Dúvidas

Tarefas de Programação

Bruna Michelly de O. S. Cordeiro – brunamos.bm@gmail.com

Tarefas de Programação

1. Implementação de Referências Remotas
2. Descoberta automática do servidor de aplicação
3. Melhorias no Servidor de Diretório
4. Servidor de Nomes Hierárquico

Bruna Michelly - brunamos.bm@gmail.com

Implementação de Referências Remotas

Bruna Michelly - brunamos.bm@gmail.com

Referências Remotas

Objetivo: ***compreender o princípio da passagem de referências remotas como parâmetro em sistemas distribuídos***

Esta tarefa consiste em executar o exemplo da Fig. 4.21 para reproduzir o cenário mostrado na Fig. 10

- Um cliente que possui a referência para o objeto remoto passa essa referência para outro cliente, que, com isto passa a ter também a capacidade de fazer chamadas para o mesmo objeto remoto.
- A execução deve ser feita utilizando três máquinas distintas, alocadas na nuvem AWS

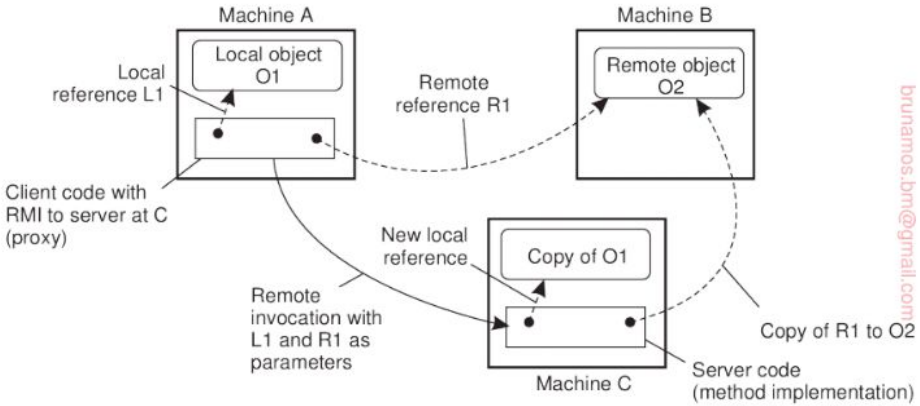


Figure 4.10: Passing an object by reference or by value.

Descoberta automática do servidor de aplicação

Tarefa 2

Bruna Michelly - brunamos.bm@gmail.com

Descoberta automática do servidor de aplicação

Utilização da biblioteca **rpyc** para implementar o cenário descrito no slide 27 - binding entre cliente e servidor.

- *Nesta tarefa, você deve implementar o servidor de diretório e as interações 2, 3 e 5.*
- *O Server deve registrar o endereço IP e o número de porta (juntamente com seu nome legível) no servidor de diretório.*
- *O cliente usa o nome legível do servidor para descobrir o endereço IP e o número de porta do servidor por meio de uma chamada remota (lookup, via RPC) ao servidor de diretório.*
- *Em seguida, o cliente faz uma sequência de chamadas (RPCs) para demonstrar o acesso ao servidor.*

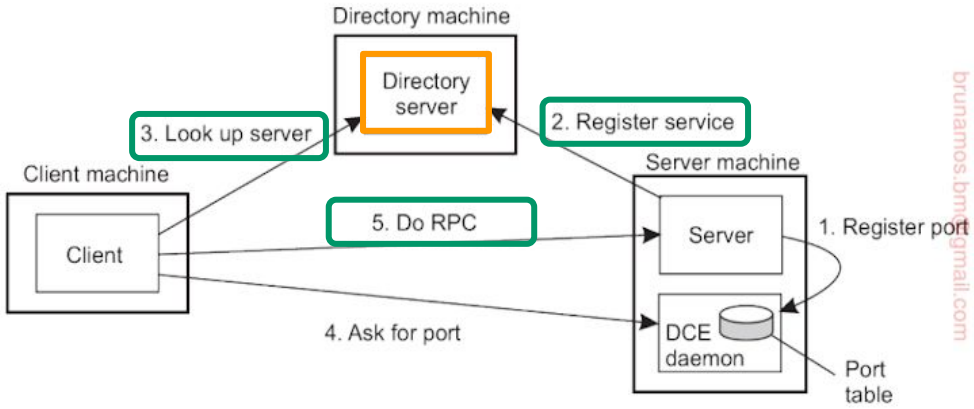


Figure 4.17: Client-to-server binding in DCE.

Descoberta automática do servidor de aplicação

Utilização da biblioteca **rpyc** para implementar o cenário descrito no slide 27 - binding entre cliente e servidor.

- *Na Figura há um servidor simples para a estrutura de dados* **DBList**.
- *Nesse caso, ele tem duas operações expostas:*
  - ***exposed_append*** *para anexar elementos;*
  - ***exposed_value*** *para exibir o que está atualmente na lista.*
- *Quando uma conexão é feita com o servidor, uma nova instância de DBList é criada e o cliente pode anexar imediatamente valores à lista.*

Servidor

```
1  import rpyc
2  from rpyc.utils.server import ForkingServer
3
4  class DBList(rpyc.Service):
5    value = []
6
7    def exposed_append(self, data):
8      self.value = self.value + [data]
9      return self.value
10
11   def exposed_value(self):
12     return self.value
13
14 if __name__ == "__main__":
15   server = ForkingServer(DBList, port = 12345)
16   server.start()
```

Cliente

```
1  import rpyc
2
3  class Client:
4    conn = rpyc.connect(SERVER, PORT) # Connect to the server
5    conn.root.exposed_append(2)       # Call an exposed operation,
6    conn.root.exposed_append(4)       # and append two elements
7    print conn.root.exposed_value()   # Print the result
```

Note 4.6 (More information: Language-based RPC in Python) - Pg 185

Bruna Michelly - brunamos.bm@gmail.com

Descoberta automática do servidor de aplicação

- *Nesta tarefa, você deve implementar o servidor de diretório e as interações 2, 3 e 5.*
- *O Server deve registrar o endereço IP e o número de porta (juntamente com seu nome legível) no servidor de diretório.*
- *O cliente usa o nome legível do servidor para descobrir o endereço IP e o número de porta do servidor por meio de uma chamada remota (lookup, via RPC) ao servidor de diretório.*
- *Em seguida, o cliente faz uma sequência de chamadas (RPCs) para demonstrar o acesso ao servidor.*

```
1
2 import rpyc
3 from constRPYC import *
4 from rpyc.utils.server import ForkingServer
5
6 class ServerDirectory(rpyc.Service):
7     registryDirectory = {}
8
9     def exposed_register(self, server_name, ip_adress, port_number):
10        self.registryDirectory[server_name] = (ip_adress, port_number)
11        print (self.registryDirectory)
12
13    def exposed_lookup(self, server_name):
14        if server_name in self.registryDirectory:
15            return self.registryDirectory[server_name]
16        else:
17            return None
18
19 if __name__ == "__main__":
20     server_dir = ForkingServer(ServerDirectory, port ='')
21     server_dir.start()
22
```

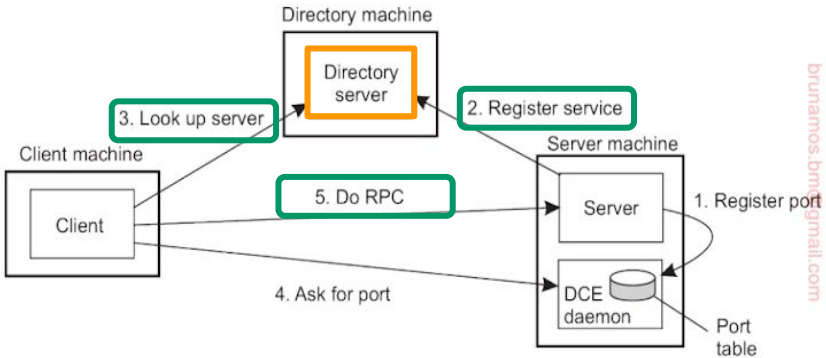


Figure 4.17: Client-to-server binding in DCE.

Melhorias no Servidor de Diretório

Tarefa 3

Melhorias no Servidor de Diretório para:

- *Registrar novamente um nome já existente;*
- *Evitar o lookup de nomes não existentes;*
- *Remover o registro de um nome existente.*

Para completar o exercício, acrescente novas funcionalidades ao servidor da aplicação para excluir e consultar elementos da lista.

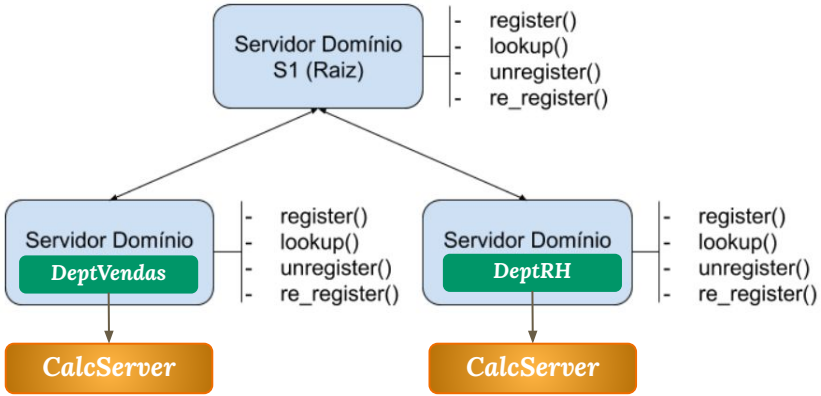
Servidor de Nomes Hierárquico

Tarefa 4

Bruna Michelly - brunamos.bm@gmail.com

Servidor de Nomes Hierárquico

- O ***servidor de nomes raiz não possui nome.***
- ***Cada servidor de nomes de segundo nível possui um nome único***, *o qual deve ser registrado no servidor de nomes raiz.*
- *Nomes de* ***servidores de aplicação (oferece o serviço)*** *são sempre registrados localmente nos servidores de nomes de segundo nível, o que significa que o* ***servidor de nomes raiz serve apenas para registrar os nomes dos servidores de segundo nível.***
- ***Servidores de aplicação diferentes podem ser registrados com o mesmo nome em servidores de nomes diferentes****. Mas observe que os FQNs (fully qualified name) sempre serão diferentes.*
- A ***resolução de nomes global pode ser feita tanto recursivamente quanto iterativamente****; o método escolhido deve ficar claro no relatório da tarefa.*
- ***O servidor de nomes raiz deve ser conhecido a priori por todos os clientes.*** *Os servidores de nomes de segundo nível devem ser descobertos a partir do servidor de nomes raiz.*



Obs.: Um FQN tem sempre este formato: <NamingContext>:<LocalName>

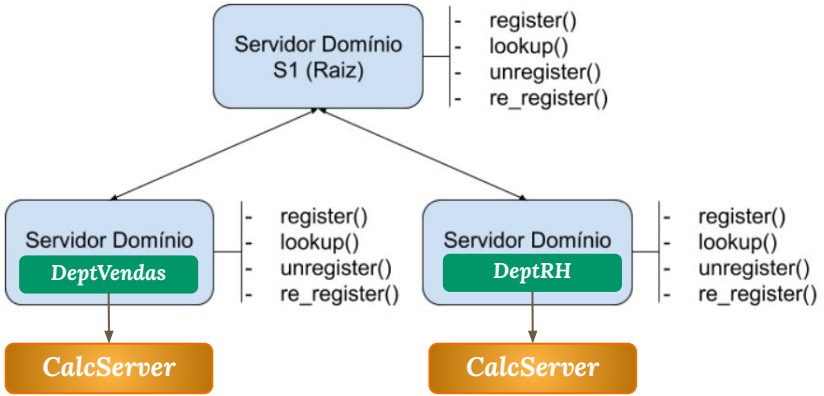
Servidor de Nomes Hierárquico

Este serviço de nomes hierárquico é composto por um *servidor de nomes raiz e dois servidores de nomes de segundo nível*.
Todos os servidores possuem exatamente a mesma interface, com as seguintes operações:

Operation name	*Parameters*	*Return value*	*Description*
register	(name, address)	Fully Qualified Name (or error message)	Registre um novo nome e associe-o ao endereço fornecido (composto por endereço IP e número da porta). Retorna FQN (Fully Qualified Name, ou seja, um nome completo no contexto do formulário: nome) Nota: nesta versão do serviço de nomenclatura hierárquica, o registro do nome é sempre local, ou seja, register () deve ser chamado exatamente no servidor onde o nome será registrado.
lookup	(name)	address (or error message)	Resolve um nome. Retorna o endereço (IP e porta #) associado ao nome. O parâmetro de nome pode ser um nome simples, caso em que é resolvido apenas localmente, ou um FQN, caso em que pode ser resolvido consultando os outros servidores de nomenclatura (seguindo a hierarquia). Nota: a resolução de nome pode ser local (onde lookup () recebe um nome simples e é chamado no mesmo servidor de nomenclatura onde o nome está registrado) ou global (onde lookup () recebe um FQN e pode ser chamado em um servidor de nomenclatura diferente do que aquele onde o nome foi registrado)
unregister	(name)	Ok message (or error message)	Exclui um nome registrado anteriormente.
re_register	(name, address)	fully qualified name (or error message)	Atualiza o endereço associado a um nome existente.

Servidor de Nomes Hierárquico

- *Para demonstrar o funcionamento do serviço de nomes hierárquico, construa pelo menos três tipos de servidor de aplicação:*
    - *Por exemplo: servidor de calculadora, servidor de previsão do tempo e servidor de hora.*
- *Crie instâncias distintas desses servidores de aplicação e as registre com seus respectivos nomes nos dois domínios de segundo nível.*
- *Demonstre a descoberta de serviços com um cliente que:*
    - *faz uma sequência de consultas locais e consultas globais;*
    - *realiza uma sequência de chamadas aos servidores de aplicação descobertos.*



Obs.: Um FQN tem sempre este formato: <NamingContext>:<LocalName>

Obrigada!

Bruna Michelly - brunamos.bm@gmail.com