Cássio de Olivera Ferraz

Trabalho de Sistemas Distribuídos

Petrópolis

2015, v-1.0

Cássio de Olivera Ferraz

Trabalho de Sistemas Distribuídos

Trabalho sobre sistemas distribuídos e suas tecnologias.

Universidade Estácio de Sá - UNESA

Sistemas de Informação

Programa de Graduação

Petrópolis

2015, v-1.0

Lista de ilustrações

Lista de abreviaturas e siglas

RPC	Remote procedure call
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
P2P	peer-to-peer
DHT	Distributed hash table
CORBA	Common Object Request Broker Architecture
IDL	Interactive Data Language
WSDL	Web Services Definition Language
SOA	Service-oriented architecture
SOAP	Simple Object Access Protocol

Sumário

Introdução

Este trabalho tem a missão de explicar alguns temas de sistemas distribuídos, estes sistemas são fundamentais para a comunicação, integração e processamento de dados via internet ou rede local. Ao longo do trabalho abordaremos os principais tópicos tais como: WebServices, Padrões de Sockets entre outros.

Cássio de Oliveira Ferraz

1 RPC

O RPC é um protocolo de execução remota para computadores ligados em rede e este pode ser implementado sobre diferentes protocolos de transporte. Sua implementação depende de qual protocolo de transporte irá operar.

No TCP, não precisa se preocupar com time-outs, retransmissões e duplicatas. Já no UDP é preciso prestar atenção aos time-outs, retransmissões e duplicatas. Quando trabalhamos com RPC, devemos manipular os erros e falhas no servidor remoto ou na rede. No RPC não trabalhamos com variaveis globais, pois estas não tem acesso ao lado cliente, com isso utilizamos somente parâmetros. A autênticação é sempre necessária na RPC.

Passos da RPC

- Coleta dos parâmetros;
- Forma a mensagem;
- Envia a mensagem;
- Espera a resposta;
- Devolve a resposta através dos parâmentros;

O RPC foi projetado para facilitar a comunicação entre cliente x servidor, as funções contidas dentro do RPC são acessíveis por qualquer programa que se comunicar usando a tecnologia cliente x servidor. A figura abaixo demonstra a arquitetura RPC.

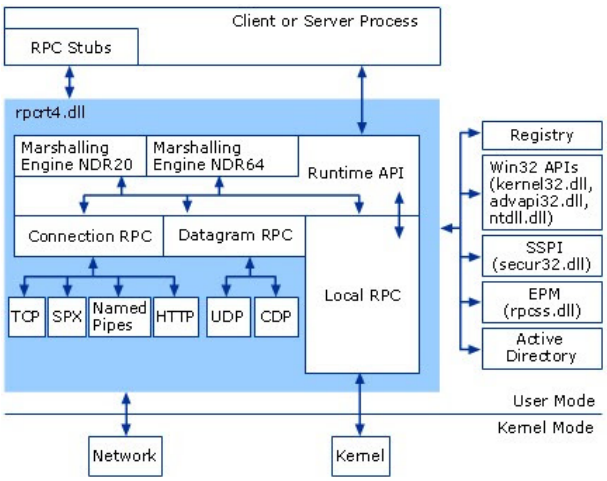

Figura 1: Diagrama RPC

COMPONENTES RPC

Cliente x Servidor
Programa ou serviço que inicia ou responde a uma chamada RPC. RPC stubs: Programa usado pelo cliente ou servidor para iniciar uma requisição RPC.

Sockets
Em sistemas distribuidos quando há comunicação interprocesso, os mais utilizados são os sockets (Serviços de transporte) para garantir que processos se comuniquem na troca de dados ou em acessos a recursos em servidores remotos. Esta é a forma mais popular da utilização do protocolo TCP/IP, todos os mecanismos de sockets são gerenciados pela camada de transporte e existem diversas APIs sockets das quais as mais populares são do ambiente Unix, bem como a WinSock do windows.

Definição de Socket: Socket é o fim de um canal de comunicação entre dois softwares rodando em rede. Cada socket possui endereços de EndPoint:

- Endereço Local (numero da porta): esse se refere ao endereço da porta de comunicação para camada de transporte.
- Endereço global (Nome host): este se refere ao endereço do computador(host) na rede

PROTOCOLOS

TCP E UDP são protocolos de transporte e ambos usam a camada IP como camada de rede.

Comunicação

Servidor
O servidor efetua a criação de um socket, associa o socket a um endereço local, aguarda conexões do cliente, aceita as conexões, faz a leitura das requisições, opcionalmente envia resposta e por fim fecha o socket.

Cliente
Este também efetua a criação do socket, estabelece a conexão, envia a requisição opcionalmente aguarda e resposta e fecha o socket.

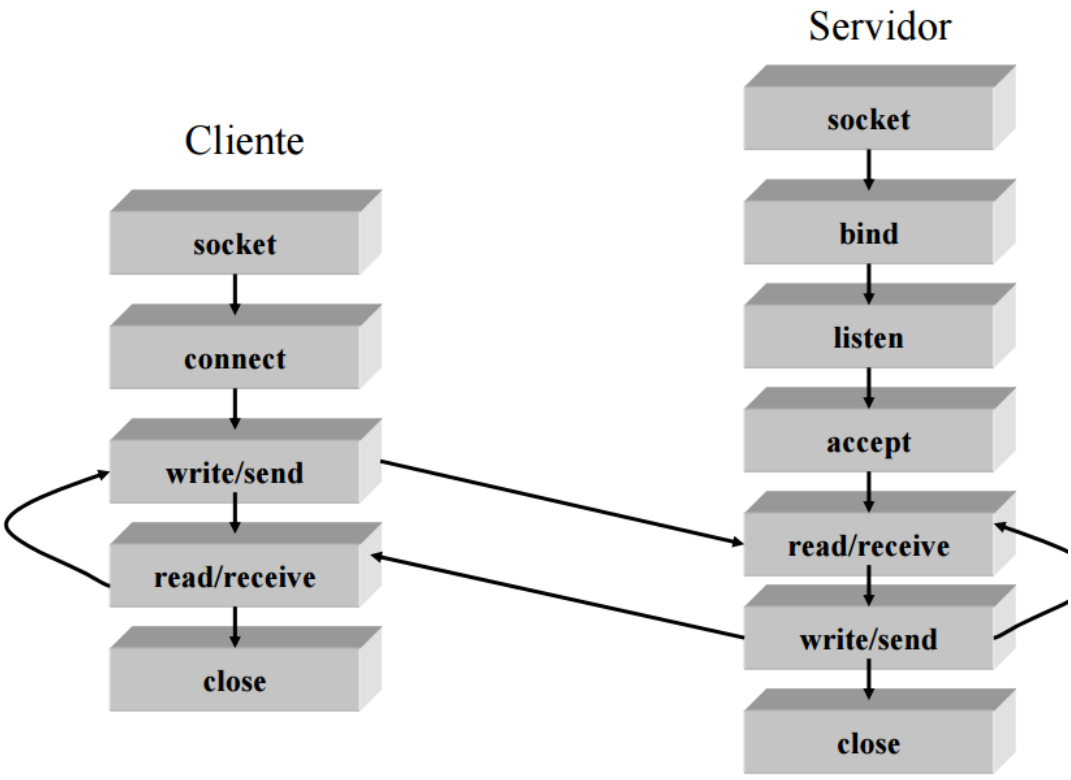

Figura 2: Comunicação Cliente x Servidor

2 P2P

DHT Tapestry

Transporte
A camada de transporte fornece os canais de comunicação de um nó para outro. Utiliza a funcionalidade nativa do sistema operacional é possivel implementar alguns canais. Atualmente é suportado o TCP/IP e UDP/IP.

Camada de Enlace
Acima da camada de transporte, a enlace fornece instalações de datagramas seguros mas não confiáveis para as camadas acima, incluindo a fragmentação e remontagem de grandes mensagens.

Router
Enquanto a camada de enlace fornece os serviçoes de básicos de rede o router implementa as funcionalidades originais da tapestry. Nessa camada está incluída a tabela de roteamento e ponteiros de objetos locais.

DHT Chord

É um protocolo e algoritmo p2p, onde é armazenado pares-chave associados a outros computadores, conhecidos como nós. O Chord especifica como as chaves são associadas aos nós.

DHT Pastry

Sistema de roteamento proposto por Roston e Druschel em 2001. Pastry é semelhante ao chord e seu principal objetivo é criar uma estrutura centralizada de p2p, tornando o encaminhamento da mensagem mais eficiente. Sua estrutua não está organizada em anel como a chord, mas sim baseada nos identificadores numericos.

Tabela Hash

Pares de chave-valor são armazenados nessa tabela e qualquer nó que participa desta tabela pode recuperar facilmente um dado associado a outra chave. Tem como responsabilidade, manter o mapeamento de chaves-valores, como são distribuidas entre

os nós. Elas gerenciam um grande número de nós e ainda cuidam das chegadas, saídas e falhas. Estas tabelas são usadas na construção de sistemas mais complexos, como por exemplo os sitemas de arquivos distribuidos, compartilhamento p2p entre outros.

3 WebServices

CORBA

Este padrão foi implementado em 1991, Como uma arquitetura que provê uma estruta para integrar objetos diferentes. Mas ele é apenas um integrante do (OMA).

O corba ajuda no gerenciamento e na manutenção de objetos. Ele fornece interfaces para criação de objetos, manipular a segurança dos mesmos e determinar suas localidades. Uma vantagem do corba para o desenvolvedor é que ele permite que o desenvolvedor concentre o foco em seus objetos, sem ter preocupação com os serviços a nível de sistema.

Facilidades do Corba

As facilidades de objetos definem modelos que oferecem interfaces a outros serviços. Estes modelos estão divididos em grandes categorias que incluem interfaces de aplicação, interfaces de domínio e facilidades de redes.

As interfaces de domínio consistem em facilidades para usuários finais em domínios de aplicações específicas. Sendo que neste caso, um domínio refere-se a um mercado vertical específico ou uma indústria.

IDL

Utilizada pelo CORBA a IDL é uma linguagem baseada em c++ que não possui algoritmos nem variaveis, ou seja é somente declarativa e independente da linguagem de programação escolhida. A IDL possibilita a interoperabilidade entre os diversos sistemas, uma vez que a separação é definida entre a interface e a execução. A interface é bem definida enquanto o código fonte e dados permanecem ocultos para o resto do sistema.

WSDL

É uma padrao que descreve webservices, que tem o objetivo de eliminar ao máximo a necessidade de comunicação entre as partes envolvidas em uma integração de dados. Podemos verificar nos webservices, além das informações contidas em seu conteúdo, o seu endereço real onde o serviço está hospedado, valores do header da requisição http, tais como a SOAPAction disponivel na tag SOAPAction entre outras, e todas elas juntas formam uma requisição http válida ao servidor de destino.

Apesar de todas as vantagens do uso adequado do WSDL, o mesmo é normalmente negligenciado pelas empresas de TI devido a sua complexidade. Ao invés disto, utilizam-se

do que se chama de abordagem de desenvolvimento bottom-up, a qual você escreve o seu Web Service utilizando sua linguagem de programação favorita e a mesma gera o documento WSDL para você. A princípio, isso parece ser muito bom, mas em longo prazo, isso pode gerar problemas como:

- Em uma migração de tecnologia (linguagem de programação, troca de servidor de aplicação), pode ser que o novo Web Service, criado para substituir o anterior, não aceite a requisição que está sendo enviada pelos programas clientes, que pode vir a parar de funcionar.
- Cada linguagem de programação ou ferramenta vai gerar um documento WSDL diferente.
- Dependendo da linguagem de programação ou da ferramenta utilizada, pode ser que sejam gerados tipos de dados não suportados oficialmente pela W3C (Órgão "regulamentador" do padrão WSDL, SOAP e XML).

SOA

Uma SOA é um modelo de projeto com um conceito profundamente amarrado à questão do encapsulamento de aplicação. A arquitetura resultante estabelece essencialmente um paradigma de projeto, no qual web services são os blocos de construção chave. Isto quer dizer que ao migrar a arquitetura da sua aplicação para uma SOA, estabelece-se um compromisso com os princípios de projeto de web services e a tecnologia correspondente, como partes fundamentais do seu ambiente técnico. Uma SOA baseada em XML web service é construída sobre camadas de tecnologia XML estabelecidas, focada em expor a lógica de aplicação existente como um serviço fracamente acoplado. Para apoiar este modelo, uma SOA promove o uso de um mecanismo de discovery por serviços via um service broker ou discovery agent.

É importante se conscientizar quanto ao acréscimo de complexidade de projeto introduzido pelo SOA. Mais ainda do que em um ambiente n-camada, projetistas de aplicação devem considerar de uma forma completa como a introdução de serviços vai afetar dados existentes e modelos de negócio.

SOAP

SOAP é um protocolo de transferência de mensagens em formato XML para uso em ambientes distribuídos. O padrão SOAP funciona como um tipo de framework que permite a interoperabilidade entre diversas plataformas com mensagens personalizadas. Aplicando este padrão em Web Services, geralmente usa-se o WSDL para descrever a estrutura das mensagens SOAP e as ações possíveis em um endpoint. Uma das maiores vantagens disso

é que várias linguagens e ferramentas conseguem ler e gerar mensagens facilmente. Várias linguagens de programação permitem a geração de objetos de domínio, Stubs e Skeletons a partir da definição do WSDL, permitindo a comunicação remota via RPC através de chamadas a métodos remotod, inclusive com argumentos complexos, como se fossem chamadas locais. O problema desse padrão, é que ele adiciona um overhead considerável, tanto por ser em XML quanto por adicionar muitas tags de meta-informação. Além disso, a serialização e desserialização das mensagens pode consumir um tempo considerável.

Considerações finais

Existem diversos sistemas distribuídos, cabe o estudo e a necessidade de cada um para saber qual metodologia implantar e quais padrões utilizar. Com sistemas interligados conseguimos muitos recursos que nao conseguiriamos sem esses padrões.

Referências

https://technet.microsoft.com/pt-br/library/cc738291(v=ws.10).aspx
http://professor.ufabc.edu.br/ francisco.massetto/sd/03-Aula3-Sockets.pdf
http://www.ccsenet.org/journal/index.php/cis/article/viewFile/4282/3729
http://stackoverflow.com/questions/144360/simple-basic-explanation-of-a-distributed-hash-table-dht
http://www.devmedia.com.br/introducao-as-tecnologias-web-services-soa-soap-wsdl-e-uddi-parte1/2873
http://www.devmedia.com.br/wsdl-simplifique-a-integracao-de-dados-via-web-service/30066

Índice