Universidade Federal do Ceará – Campus de Quixadá

Disciplina: Sistemas Distribuídos - 2020.2

Prof. Marcos Dantas Ortiz (mdo@ufc.br)

Trabalho de Implementação – Representação Externa

Informações importantes:

- **valor: AP2**
- **equipe: 2 ou 3 alunos**
- Implementar um Serviço Remoto seguindo a arquitetura cliente/servidor e o modelo de protocolo requisição/resposta. O acesso às operações do serviço deve implementado como uma chamada remota a métodos (transparente).
- **Datas importantes**
  - **10/03/21**: **Envio via moodle.** Definição da equipe, do serviço remoto e dos argumentos de requisição e de resposta (parâmetros de entrada e retorno dos métodos) e as exceções que podem ser geradas
  - **17/03/21**: **Envio via moodle**. Implementação da Mensagem (Envelope) e dos Argumentos de Entrada (IN) e Saída (OUT). Caso seja utilizada a IDL (ProtocolBuffer) deve-se enviar arquivos .proto. Para JSON ou XML, deve-se enviar as Classes na linguagem de Origem.
  - **24/03/21**: Troca das mensagens empacotadas.
  - **31/03/21**: **Envio via sippa: Entrega da versão final (código + relatório) e apresentação**

O serviço remoto deve ser implementado respeitando as seguintes restrições:

- O serviço deve prover **pelo menos três métodos remotos diferentes.**
- Os argumentos de entrada e/ou de saída devem ser **complexos** (objetos definidos pelo programador ou estruturas de dados).
  - Dessa forma, poderemos analisar o benefício da representação externa
- Arquitetura cliente-servidor
  - Implementação de Cliente e Servidor em **linguagens diferentes** obterá pontuação extra.
- Protocolo no formato **requisição-resposta** (ver seção 4.4 do livro texto - ilustrado na Figura 1)
  - Modelo síncrono
  - Uma nova requisição apenas será enviada quando a resposta da requisição corrente for recebida.
  - A chegada de uma resposta serve para confirmar o recebimento da requisição
- A comunicação entre cliente e servidor deve ser implementada via *sockets* (**UDP**) que trocam fluxos de bytes (mensagens empacotadas).
- O modelo de falhas UDP deve ser tratado.
  - Lado Cliente
    - Retransmissão de requisições perdidas
      - Uso do *timeout*
  - Lado Servidor
    - Tratamento de duplicadas
      - Histórico de Respostas
        - Caso as operações não sejam idempotentes
- A requisição deve ser implementada como uma chamada remota a método (transparente).
  - Para isso, utilize *Proxy*, Esqueleto e Despachante.

- O despachante deve ser genérico – **utilize Reflexão** para instanciar o objeto adequado e chamar o método solicitado.
- O uso de reflexão no lado servidor é necessário para torná-lo flexível à adição/remoção de serviços (métodos).

- Foi sugerido em sala de aula que a representação externa utilizada no empacotamento/desempacotamento dos argumentos e das mensagens de requisição e resposta fosse via *protocol buffers*, no entanto, também serão aceitos outras soluções como XML e JSON.

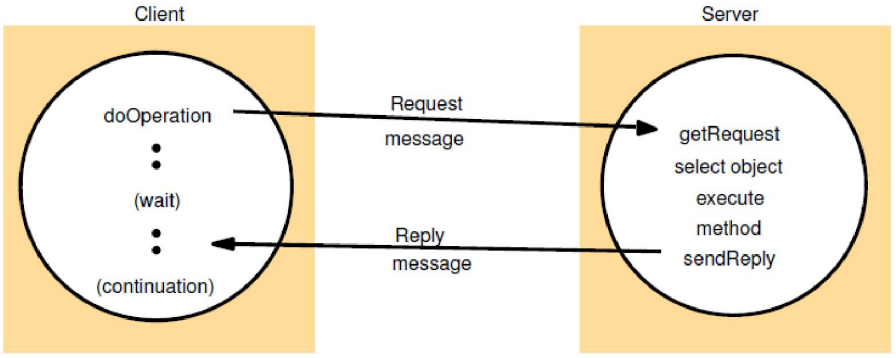

Figura 1 - Comunicação requisição-resposta – (Coulouris 4ed).

O uso dos **métodos sugeridos no livro texto**, que implementam o protocolo requisição-resposta, **deve ser seguido**, mas pequenas alterações em suas assinaturas podem ser analisadas:

- ***public byte[] doOperation (RemoteObjectRef o, int methodId, byte[] arguments)****:*
  - envia uma mensagem de requisição para o objeto remoto e retorna a resposta. Os argumentos especificam o objeto remoto, o método a ser chamada e os argumentos para aquele método.
- ***public byte[] getRequest ():***
  - obtém uma requisição de um cliente através de uma porta servidora.
- ***public void sendReply (byte[] reply, InetAddress clientHost, int clientPort):***
  - envia a mensagem de resposta para o cliente, endereçando-a a seu endereço IP e porta.

As mensagens de requisição/resposta devem ser empacotadas como ilustrado na Figura 2. A função de cada elemento da mensagem é descrita também na seção 4.4 do livro texto.

messageType	*int (0=Request, 1= Reply)*
requestId	*int*
objectReference	*RemoteObjectRef*
methodId	*Int*
arguments	*array of bytes*

Figura 2 - Estrutura da mensagem de requisição e resposta (Coulouris 4ed).

Os elementos *objectReference* e *methodID* podem ser ***Strings*** que representam, respectivamente, o nome do objeto que fornece o serviço e o nome do método a ser invocado.

Cada equipe **deve entregar**, além do código fonte, um **relatório** descrevendo o serviço remoto implementado e os arquivos *.proto* (ou demais IDLs) utilizados para representar os dados. A entrega deve ser feita via ***upload*** **no SIPPA**.

Não há necessidade de prover Interface Gráfica com o Usuário (GUI). Interação em modo texto é suficiente.