1º Trabalho Prático de Avaliação

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

Nota: O trabalho deve ser realizado até 08 de Maio de 2017, incluindo um relatório que descreva o trabalho com as opções tomadas ao longo da sua realização. (Enviar, os projetos 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, qual o objetivo da solução, os requisitos funcionais e não funcionais (note que os requisitos não são a transcrição deste enunciado), a arquitetura do sistema, os pressupostos nas interações entre as componentes do sistema, nomeadamente as interfaces bem como os aspetos relevantes da implementação dessa arquitetura realçando os pontos fortes e fracos da solução que desenvolveu. Evite descrever código a menos que se justifique nalguma situação, por exemplo, as interfaces dos objetos envolvidos.

Pretende-se a implementação de um sistema distribuído que suporte o armazenamento e posterior acesso de pares **(Chave, Valor)**, com as seguintes especificações:

- Existem N servidores para armazenar pares **(Chave, Valor)**. Os N servidores estão ligados através de um anel virtual gerido por um servidor central ***Ring&Key Manager*** (ver Figura 1);

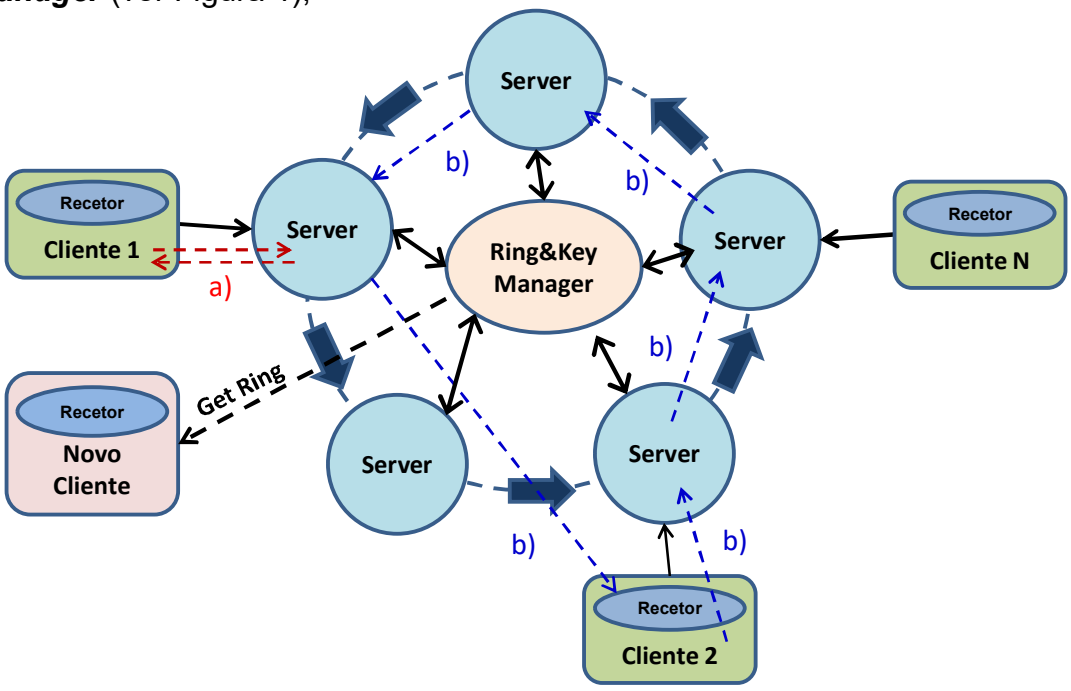

Figura 1 – Esboço geral do sistema

- O gestor ***Ring&Key Manager*** deve garantir que as chaves são únicas, isto é, antes de guardar um novo par os servidores têm de verificar se não existe já um par com essa chave;

- Os objetos ***Chave*** e ***Valor*** são qualquer tipo de dados *Serializable*.
- Cada aplicação cliente do sistema liga-se unicamente a um servidor escolhido a partir da estrutura do anel de servidores, obtida a partir do ***Ring&Key manager*** (ver figura 1).
- Os servidores implementam a interface ***IServer*** que, entre outras operações que ache adequadas, deve oferecer as seguintes funcionalidades:
  - ***storePair:*** Permite armazenar um par (Chave, Valor) no sistema garantindo que, por questões de tolerância a falhas um par é replicado em 3 servidores indicados pelo *Ring&Key Manager*. A atribuição dos servidores para armazenar o par deve obedecer a critérios de garantir eficiência balanceamento de carga, isto é, evitar que um nó do anel fique sobrecarregado só porque tem mais clientes conectados.
  - ***readPair:*** Dada uma chave permite obter o Valor associado. Note que uma réplica do par pretendido pode estar em nós do anel diferentes aquele a que o cliente está conectado;
  - ***deletePair:*** Todas as réplicas do par são removidas do sistema.
- Cada cliente disponibiliza um objeto **Recetor**, que permite receber respostas a pedidos vindos de qualquer servidor.
- Quando um cliente faz um pedido ao servidor a que está conectado, podem acontecer duas situações (ver Figura 1):
  a) O pedido pode ser satisfeito diretamente pelo servidor;
  b) O pedido não pode ser satisfeito, então o servidor reencaminha o pedido pelo anel, até que o mesmo seja satisfeito por outro servidor qualquer que, através do objeto **Recetor,** devolve o resultado. Se após percorrer o anel o pedido regressar ao mesmo servidor, significa que o pedido não pode ser satisfeito.
- A operação ***deletePair*** pode ser realizada por qualquer cliente que conheça a chave de um par;
- Quando um cliente ou um servidor deteta uma falha de algum servidor, informa o ***Ring&Key manager*** do facto, sendo este responsável por reconstruir o anel e notificar os restantes servidores da nova estrutura do anel. Assuma um modelo de falhas em que o servidor ***Ring&Key manager*** nunca falha;
- Assumindo que a estrutura do anel de servidores é conhecida globalmente por todos os servidores, a reconfiguração do anel perante a falha dos servidores deve ser dinâmico, isto é, quando o servidor *k* não consegue contactar o próximo no anel *k+1* deve contactar o servidor *k+2*;
- Embora não seja um requisito obrigatório, recomenda-se que a aplicação Cliente tenha uma interface gráfica (WinForms ou WPF), pois facilita a demonstração da operacionalidade do sistema, nomeadamente a possibilidade de monitorizar o objeto *Recetor*;
- **[Opcional]** Valoriza-se na realização do trabalho e no relatório a possibilidade de o sistema contemplar a possibilidade de o anel poder ser aumentado dinamicamente, isto é, a possibilidade de introduzir um novo servidor ou a reposição de um servidor que anteriormente falhou.

Sugestões:

1. Qualquer questão ou dúvida sobre requisitos, deve ser discutida com o professor;
2. Antes de começar a escrever código desenhe a arquitetura do sistema, as interfaces dos objectos envolvidos bem como os diagramas de interacção mais importantes;
3. Utilizar ficheiros de configuração tanto no servidor do anel como no *Key&Ring Manager*, simplificando assim a construção de um protótipo de demonstração com pelo menos 5 servidores, com diferentes clientes conectados a diferentes servidores;
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, modo de ativação de objetos (*SingleCall* ou *Singleton*) bem como as técnicas de controlo de concorrência utilizadas;
7. Não utilize valores *hardcoded* que possam ser parâmetros de configuração, por exemplo URLS, portos etc.;
8. Quando tiver questões sobre os requisitos, verifique no site *Moodle* da Unidade Curricular de Sistemas Distribuídos se existem *FAQs - Frequently Asked Questions* com esclarecimentos sobre o trabalho.

Nota: Alguns aspetos importantes para a realização do trabalho serão discutidos nas próximas aulas teóricas e práticas, por exemplo, ficheiros de configuração, tempo de vida de objetos etc.

Luís Assunção