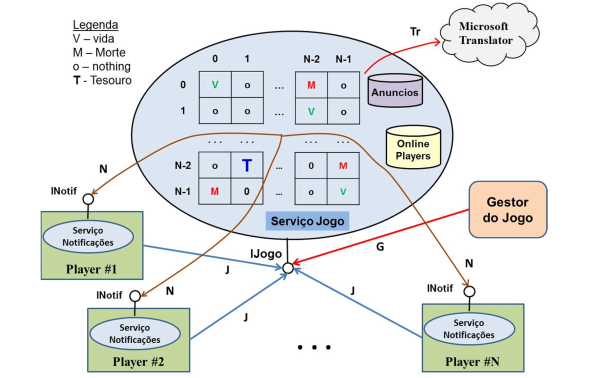
Introdução

Este trabalho serve para consolidar as temáticas aprendidas ao longo do semestre na unidade curricular de Sistemas Distribuídos. Como tal, foi-nos pedido para desenvolver um jogo usando arquiteturas orientadas aos serviços (SOA) com Windows Communication Foundation (WCF).

Requisitos

* Arquitectura orientada ao serviço com WCF;
* Existe um servidor central que aloja um serviço Jogo que suporta um jogo de procura de um tesouro, a ser jogado por múltiplos jogadores;
* Existe uma aplicação Player que, para além de ser usada pelos jogadores, aloja um serviço de notificações para que o serviço jogo possa notificar os jogadores de eventos ocorridos durante o jogo;
* Existe uma aplicação usada pelo gestor do jogo que permite iniciar o jogo e inserir anúncios publicitários que serão difundidos para os jogadores em jogo através do serviço de notificações;
* O serviço Jogo utiliza o serviço Microsoft Translator sempre que tiver que traduzir anúncios de acordo com o perfil dos diversos jogadores;
* Possibilitar as ações seguintes:
  + J: Registo de um jogador no serviço Jogo, indicando um nome que o permita identificar, a língua da sua preferência, bem outros detalhes que ache pertinentes;
  + Jogar numa posição;
  + Desistir do jogo;
  + N: Notificação de início e fim de jogo;
  + Notificação de anúncios;
  + Notificação de número de vidas ou morte;
  + Notificação que o jogo foi suspenso pelo Gestor;
  + G: Iniciar, terminar ou suspender um jogo;
  + Publicar anúncios indicando a língua respetiva.

Cenário



# Desenvolvimento

## Interfaces

O desafio inicial foi o desenvolvimento das duas Interfaces que o Serviço teria de implementar. A existência de duas interfaces é importante pois existem duas entidades que se podem ligar a este serviço, um Manager e um Cliente.

O Manager será a entidade que fará a gestão de toda a aplicação, fazendo as decisões de criar um jogo, recomeçar, parar ou até mesmo enviar anúncios publicitários. Um cliente será a entidade que poderá jogar o jogo, terá inicialmente de fazer um registo e após poderá fazer as jogadas.

As interfaces anteriormente faladas terão os métodos que cada uma das entidades poderá utilizar. IServiceCliente irá ter os métodos que o cliente poderá utilizar do serviço e IServiceManager os métodos do Gestor do serviço.

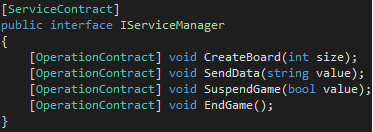


Figura 1: Declaração da interface de IServiceManager

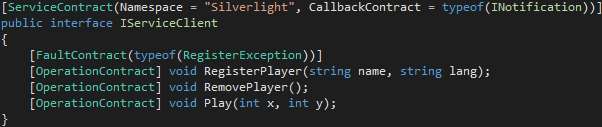


Figura 2: Declaração da interface IServiceClient

Gestor

Como se pode averiguar pela figura 1, um gestor poderá efectuar quatro tipos de operações. Criar um novo tabuleiro, que tanto pode ser utilizado quando este ainda não existe ou mesmo a meio de um jogo, que irá criar um novo jogo. Para isto é necessário que o gestor disponibilize um número que será utilizado para o tamanho do tabuleiro. Sobre o jogo é ainda possível suspender um jogo ou finaliza-lo. O método SendData é aquele que permitirá ao Gestor fazer a publicação de anúncios.

Cliente

Por outro lado o jogador terá menos opções (figura 2). Para começar este deve-se regista no jogo, sem esse registo nenhuma das outras opções funcionará. Para o registo de um jogador é possível especificar qual a linguagem que este pretende utilizar. Após registo o jogador pode retirar-se desse jogo ou fazer as jugadas que pretender, mas isto apenas enquanto tiver vidas, sendo automaticamente apagado da lista de jogadores.

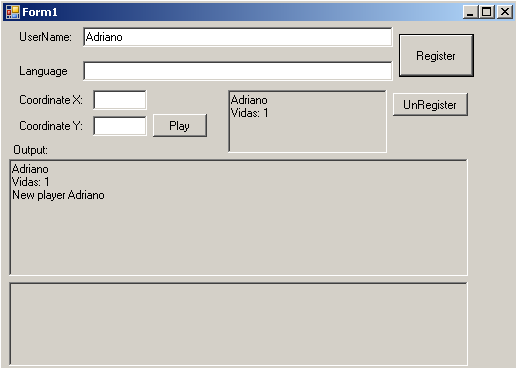


Figura 3: Interface gráfica apresentada ao jogador após registo.

Durante o registo do jogo pode ainda ser gerada uma excepção caso o gestor ainda não tenha dado ordens arranque do jogo. Sendo que esta excepção tem de ser passada do Serviço para o cliente, foi necessária a criação de um tipo (RegisterException) com contrato para que fosse possível passar enviar a informação (utilizando DataContract e DataMember).

Do lado do cliente foi ainda necessário apanhar a excepção como uma FaultException<RegisterException>, desta forma já será possível utilizar os dados passados pela excepção.

Com o sucesso do registo, irá então aparecer os dados do jogador numa das janelas. Sempre que o jogador fizer uma jogada que altere o estado desse jogador (ganhar ou perder vidas), esta janela será actualizada, de forma a apresentar ao jogador com actualizações constantes.

Para efectuar uma jogada basta inserir uma coordenada x e outra y. caso estes valores sejam inválidos toda a informação será disponível na caixa de texto central, informando se a jogada foi ou não válida e qual o resultado dessa mesma jogada.

Serviço

O serviço como referido anteriormente necessita implementar as interface IServiceManager e IService Client. É ainda composto por uma lista de jogadores activos no jogo e um tabuleiro.

Sempre que um jogador efectua uma jogada, é feita uma verificação se este está nessa lista. Esta verificação é útil para casos como quando um jogador anula o seu registo ou morre, sendo removido dessa lista.

Como existe a opção do gestor enviar anúncios, foi ainda necessária a criação de outra interface INotification. Esta por seu lado deve ser implementada pelo cliente, de forma a permitir comunicação do serviço para o cliente.

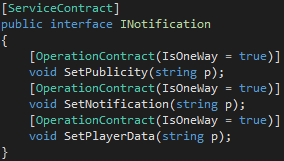


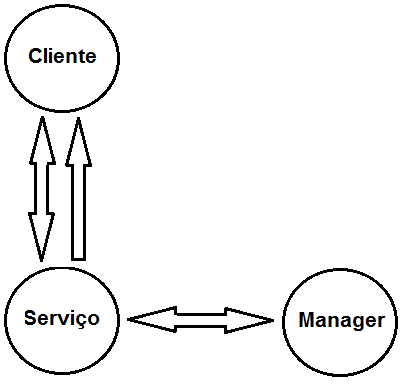
Figura 4: Interface INotification.

Foi ainda necessário alterar a interface que o cliente utilizará, de modo a informar o cliente que a utilizará que deverá implementar INotification, como se pode ver na Figura5.



Figura 5: Alteração na interface IServiceCliente para a utilização de INotification

Esta implementação implicou ainda a alteração do ficheiro de configuração do serviço. Este continha dois basicHttpBinding, um para cada interface. Mas para a utilização do INotification foi necessário alterar o binding do IServiceClient, onde passou a ser um wsDualHttpBinding.



# Conclusão

Durante a realização deste projeto, foram necessários bons conhecimentos em cadeiras anteriores, sem as quais o projeto se tornaria mais complicado de realizar, nomeadamente:

* RCp (Redes de Computadores) – protocolos de comunicação, nomeadamente HTTP, baseado em ligações TCP.
* PC (Programação Concorrente) – noção de *thread* e contexto de sincronização.

A cada fase, conseguiu-se superar todas as dificuldades que foram surgindo, o que requer uma boa análise do problema.

Atingiu-se o objetivo deste trabalho que servia para consolidar as temáticas aprendidas ao longo do semestre na unidade curricular de Sistemas Distribuídos. Nomeadamente, o desenvolvimento de um jogo usando arquiteturas orientadas aos serviços (SOA) com Windows Communication Foundation (WCF).

# Referências

* <https://msdn.microsoft.com/en-us/library/ms731082%28v=vs.110%29.aspx>
* <http://www.dotnetfoundation.org/blog/wcf-is-open-source>
* <https://pt.wikipedia.org/wiki/Service-oriented_architecture>
* http://www.service-architecture.com/articles/web-services/service-oriented\_architecture\_soa\_definition.html