VOLTAR

Tipos de Sistemas Distribuídos

Comparativo entre algumas arquiteturas e o middleware

NESTE TÓPICO

Neste tópico, serão apresentados os tipos de arquiteturas adotadas em projetos de sistemas distribuídos tais como os modelos de sistemas de computação distribuída, sistemas de informação distribuídos e os sistemas distribuídos pervasivos. Para cada um destes modelos distribuídos, veremos as seguintes arquiteturas:

- Computação em Cluster
- Computação em Grade
- Sistemas de Processamento de Transações
- Integração de Aplicações Empresariais
- Computação em Nuvem
- Arquitetura Orientada a Serviços (SOA)
- Computação Pervasiva
- Computação Orientada a Transações
- Redes de Sensores

Sistemas de Computação Distribuída

Os sistemas de computação distribuída são modelos projetados com objetivos de compartilhamento de recursos, desempenho, confiabilidade, tolerância a falhas e escalabilidade. Dessa forma, a computação distribuída visa acrescentar o número de réplicas (cópias) de processadores disponíveis na rede para atender aos objetivos do sistema.

A seguir, veremos os modelos de computação distribuída denominados por Cluster e Grade.

Computação em Cluster

A computação em cluster é um modelo de computação distribuída de alto desempenho, mas com restrições bem definidas em sua configuração e arquitetura. Tais restrições determinam que seja o mesmo sistema operacional em todos os computadores, hardwares semelhantes e sejam configurados em uma rede local de alta velocidade (veja na Figura 1). Isto é, trata-se de um ambiente computacional homogêneo. A sua popularização se deu a partir da melhoria no desempenho dos computadores pessoais e a redução do seu custo de aquisição.

O cluster é formado por um computador mestre (nó mestre) que é o responsável pelo controle, envio de tarefas e o efetivo balanceamento de carga dos computadores escravos (nós escravos). Dessa forma, o nó mestre possui um serviço de gerenciamento do cluster, bibliotecas para programação paralela e o seu sistema operacional local. Os nós escravos possuem os componentes do serviço de processamento paralelo e o seu sistema operacional local.

Para sua adoção efetiva visando os ganhos em desempenho, confiabilidade, escalabilidade e tolerância a falhas, a aplicação (sistema de informação) que executará sobre o cluster deverá ser programado de tal maneira que possa consumir os recursos do cluster. Isto é, cada tarefa será inicializada por um usuário diferente através da aplicação, poderá ser executada paralelamente em cada nó escravo. Por exemplo, um usuário A solicita um relatório, o usuário B solicita uma alteração de telefone e o usuário C solicita uma exclusão de dados. Cada uma destas tarefas poderão ser executadas em três processadores ou nós diferentes.

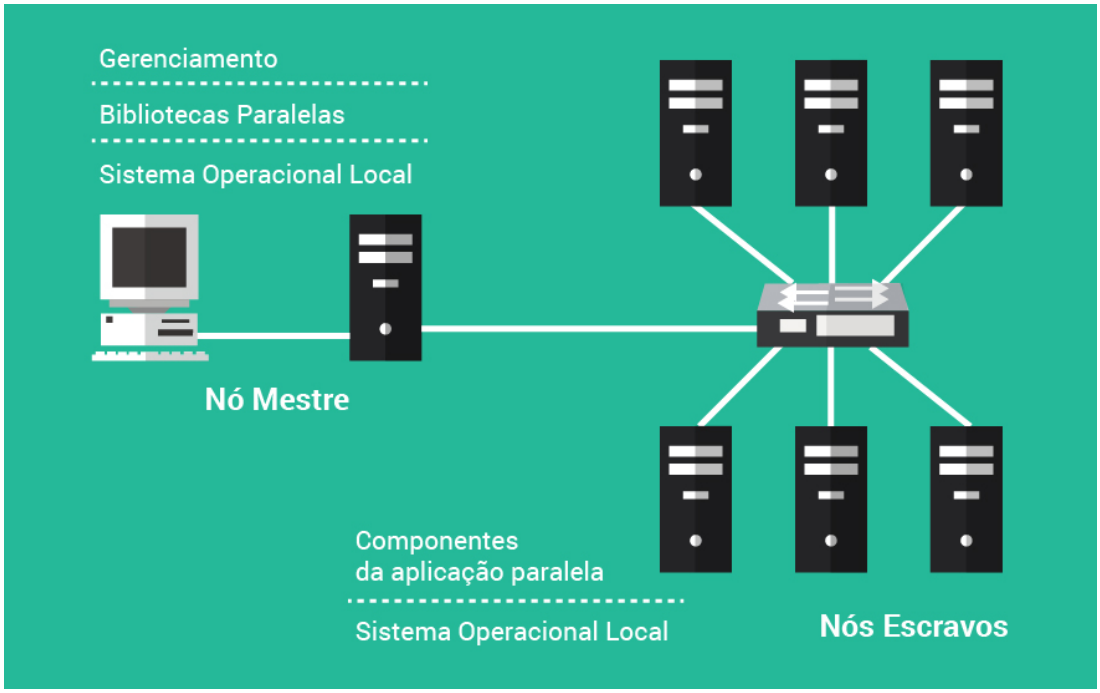


Figura 1 - Arquitetura básica da Computação em Cluster

Fonte: TANENBAUM, A. S.; STEEN, M. V.; Sistemas Distribuídos; Princípios e Paradigmas; Pearson Prentice Hall; 2ª edição; 2007.

O cluster Newton (Figura 2) pode atingir uma performance teórica de pico de 13 TFlop/s (*FLoating-point Operations Per Second* - Operações de Ponto Flutuante por Segundo) ou Trilhões de Cálculos Matemáticos por Segundo.

No site Top500 (www.top500.org), são realizadas análises semestrais de desempenho dos supercomputadores ao redor do mundo. Na lista do *ranking* dos três melhores pontuados, encontramos os supercomputadores com o pico teórico de 44 PFlop/s (TIANHE-2), 18 PFlop/s (TITAN - CRAY XK7) e 17 PFlop/s (SEQUOIA - BLUEGENE/Q). Apesar da performance surpreendente do TIANHE-2, o mesmo foi projetado para alcançar a performance teórica de 54 TFlop/s. Em todos os casos, o sistema operacional que executava em todos os computadores da lista é baseado no GNU/Linux.

O maior supercomputador brasileiro é o Cray XT6 do INPE (Instituto Nacional de Pesquisas Espaciais) atinge a performance teórica de pico de 258 TFlop/s, mas com desempenho efetivo (real) de 17 TFlop/s.

Figura 2 - Cluster Newton - CESUP - UFRGS

Fonte: http://www.cesup.ufrgs.br/recursos-e-servicos/hardware-1/sun-fire

O cluster Altix (Figura 3) pode atingir uma performance teórica de pico de 16 TFlop/s.

Figura 3 - Cluster Altix - CESUP - UFRGS

Fonte: http://www.cesup.ufrgs.br/recursos-e-servicos-1/hardware-1/sgi-altix

Computação em Grade (*Grid Computing*)

Com o crescimento da Internet, do número de sites e aplicações móveis e ubíquas, e do número de serviços oferecidos na Web, proporcionalmente o número de usuários também cresceu. Isso quer dizer que muitos usuários distantes geograficamente estão acessando aplicações distribuídas, mas que ainda estão muito longe deles. A distância numa rede que sofre com picos de congestionamento em determinados horários é um fator que deve ser levado em consideração. Quanto mais distante os usuários estão dos recursos computacionais que dão suporte às aplicações, maior será a latência da rede. Em outras palavras, o tempo de resposta (TR) será maior e causará impaciência aos usuários, podendo levar a aplicação ao total descrédito. Em termos mais técnicos e discutidos anteriormente, as aplicações com problemas de TR poderão ficar indisponíveis resultando em problemas de confiabilidade e desempenho de devido ao aumento contínuo da concorrência.

A computação em grade ou em grelha é um modelo de computação distribuída de alto desempenho. O modelo propõe a configuração de organizações virtuais (OV) ou federações. Em cada OV contém um grupo de

servidores dedicados para executar tarefas da localidade em questão. Assim, o modelo permite a configuração em rede de longa distância e não tem restrições quanto ao hardware, sistema operacional e rede. Isto é, trata-se de um ambiente computacional heterogêneo.

Alinhado ao problema inicial deste tópico, veja que a Figura 4 mostra o modelo da grade JPPF (*JavaParallel Processing Framework* - *Framework* em Java para Processamento Paralelo) contendo três JPPF *Servers*. Cada JPPF *Server* e os *Nodes* (nós) formam uma OV, resultando num modelo de rede P2P e Cliente/Servidor com escalonamento de recursos de processamento. Além disso, esta grade aberta é desenvolvida em Java e pode ser executada em sistemas operacionais baseados no Unix, GNU/Linux e Windows, o que garante maior portabilidade da solução. Se o acesso a esta grade for através de *Web Services (*WS), qualquer aplicação - independentemente da linguagem de programação que dê suporte ao WS - poderá acessar aos recursos de processamento paralelo da grade. Isso garante maior portabilidade, compartilhamento de recursos e distribuição da solução.

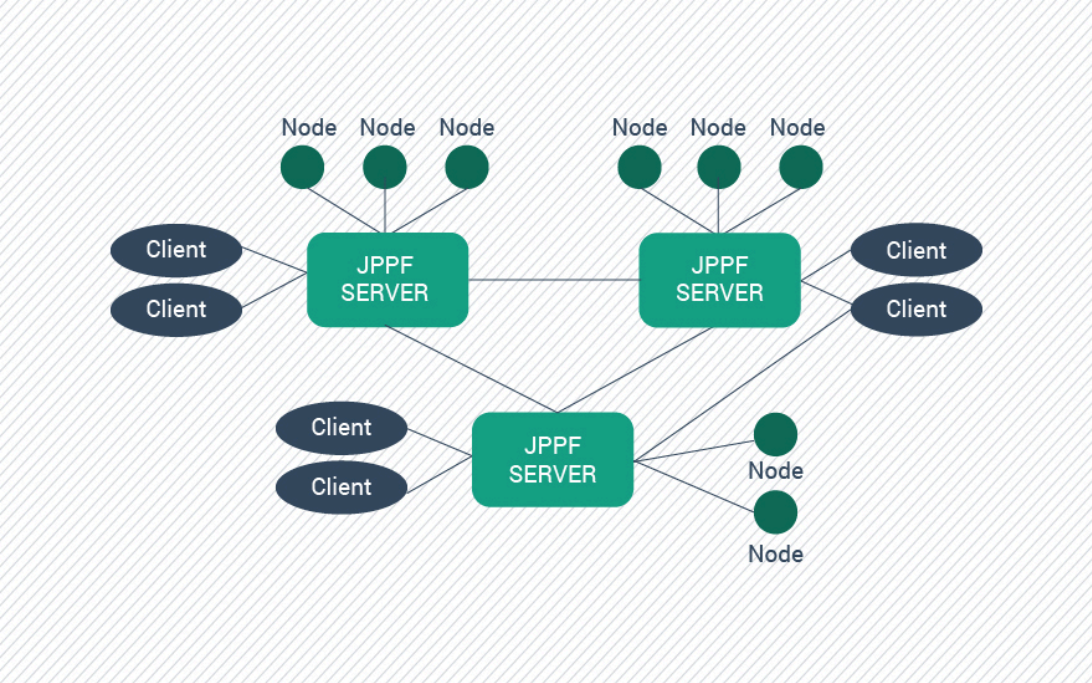


Figura 4 - Grade Computacional JPPF

Fonte: XIONG, Jing; WANG, Jianliang; XU, Jianliang. Research of Distributed Parallel Information73 Retrieval Based on JPPF. In: 2010 International Conference of Information Science and Management Engineering. 2010. v.1, p. 109-111.

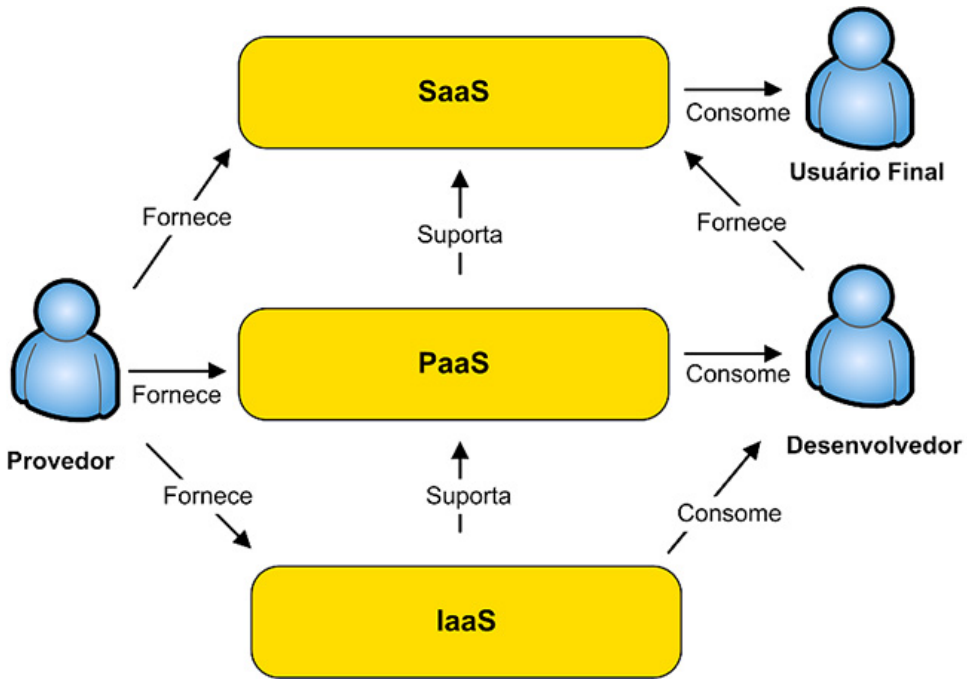
Computação em Nuvem (*Cloud Computing*)

A computação em nuvem é um modelo de arquitetura distribuída que oferece um ambiente de serviços de infraestrutura e de sistemas computacionais. A ideia inicial da nuvem veio para propor serviços limitados aos usuários que desejavam armazenar e acessar documentos e arquivos de áudio/vídeo através da Internet. Com a disseminação da nuvem, atualmente ela abrange também serviços de infraestrutura e desenvolvimento de sistemas que são cobrados mediante o uso destes recursos de tecnologia da informação.

Para lidar com diferentes nichos de mercado e usuários, o projeto das nuvens foi revisto quanto a sua infraestrutura e formas de utilização e pagamento. Dessa forma, a estrutura de serviços foi dividida em três camadas que atendem a diversos usuários e empresas da seguinte maneira:

1. *Software as a Service* (SaaS) - Software como um Serviço: Camada de software que oferece aplicativos aos usuários tais como webmail, aplicativos de áudio, vídeo e armazenamento de dados.
2. *Plataform as a Service* (PaaS) - Plataforma como um Serviço: Camada para o ambiente de desenvolvimento de sistemas. Possui frameworks para desenvolvimento de software e sistemas gerenciadores de bancos de dados.
3. *Infraestructure as a Service* (IaaS) - Infraestrutura como um Serviço: Camada para a virtualização de hardware, sistemas operacionais, relatórios gerenciais e dispositivos de armazenamento de dados (*storages*).

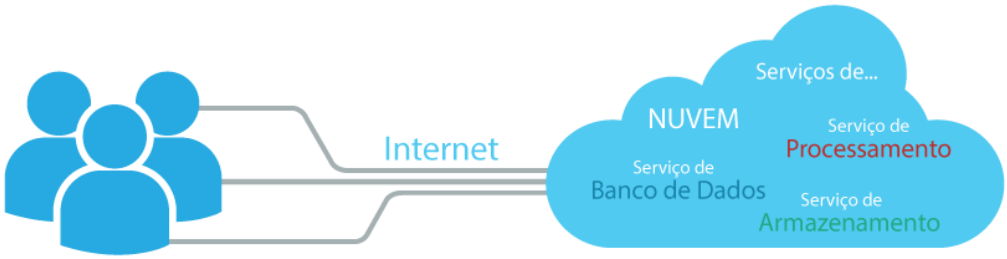
A Figura 5 mostra os diferentes serviços e usuários consumidores para demonstrar o que discutimos acima.



Serviços da Computação em Nuvem

Fonte: SOUSA, Flávio RC; MOREIRA, Leonardo O.; MACHADO, Javam C. Computação em nuvem: Conceitos, tecnologias, aplicações e desafios. II Escola Regional de Computação Ceará, Maranhão e Piauí (ERCEMAPI), p. 150-175, 2009.

A Figura 6 demonstra o acesso de usuários (à esquerda) aos recursos e serviços da nuvem (à direita). Os usuários utilizam, em sua grande maioria, serviços do SaaS para ter acessos aos aplicativos disponíveis na nuvem.



Estrutura da computação em nuvem e a comunicação com os usuários

Fonte: GUIMARÃES, Leandro; CORRÊA, Matheus; OSÓRIO, Tomás. Computação em Nuvem.

A Figura 7 representa o acesso de empresas ou instituições (à esquerda) aos recursos e serviços da nuvem (à direita). Estas instituições utilizam, comumentemente, os serviços do SaaS, PaaS e o IaaS. Não necessariamente as três camadas ao mesmo tempo, ou seja, tudo depende obviamente das necessidades da instituição.

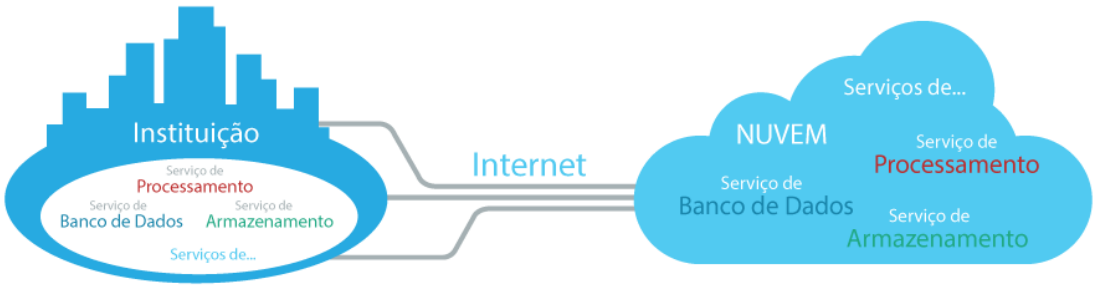


Figura 7 - Estrutura da computação em nuvem e a comunicação com instituições

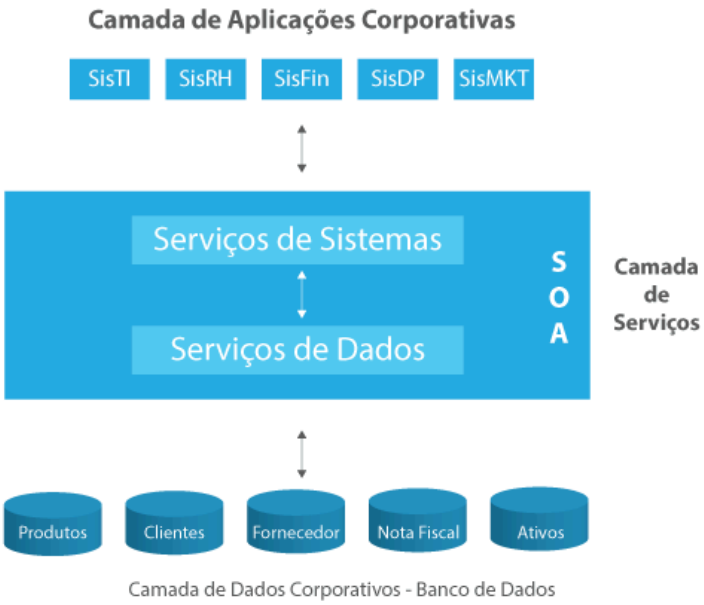
Fonte: GUIMARÃES, Leandro; CORRÊA, Matheus; OSÓRIO, Tomás. Computação em Nuvem.

Arquitetura Orientada a Serviços (*Service-oriented Architecture* - SOA)

A arquitetura orientada a serviços é um novo paradigma no desenvolvimento de soluções em software, pois apresenta uma proposta de interoperabilidade e acoplamento fraco de serviços. Estes serviços podem ser entendidos como um ou mais métodos disponíveis ou um processo grande em execução. Estes serviços são unidades básicas e fundamentais para o desenvolvimento de software. Além disso, utilizam padrões e protocolos de comunicação abertos, como, por exemplo, o *HiperText Transfer Protocol* (HTTP). Assim, os serviços viabilizam a comunicação entre os departamentos de uma empresa ou entre empresas diferentes que possuem algum tipo de parceria de negócios.

A arquitetura orientada a serviços deve possuir um barramento de serviços (*Enterprise Service Bus* - ESB) onde cada serviço deve ter uma interface de comunicação que fornece a assinatura. Esta assinatura deve conter os parâmetros de entrada e saída, tratamentos de erro e tipos de mensagens padronizadas. A Figura 7 demonstra a estrutura básica da arquitetura SOA contendo os sistemas de informação (SI), o barramento ou

camada de serviços (ESB) e a camada de bancos de dados (BD). Na camada de SI, temos um exemplo de sistemas para os departamentos de uma empresa (Marketing, Recursos Humanos, Financeiro, Departamento Pessoal etc.). Na camada ESB, temos os serviços de software (regras de negócio) e serviços de dados (objetos de banco de dados, conexões, acesso a dados etc.). Na camada de BD, temos dados de diversos assuntos importantes para os departamentos, tais como ativos, clientes, fornecedores, produtos etc.



Estrutura básica da Arquitetura Orientada a Serviços

Sistemas de Informação Distribuída

Os sistemas de informação distribuída são modelos computacionais que envolvem um conjunto de aplicações e bancos de dados que compõem os sistemas de informação empresarial. Desta forma, discutiremos a seguir dois modelos de sistemas de informação distribuída:

1. Sistemas de Processamento de Transações
2. Middleware de Aplicações Corporativas

Sistemas de Processamento de Transações

Este modelo é formado por computadores clientes que submetem suas requisições de transações a vários servidores ou nós diferentes na rede. Assim, uma transação cliente se torna distribuída quando envoca dois ou mais servidores na requisição. Conforme a Figura 9 a seguir, existem dois modelos transacionais, sendo o modelo de transação plana e aninhada.

- Transação Plana: Ocorre quando o computador cliente faz a solicitação da transação para dois ou mais servidores. Uma transação plana somente é concluída quando todas as suas requisições são atendidas. Portanto, cada transação acessa sequencialmente os objetos dos servidores.

- Transação Aninhada: Permite que cada transação requisitada pelo cliente poderá abrir uma subtransação. Consequentemente, a subtransação poderá abrir outra subtransação em uma camada mais profunda de aninhamento. Assim, o modelo permite que cada transação seja executada paralelamente, pois são executadas em servidores diferentes.

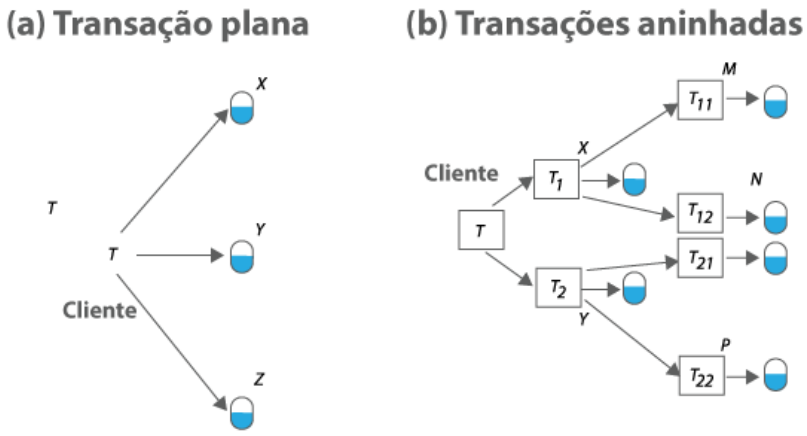
Estas transações distribuídas são invocadas por computadores clientes ou servidores através de Chamadas de Procedimentos Remotos ou *Remote Procedure Calls* (RPC). Existem algumas soluções ou produtos disponíveis para o desenvolvimento de transações distribuídas, são elas:

- DCOM (*Distributed Component Object Model*): O DCOM é uma solução da Microsoft para RPC para o desenvolvimento de aplicações distribuídas, mas foi substituída na plataforma .NET pela API (*Application Programming Interface*) .NET *Remoting.*
- CORBA (*Common Object Request Broker Architecture*): O CORBA é uma solução para o desenvolvimento de aplicações distribuídas criada pela OMG (*Object Management Group*). Assim, permite que um objetivo CORBA possa ser invocado por um outro computador. Este objeto remoto, pode invocar outro objeto em execução em outro computador. O CORBA é o RPC da OMG.
- RMI (*Remote Method Invocation*): O RMI é uma solução Java para o desenvolvimento de aplicações distribuídas e, como utiliza a *Java Virtual Machine (*JVM), oferece a portabilidade do código para outras plataformas de sistemas operacionais. O RMI é o RPC para Java.
- WS (*Web Services*): O WS é o RPC utilizado para que aplicações distribuídas possam se comunicar através da Internet. O WS utiliza o protocolo SOAP (*Simple Object Access Protocol*) que empacota as mensagens em XML (*eXtensible Markup Language*) para a comunicação. Assim, o modelo permite que outras aplicações desenvolvidas em outras linguagens possam consumir os recursos do WS, garantindo assim, maior portabilidade da solução.

As soluções acima podem ser implementadas tanto para transações planas ou aninhadas, ou seja, depende dos propósitos da aplicação distribuída. Além disso, as soluções oferecem maior portabilidade e interoperabilidade da aplicação - requisitos importantes para lidar com ambientes heterogêneos e inerentemente distribuídos.

Middleware de Aplicações Corporativas

Como vimos anteriormente, o *Middleware* é uma solução que causa a coesão ou integração do sistema. Assim, muitas aplicações antigas e desenvolvidas para um propósito único (conhecidas também por sistemas legados), como por exemplo, sistemas de faturamento ou de emissão de notas fiscais, não possuem interoperabilidade alguma e dificultam o acesso as informações por elas mantidas. O modelo de *Middleware* permite criar uma camada intermediária entre as aplicações distribuídas ou não, onde serviços podem ser executados viabilizando a coesão do sistema.



Transações Planas e Aninhadas

Fonte: COULOURIS, George et al. Sistemas Distribuídos-: Conceitos e Projeto. Bookman Editora, 2013

Sistemas Distribuídos Pervasivos

Os sistemas distribuídos pervasivos são modelos que envolvem dispositivos móveis e ubíquos e, que de maneira geral, estão associados ao avanço tecnológico para a redução de tamanho (miniaturização) e a conectividade com as redes sem fio.

Computação Móvel

A computação móvel surgiu a partir da década de 80 quando a produção de computadores pequenos, em larga escala, se tornou possível. Dessa forma, os computadores foram ficando menores ao ponto de poderem ser carregados (*notebooks)* e conectados em outros computadores. A conexão com outros dispositivos se dava a partir de um modem que utilizava as linhas telefônicas. Logo em seguida, surgiram as conexões *infrared* ou infravermelho. Com a chegada dos *laptops*, outros meios de comunicação se tornaram viáveis, tais como Wifi, Bluetooth, GPRS e 3/4G (com possibilidade de ancoragem).

Após esse processo evolutivo, surgiram os computadores de mão (*handheld computing*) denominados como *Personal Digital Assistants* (PDA) ou Assistentes Pessoais Digitais, telefones celulares e outros aprarelhos que cabiam no bolso e eram operados manualmente. Estes dispositivos evoluíram e ganharam sistemas operacionais inteligentes, tais como Linux, Android (que utiliza o kernel do Linux), Symbian, iOS e MS-Windows Phone. Além disso, estes dispositivos possui uma quantidade muito grande de aplicativos para jogos, escritório, conectividade, computação embarcada e sensores que medem distância percorrida e passos, navegação em GPS, além de aplicações bancárias que aumentam a distribuição do sistema.

Computação Ubíqua

A computação ubíqua representa "em toda parte", pois os sistemas computacionais evoluíram de tal maneira que os computadores passaram a fazer parte do cotidiano do homem em qualquer lugar. Estes dispositivos, muitos deles formados por PDAs, computadores, leitores de código de barras, TVs, dispositivos móveis etc, estão presentes no ambiente do usuário que consome estes recursos de conexão e informação. Um exemplo de computação ubíqua temos as residências, bibliotecas, lojas de livros, universidades, hipermercados e grandes lojas de varejo.

Sistemas Domésticos

Os sistemas domésticos são formados por computadores pessoais, TVs, telefones celulares, *tablets*, sistemas de áudio e consoles de videogame. Estes aparelhos compõem o modelo de computação ubíqua para ambientes domésticos. Algumas questões quanto a privacidade e a manutenção dos equipamentos são preocupações que devem ser consideradas, pois os usuários destes ambientes não possuem conhecimento suficiente para lidar com inúmeras possibilidades de ataque e atualizações de segurança necessárias. Neste caso, existem três alternativas plausíveis:

1. Contratação de um suporte especializado. Algumas seguradoras de olho no crescente mercado de computação doméstica, oferecem seguro residencial com suporte à informática.
2. Treinamento especializado. Como são ambientes relativamente novos para os usuários no Brasil, não existe muita oferta neste caso, mas vale a pena buscar alguma especialização no assunto.
3. Avanço da tecnologia: Aguardar ou buscar tecnologias mais recentes que fazem a autoconfiguração e a automanutenção de firmware e aplicações nestes dispositivos poderá ser um caminho.

Sistemas Eletrônicos Acoplados ao Corpo

Os sistemas eletrônicos acoplados ao corpo vêm ganhando espaço entre os usuários aficionados em tecnologia. Assim, os dispositivos presos ao corpo (ou dentro dele) tais como os relógios, pulseiras, óculos, são facilmente manipulados e trabalham com medições de passos ou corridas, acesso à Internet, batimento cardíaco, pressão arterial etc.

Neste sentido que um serviço vem ganhando o mercado do setor da saúde: O *Home Care* (Tratamento em Casa). Este serviço oferece ao paciente o tratamento e o monitoramento com sensores conectados ao corpo que são enviados para um computador local (contendo um banco de dados local ou não) e posteriormente enviados ao servidor do hospital, onde enfermeiros e médicos acompanham e analisam os dados coletados pelos sensores.

Além destes exemplos, as **Redes de Sensores** são sistemas que podem estar conectados ao homem e outros dispositivos contidos no ambiente. Por exemplo, imagine um sensor que identifique a temperatura corporal e, quando o usuário entra em sua casa, o ar condicionado é acionado com a temperatura ideal naquele momento e vai ajustando a temperatura com o

passar do tempo. Outros exemplos são os sensores que medem o leito dos rios e acionam um alerta em caso de nível crítico, e os sistemas de monitoramento de tráfego aéreo e terrestre.

Quiz

Exercício

Tipos de Sistemas Distribuídos

INICIAR

Referências

JUNIOR, Moacir A. Campos et al. Computação em Grade na Saúde: Proposta de Uma Arquitetura Para Interação de Informações Médicas Distribuídas.

TANENBAUM, A. S., STEEN, M. V. Sistemas Distribuídos: Princípios e Paradigmas, Pearson Prentice Hall, 2ª edição, 2007.

COULOURIS, George et al. **Sistemas Distribuídos: Conceitos e Projeto**. Bookman Editora, 2013.

XIONG, Jing; WANG, Jianliang; XU, Jianliang. Research of Distributed Parallel Information73 Retrieval Based on JPPF. In: 2010 International Conference of Information Science and Management Engineering. 2010. v.1, p. 109-111.

Avalie este tópico

ANTERIOR
Conceitos e Metas de Sistemas Distribuídos

Índice

PRÓXIMO
Modelos de Arquitetura

Ajuda?
(https://ava.un
idCurso=)

Biblioteca
(https://www.uninove.br/conheca-a-uninove/biblioteca/sobre-a-biblioteca/apresentacao/)
Portal Uninove
(http://www.uninove.br)
Mapa do Site