VOLTAR

Arquitetura de Sistemas Colaborativos

Apresentar as características básicas dos modelos de arquitetura que servem de base para implementar aplicativos colaborativos.

NESTE TÓPICO

A complexidade tecnológica enfrentada para o projeto e desenvolvimento de sistemas colaborativos depende da forma como os usuários precisam interagir entre si. As aplicações usadas para prover a colaboração entre usuários, permitem que a comunicação entre eles ocorra de modo assíncrono ou síncrono. No modo síncrono, um usuário envia um conteúdo para os outros usuários que usam a aplicação. Esse trabalho é transmitido imediatamente aos outros, que ao receber esse conteúdo, o visualizam, e todos eles, incluindo o usuário que enviou o conteúdo, visualizam exatamente o mesmo conteúdo, e isso sempre ocorre em tempo real. No modo assíncrono, um usuário envia um conteúdo aos demais usuários do aplicativo. Entretanto, cada um desses usuários pode receber esse conteúdo num momento diferente dos demais. Isso quer dizer que um determinado usuário pode estar visualizando um conteúdo dos demais usuários, uma vez que outros usuários já podem ter modificado esse conteúdo.

Esse problema que envolve a sincronia dos dados também ocorre com outros tipos de sistemas, como aqueles que são usados em automação comercial ou industrial. O fato é que vários tipos de sistemas possuem determinados componentes de hardware ou software que precisam receber informações e processá-las, gerando um certo resultado. Normalmente, esse processamento foi solicitado por outro componente do sistema, que depende desse resultado para poder terminar o que estava fazendo. Isso exige que a troca de dados entre os vários componentes do sistema seja realizado de modo bastante eficiente. Essa comunicação entre os vários componentes apresenta uma característica comum aos sistemas

distribuídos, que funcionam através da colaboração entre vários componentes de software para prover um determinado serviço aos usuários de uma aplicação. Essa colaboração entre os componentes de depende da forma como a comunicação entre eles ocorre, e das limitações que ela enfrenta.

Existem várias definições para sistemas distribuídos, sendo a mais comum a que define um *sistema distribuído* como sendo um ***conjunto de componentes de hardware e software localizados em computadores autônomos, que se comunicam através da troca de mensagens***. A maior parte dos sistemas de grande porte são implementados sob a forma de sistemas baseados em componentes distribuídos, com os quais os usuários interagem através de computadores pessoais ou dispositivos móveis. A distribuição dos componentes de software facilita o compartilhamento dos recursos existentes num sistema, principalmente quando esses recursos também são outros componentes, sejam eles de hardware ou software. Isso implica que eles também podem ter o acesso a eles controlado por outros componentes, uma vez que vários usuários podem tentar usar esses recursos ao mesmo tempo, gerando concorrência no uso desses recursos.

Isso significa que o sistema distribuído necessita ter pelo menos um componente que gerencie a concorrência no acesso aos demais componentes de hardware ou software. Esse gerenciador de acessos concorrentes pode adotar uma das abordagens:

- Controle de concorrência pessimista: o componente controla, serializa e sincroniza todas as operações sobre os recursos compartilhados;
- Controle de concorrência otimista: o componente que gerencia o uso dos recursos, ao invés de evitar problemas devidos à inconsistência, ele tenta

detectar que quando esses problemas estão prestes a ocorrer, e se propõe a evitar que esses problemas ocorram;

Assim, uma vez que o componente que gerencia o acesso aos recursos compartilhados esteja pronto, é necessário considerar outras características dos sistemas distribuídos:

- Transparência da distribuição dos componentes: independentemente da complexidade do sistema, e da forma como os componentes estejam disponíveis no sistema, sob o ponto de vista do usuário, seja ele um ser humano ou um sistema automatizado, é como se todos os componentes estão instalados e sendo executados num único local. Como a percepção do usuário é essa mesma, uma vez que ele só tem acesso à interface que ele usa para acessar o sistema, se houver uma redistribuição no uso desses componentes, seus usuários não perceberão;
- Extensibilidade: isso implica que seja possível adicionar novas funcionalidades ao sistema, aumentando o número de serviços que ele é capaz de executar, ou em outro caso, aumentar a capacidade de processamento de um sistema, através do aumento da disponibilidade de um determinado componente do sistema, mas sem duplicá-lo, ou interromper o uso de componentes;
- Escalabilidade: isso requer que o sistema seja capaz de atender o aumento de solicitações de processamento, e realizar esse processamento, sem aumentar a capacidade atual de processamento que o sistema possui, e sem reduzir a quantidade e/ou a qualidade dos serviços que já estão sendo usados;
- Tolerância a falhas: isso quer dizer que o sistema deve ser capaz de continuar executando os serviços que ele foi projetado para executar, mesmo quando falhas ocorrerem;
- Interoperabilidade: isso quer dizer que o sistema poderá conter componentes de hardware e/ou software que são fornecidos pelos mais variados fabricantes. Isso quer dizer que todos eles devem ser capazes de se intercomunicar, o que pode ser feito através de protocolos de comunicação ou de mecanismos específicos para esse fim;
- Portabilidade: isso significa que cada um dos componentes de software usados para um sistema distribuído pode ser transferido para outro sistema distribuído, e eles continuarão funcionando exatamente do mesmo modo;

Até aqui, acho que é possível verificar que um sistema distribuído costuma ser heterogêneo, uma vez que ele pode ser composto por diferentes plataformas de hardware ou software, e isso inclui as linguagens de programação. Entretanto, a heterogeneidade não pode ser obstáculo ao compartilhamento de recursos à comunicação entre os vários componentes. Os componentes de hardware e software podem se comunicar através de padrões de comunicação, como os arquivos de metadados baseados no padrão JSON e XML, ou em camadas de software, como o CORBA (Common Object Request Broker Architecture), ou baseados no uso de uma máquina virtual, como a JVM de Java que permite a execução de componentes desenvolvidos de acordo com o modelo EJB (Enterprise Java Beans).

Os desafios pela comunicação entre os componentes que formam um sistema distribuído residem principalmente no uso das redes de comunicação, que apesar de serem bastante confiáveis atualmente, isso não significa que elas estão livres de falhas. Sendo assim, os componentes devem ser capazes de lidar com atrasos na comunicação, falhas no envio e recepção de dados, perda no sincronismo de transferência de dados, etc... Outro desafio se refere à segurança no tráfego dessas informações, e no acesso às informações compartilhadas. Isso significa que os componentes devem garantir a confidencialidade das informações trocadas entre os vários usuários desses sistemas, e garantir também a autencidade de quem está conectado ao sistema.

Falando especificamente dos sistema colaborativos, alguns dos problemas encontrados nos sistemas distribuídos são potencializados, como por exemplo:

- O controle do uso de recursos compartilhados é muito mais complexo quando ocorre a colaboração entre vários usuários, uma vez que um grande número deles pode acessar simultaneamente um determinado recurso exatamente ao mesmo tempo;
- A colaboração entre os usuários deve ser complementada através de serviços que permitam a troca de mensagens instantâneas, conversas por áudio ou vídeo, entre seus usuários, e sempre em tempo real;
- Como pode ser observado nas comunidades existentes nas redes sociais, é possível que um sistema colaborativo possua grupos específicos de colaboração, o que pode exigir uso de políticas descentralizadas para gerenciar as informações relacionadas a esses grupos, de segurança e de compartilhamento de informações e recursos;
- O controle de uso do sistema é feito sobre os usuários, enquanto que nos demais tipos de sistemas distribuídos, o controle é exercido sobre os componentes que formam o sistema;

Todas essas informações implicam que o sistema colaborativo deve ser baseado sobre a arquitetura de um sistema distribuído, mas o projetista do software deve sempre considerar as peculiaridades específicas aos sistemas colaborativos. Assim, a partir de um modelo que indique o conceito da arquitetura que se pretende para o sistema, o projetista pode usar alguns modelos de arquitetura, e projetar a implementação dos componentes que formarão o sistema distribuído numa arquitetura que atenda aos requisitos para o aplicativo que se pretende desenvolver.

Arquitetura Centralizada

O modelo de arquitetura centralizada segue o mesmo tipo de processamento realizado na já conhecida arquitetura cliente-servidor, onde todo o processamento que o aplicativo precisa é realizado no servidor, enquanto que os clientes são responsáveis por somente servirem de mecanismo de interação com o usuário.

Os aplicativos web baseados somente em páginas JSP, PHP ou ASP são bons exemplos disso. Essas páginas ficam instaladas no servidor e são processadas por ele, entretanto, as páginas web que resultam desse

processamento são mostradas no browser, que é considerado o cliente.

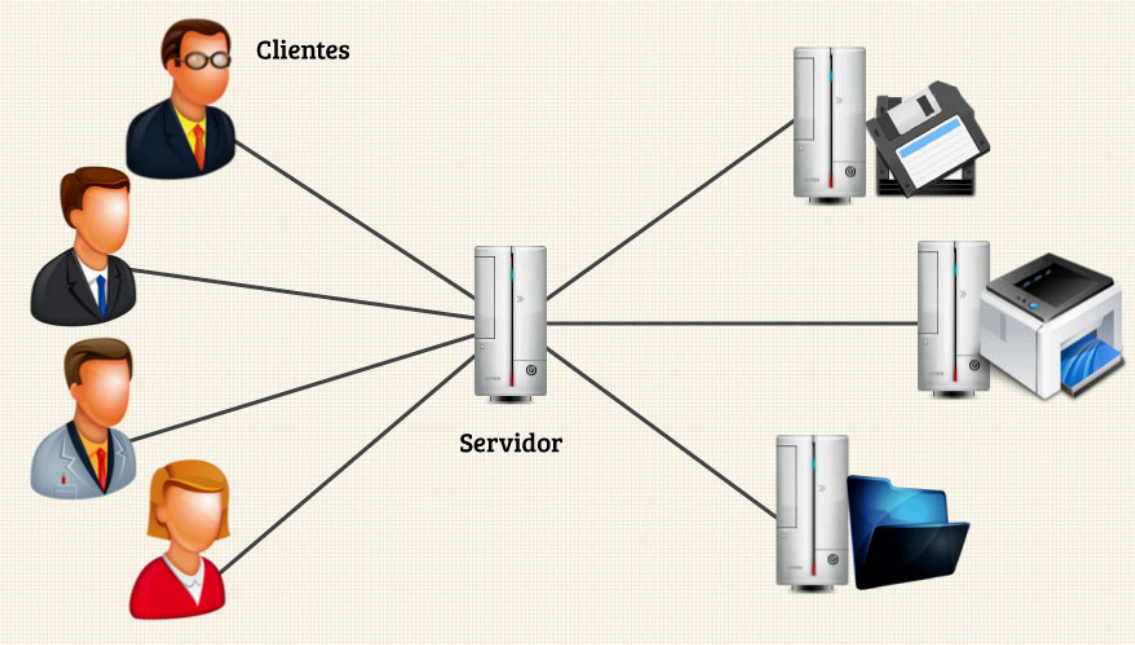


Figura 2 - Exemplo de centralização num servidor

Se por um lado, por ter todo o trabalho centralizado no servidor pode facilitar o desenvolvimento do aplicativo, e consequentemente do sistema, quando houver muitos usuários usando o aplicativo, o servidor pode demorar a enviar o resultado do processamento, causando atrasos no que o usuário recebe como resultado. Uma forma de diminuir esse efeito, é distribuir o processamento entre o cliente e o servidor, para diminuir a carga de processamento.

Além disso, ter um único componente responsável pelo processamento que é usado por todos os clientes, se por algum motivo, ele for desconectado da rede de comunicação, a aplicação ficará indisponível. Isso pode ser evitado se for adicionado ao sistema outra instância do servidor, um backup de serviços, que consiga executar os serviços do primeiro servidor, em caso de pane.

A centralização do processamento em um único servidor também facilita a gestão dos recursos compartilhados, evita boa parte dos problemas de concorrência no uso desses recursos e da segurança a ser aplicada no sistema no acesso e no compartilhamento das informações que ele armazena, além de facilitar a autenticação dos usuários que se conectam a esse sistema.

Apesar dessas vantagens, uma arquitetura centralizada tem as desvantagens de ter pouca escalabilidade, e como já visto antes, ter pouca tolerância a falhas, uma vez que todos os recursos estão disponíveis num único local, dificilmente será possível adicionar novos componentes de hardware e software para aumentar a capacidade de processamento, e em casos de falhas, provavelmente, elas serão notadas pelos usuários.

Arquiteturas Descentralizadas

Num aplicação implementada numa arquitetura descentralizada, o processamento executado pelo aplicativo é distribuídos pelos vários componentes do aplicativo. Isso significa que todos os componentes espalhados no sistema realizam algum tipo de processamento, sendo que esse processamento pode ser feito somente por esse componente, ou através da cooperação com outros componentes que existam no sistema.

O exemplo mais conhecido para esse tipo de arquitetura pode ser a própria Internet, onde vários usuários conseguem acessar vários sistemas on-line, seja para realizar compras, interagir com outros usuários, visualização de algum conteúdo específico, etc... Para que um usuário possa acessar um determinado site, ele precisa que o sistema saiba identificar o endereço físico (IP) do site a partir de um endereço digitado pelo usuário (a URL do site). Sem um servidor que faça a ligação entre a URL e o IP de um site (servidor de DNS), é muito provável que o usuário não consiga acessar o site através da URL, pois ele precisará saber o IP do site para acessá-lo diretamente.

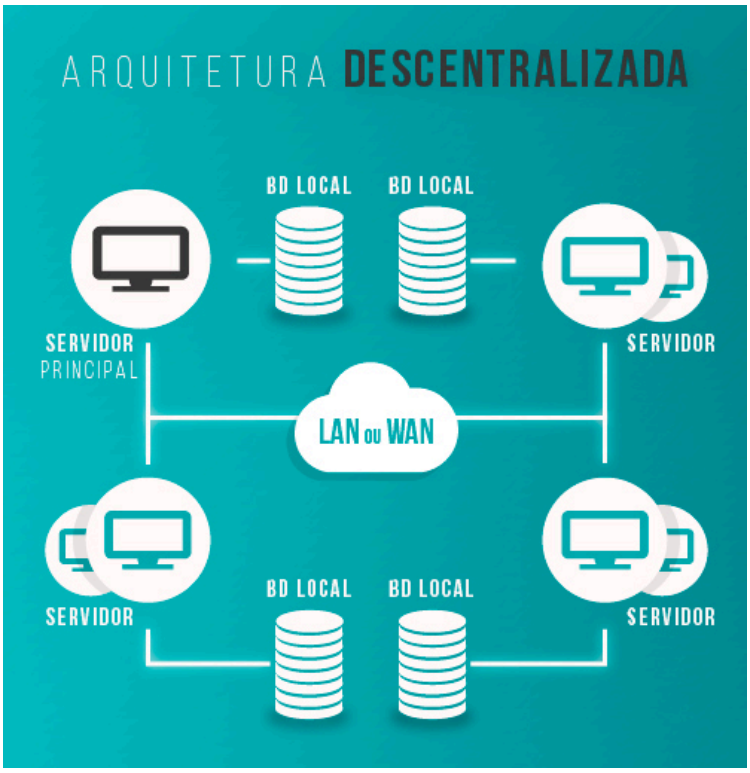


Figura 3 - Exemplo de arquitetura descentralizada

Este exemplo demonstra uma vantagem da arquitetura descentralizada. Como há vários componentes espalhados pelo sistema, ele possui maior tolerância a falhas, neste exemplo, se o servidor de DNS não estiver funcionando, isso não significa que o servidor que o usuário deseja acessar está indisponível. O usuário pode conseguir acessá-lo se souber o endereço IP.

Um sistema baseado em arquitetura descentralizada é mais difícil de ser controlado do que um sistema baseado em arquitetura centralizada, pois é mais difícil gerir o compartilhamento de recursos compartilhados, a

segurança que o sistema precisa ter, até mesmo, a localização dos vários componentes, e saber se os usuários que estão conectados ao sistema, e o usando, são realmente os clientes que deveriam estar usando esses sistemas. Além disso, a arquitetura descentralizada exige uso maciço da rede de comunicação, pois os componentes precisam trocar dados entre si com muito mais frequência do que numa arquitetura centralizada.

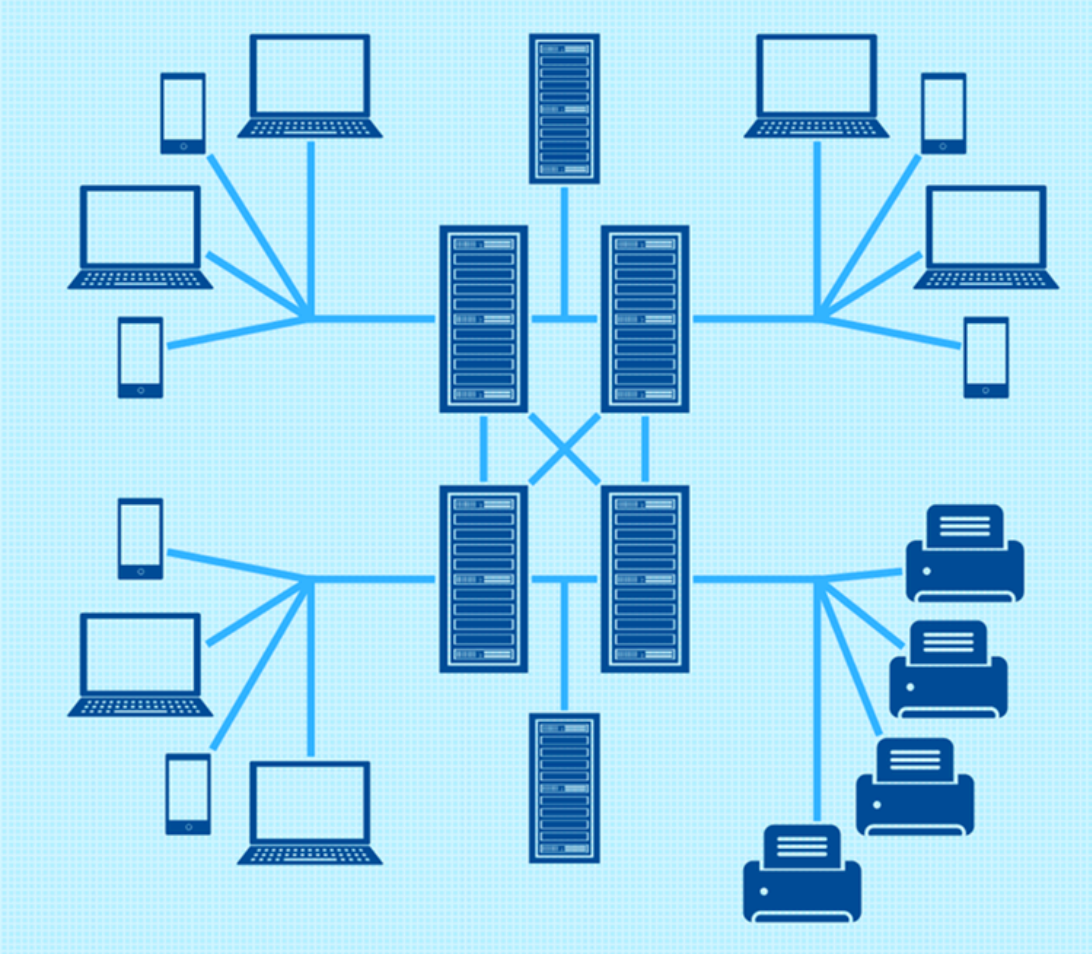
Esses aspectos demonstram que é necessário ter também um controle maior da consistência e da integridade dos dados, para que o aplicativo tenha certeza de que os dados sejam realmente os dados de que o usuário precisa, e que são os dados destinados àquele usuário.

Neste ponto, como há vários componentes espalhados pelos vários elementos que compõem o sistema, e a rede de dados é bastante usada, pode ser que ocorra um atraso na sincronização das informações, o que deve ser considerado pelo projetista do software, para evitar que esse atraso no sincronismo fique num nível em que a colaboração entre os vários componentes do sistema deixe de acontecer.

As vantagens da arquitetura descentralizada estão o fato de serem adicionados novos serviços ao sistema sem interferir nos serviços já existentes, ser mais fácil aumentar a capacidade de processamento do aplicativo, pela simples adição de novos componentes de software ou de hardware. Além disso, é possível reusar um componente do sistema em outro sistema, sem haver interferência nos demais componentes do sistema.

Arquiteturas Híbridas

Dependendo da aplicação a ser desenvolvida, o desenvolvedor pode optar por um modelo de arquitetura que reúna as vantagens dos modelos centralizados e descentralizados, na tentativa de eliminar ao máximo os aspectos negativos dos dois modelos.



Arquitetura Híbrida

Um modelo de arquitetura híbrida bastante interessante é o modelo que centraliza os componentes que realizam funções bastante específicas do aplicativo, em servidores únicos, e esses componentes podem ser acessados pelos diversos componentes espalhados na rede usada pelo aplicativo.

A Internet é quem usa esse modelo de arquitetura, e de modo bastante eficiente. Por exemplo, todo site de uma empresa fica localizado em um único servidor, que possui um único endereço IP. Para acessar esse servidor através de uma URL, o usuário deve fazer uso de um servidor DNS. Para evitar que um endereço IP não seja localizado caso um servidor DNS fique fora do ar, há vários outros servidores DNS na rede, que podem substituir aquele servidor DNS enquanto ele estiver indisponível, e os usuários não percebem o ocorrido.

Nessa arquitetura híbrida, um servidor que centraliza todo um processamento, pode ser implementado na forma de componentes distribuídos dentro de si.

Há várias possibilidades para implementar um sistema distribuído usando um modelo de arquitetura híbrida. A melhor forma de fazer essa implementação, depende do entendimento que o desenvolvedor tem do problema que ele está enfrentando, e de como ele acha mais conveniente implementar a solução que resolva esse problema.

Quiz

Exercício Final

Arquitetura de Sistemas Colaborativos

INICIAR

Referências

PIMENTEL, Mariano; FUCKS, Hugo. *Sistemas Colaborativos*. Editora Campus, 2011.

Avalie este tópico

ANTERIOR
Sistemas colaborativos

Índice

PRÓXIMO
Arquitetura de Software

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

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