1º Trabalho Prático de Avaliação

Objectivo: Desenvolvimento de sistemas distribuídos usando objectos distribuídos na plataforma .NET

Nota: O trabalho deve ser realizado até 21 de Maio de 2012, incluindo um relatório que descreva o trabalho com as opções tomadas ao longo da sua realização. (Enviar, os projectos em Zip file e relatório, para lass@isel.ipl.pt). Note que, como foi referido na apresentação da disciplina, a qualidade do relatório terá peso na avaliação do trabalho realizado. O relatório deverá permitir ao leitor entender, quais os requisitos funcionais e não funcionais, a arquitectura do sistema, as interacções entre as partes, bem como os aspectos relevantes da implementação dessa arquitectura realçando os pontos fortes e fracos da solução. Evite descrever código a menos que se justifique nalguma situação especial.

Considere um cenário de um sistema distribuído com os seguintes requisitos (ver Figura 1):

- Cada utilizador (*Peer*), através de uma aplicação gere informação que consiste numa colecção de artigos científicos {Titulo, Autores, Ano de publicação, Resumo};
- A colecção de artigos pode ser armazenada num ficheiro XML com a estrutura que ache mais adequada e fácil de carregar/guardar, por exemplo, através da classe *System.Xml.Serialization.XmlSerializer*);
- Um *Peer* pode ser promovido a *SuperPeer*, possibilitando o registo de outros *Peers*;
- Um *SuperPeer* pode ligar-se a outros *SuperPeers* criando, assim, uma rede global de múltiplos *Peers* que pode crescer ao longo do tempo;
- Um *Peer* conhece sempre um *SuperPeer*, no qual se regista e ao qual pode pedir informações sobre outros *Peers;*

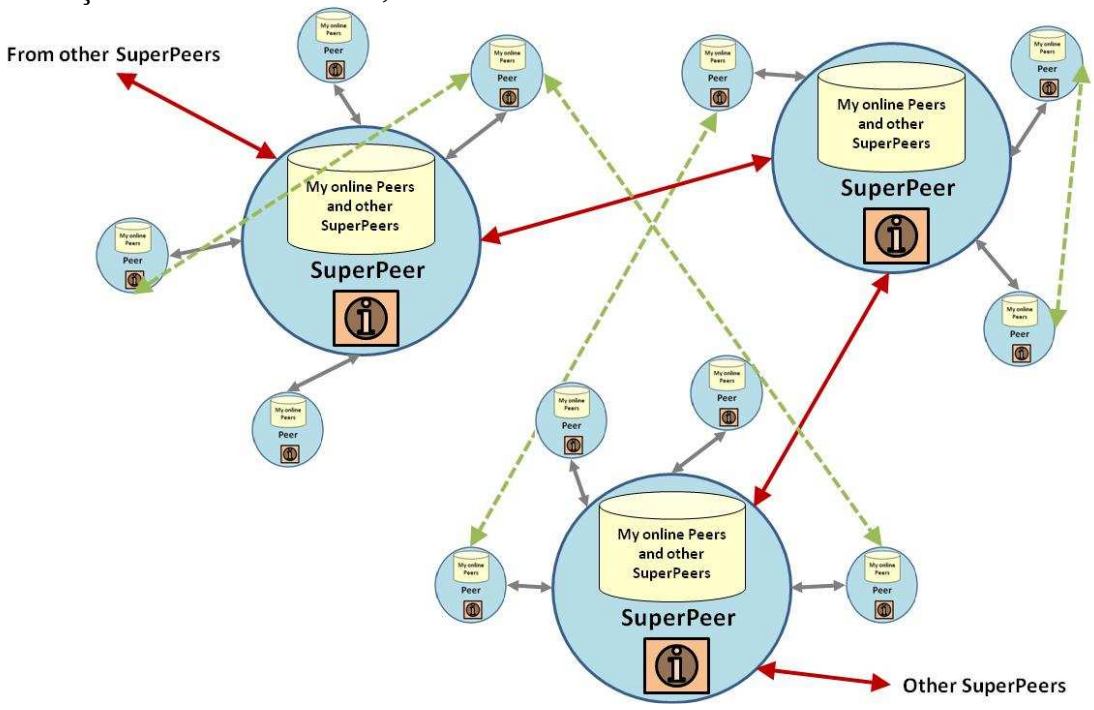

Figura 1 – Sistema distribuído de procura de artigos

- Quando o utilizador faz uma pesquisa por uma determinado artigo que não existe localmente, é enviado um pedido (getPeers) ao *SuperPeer* para que este lhe forneça potenciais *Peers* onde pode ser encontrado o artigo;
- Há medida que cada utilizador (*Peer*) vai conhecendo outros *Peers* vai criando um registo de *Peers online* de forma a aumentar a sua rede de contactos para pesquisas de artigos;
- Os *SuperPeers* reenviam sempre os pedidos (getPeers) recebidos para os outros *SuperPeers* que sejam conhecidos;
- Deve existir um mecanismo para evitar que um pedido (getPeers) continue a circular na rede global eternamente;
- A resposta a um pedido (getPeers) pode vir de qualquer *SuperPeer* ;
- À medida que um *Peer* vai coleccionando referências para outros *Peers*, pode contactá-los para fazer pesquisas de artigos;
- Devem ser tratadas situações de falha, principalmente:
  - ✓ Um determinado *Peer* pode desligar-se ou falhar independentemente dos outros *Peers* que mantêm eventualmente referências para o mesmo;
  - ✓ Um *SuperPeer* também pode falhar;
  - ✓ Dado que um pedido (getPeers) pode demorar tempo até atingir um *SuperPeer*, pode acontecer que quando vai ser entregue a resposta o *Peer* que fez o pedido, este já não está conectado;
- Enquanto um utilizador está a interactuar com a aplicação deve ser possível visualizar os pedidos que outros *Peers* estão a fazer ao *Peer* local. Sugere-se a utilização de modo gráfico (*WinForms*) com duas partes, uma com a interacção com o utilizador e a segunda com a visualização da actividade local do *Peer*. No caso do *Peer* ser *SuperPeer* deve também mostrar os pedidos (getPeers) que lhe estão a ser dirigidos.

Sugestões:

1. Qualquer questão ou dúvida sobre requisitos, deve ser discutida com o professor;
2. Antes de começar a escrever código, defina os requisitos funcionais e não funcionais, desenhe a arquitectura do sistema, as interfaces dos objectos envolvidos bem como os diagramas de interacção mais importantes;
3. Deve utilizar ficheiros de configuração, simplificando assim a construção de um protótipo de demonstração com múltiplos *Peers*;
4. Tenha em atenção o tratamento e propagação de excepções para assim o sistema ser mais fiável e permitir tratar as falhas;
5. Tenha em atenção o tempo de vida de objectos remotos;
6. No relatório discuta e justifique as opções tomadas, por exemplo, o modo de activação de objectos (*SingleCall* ou *Singleton*) bem como as técnicas de manutenção de estado e de controlo de concorrência utilizadas.
7. Quando tiver questões sobre os requisitos, verifique no site *Moodle* se existem "*Frequently Asked Questions*" com esclarecimentos sobre o trabalho.

Luís Assunção