Sistemas distribuídos

Aluno: Lucas de Lima Castro
Email: llc2@icomp.ufam.edu.br
MAT: 21551892

Resumo dos capítulos (1 e 2)

A definição de Sistemas distribuídos tem vários aspectos importantes. O primeiro é que um sistema distribuído consiste em componentes (isto é, computadores) autônomos. Um segundo aspecto é que os usuários, sejam pessoas ou programas, acham que estão tratando com um único sistema. Isso significa que, de um modo ou de outro, os componentes autônomos precisam colaborar. Como estabelecer essa colaboração é o cerne do desenvolvimento de sistemas distribuídos. Em princípio, até mesmo dentro de um único sistema, eles poderiam variar desde computadores centrais (mainframes) de alto desempenho até pequenos nós em redes de sensores. Da mesma maneira, nenhuma premissa é adotada quanto ao modo como os computadores são interconectados.

Uma característica importante é que as diferenças entre os vários computadores e o modo como eles se comunicam estão, em grande parte, ocultas aos usuários. O mesmo vale para a organização interna do sistema distribuído. Uma outra característica importante é que usuários e aplicações podem interagir com um sistema distribuído de maneira consistente e uniforme, independentemente de onde a interação ocorra.

Em princípio, também deveria ser relativamente fácil expandir ou aumentar a escala de sistemas distribuídos. Essa característica é uma consequência direta de ter computadores independentes, porém, ao mesmo tempo, de ocultar como esses computadores realmente fazem parte do sistema como um todo. Em geral, um sistema distribuído estará continuamente disponível, embora algumas partes possam estar temporariamente avariadas. Usuários e aplicações não devem perceber quais são as partes que estão sendo substituídas ou consertadas, ou quais são as novas partes adicionadas para atender a mais usuários ou aplicacöęs.

Para suportar computadorès e redes heterogêneos e, simultaneamente, oferecer uma visão de sistema ünico, os sistemas distribuídos costumam ser organizados por meio de uma camada de software -- que é situada logicamente entre uma camada de nível mais alto, composta de usuários e aplicações, e uma camada subjacente, que consiste em sistemas operacionais e facilidades básicas de comunicaçäol. Por isso, tal sistema distribuído às vezes é denominado middleware.

O fato de ser possível montar sistemas distribuídos não quer dizer necessariamente que essa seja uma boa ideia. Afinal, dada a tecnologia corrente, também é possível colocar quatro drives de disco flexível em um computador pessoal.

Um sistema distribuído deve oferecer fácil acesso aos seus recursos; deve ocultar razoavelmente bem o fato de que os recursos são distribuídos por uma rede; deve ser aberto e deve poder ser expandido.

A principal meta de um sistema distribuído é facilitar aos usuários, e às aplicações, o acesso a recursos remotos e seu compartilhamento de maneira controlada e eficiente. Os recursos podem ser muito abrangentes, mas entre os exemplos típicos estão impressoras, computadores, facilidades de armazenamento, dados, páginas Web e redes, só para citar alguns. Há muitas razões para querer compartilhar recursos, e uma razão óbvia é a economia.

Um outro problema de segurança é o rastreamento de comunicações para montar um perfil de preferências de um usuário específico (Wang et al.. 1998). Esse rastreamento é uma violação explícita da privacidade, em especial se for feito sem avisar o usuário. Um problema relacionado é que maior conectividade também pode resultar em comunicação indesejável, como envio de mala direta sem permissão, muitas vezes denominada spam. Nesses casos, talvez precisemos nos proteger usando filtros especiais de informações que selecionam mensagens de entrada com base em seu conteúdo.

Uma meta importante de um sistema distribuído é ocultar o fato de que seus processos e recursos estão fisicamente distribuídos por vários computadores. Um sistema distribuído que é capaz de se apresentar a usuários e aplicações como se fosse apenas um único sistema de computador é denominado transparente. Em primeiro lugar, vamos examinar quais são os tipos de transparência que existem em sistemas distribuídos. Em seguida, abordaremos a questão mais geral sobre a decisão de se a transparência é sempre requerida.

Transparência de acesso trata de ocultar diferenças em representação de dados e o modo como os recursos podem ser acessados por usuários, Em um nível básico, desejamos ocultar diferenças entre arquiteturas de máquinas, porém o mais importante é chegar a um acordo sobre como os dados devem ser representados por máquinas e sistemas operacionais diferentes.

Diferenças entre convenções de nomeação e também o modo como os arquivos devem ser manipulados devem ficar ocultos dos usuários e das aplicações. Um importante grupo de tipos de transparência tem a ver com a localização de um recurso. **Transparência de localização** refere-se ao fato de que os usuários não podem dizer qual é a localização física de um recurso no sistema. A nomeação desempenha um papel importante para conseguir transparência de localização.

Em particular, pode-se conseguir transparência de localização ao se atribuir somente nomes lógicos aos recursos, isto é, nomes nos quais a localização de um recurso não está secretamente codificada. Diz-se que sistemas distribuídos nos quais recursos podem ser movimentados sem afetar o modo como podem ser acessados proporcionam **transparência de migração**. Ainda mais vantajosa é a situação na qual recursos podem ser realocados enquanto estão sendo acessados sem que o usuário ou a aplicação percebam qualquer coisa. Nesses casos, diz-se que o sistema suporta **transparência de relocação**.

Transparência de replicação está relacionada a ocultar o fato de que existem várias cópias de um recurso. Para ocultar a replicação dos usuários, é necessário que todas as réplicas tenham o mesmo nome. Por consequência, um sistema que suporta transparência de replicação em geral também deve suportar transparência de localização porque, caso contrário, seria impossível referir-se a réplicas em diferentes localizações.

Já mencionamos que uma importante meta de sistemas distribuídos é permitir compartilhamento de recursos. Em muitos casos, esse compartilhamento é cooperativo, como no caso da comunicação. É importante que cada usuário não perceba que o outro está utilizando o mesmo recurso. Esse fenômeno é denominado **transparência de concorrência**.

Fazer com que um sistema distribuído seja **transparente à falha** significa que um usuário não percebe que um recurso (do qual possivelmente nunca ouviu falar) deixou de funcionar bem e que, subsequentemente, o sistema se recuperou da falha.

Uma outra meta importante de sistemas distribuídos é a abertura. Um sistema distribuído aberto é um sistema que oferece serviços de acordo com regras padronizadas que descrevem a sintaxe e a semântica desses serviços. No caso de sistemas distribuídos, em geral os serviços são especificados por meio de **interfaces**, que costumam ser descritas em uma **linguagem de definição de interface (Interface Definition Language — IDL).**

Interoperabilidade caracteriza até que ponto duas implementações de sistemas ou componentes de fornecedores diferentes devem coexistir e trabalhar em conjunto, com base na mera confiança mútua nos serviços de cada um, especificados por um padrão comum.

Portabilidade caracteriza até que ponto uma aplicação desenvolvida para um sistema distribuído A pode ser executada, sem modificação, em um sistema distribuído diferente B que implementa as mesmas interfaces que A.

Uma outra meta importante para um sistema distribuído aberto é que deve ser fácil configurá-lo com base em componentes diferentes (possivelmente de desenvolvedores diferentes). Além disso, deve ser fácil adicionar novos componentes ou substituir os existentes sem afetar os que continuam no mesmo lugar. Em outras palavras, um sistema distribuído aberto deve ser também **extensível**.

A realização efetiva de um sistema distribuído implica que especifiquemos e coloquemos componentes de software em máquinas reais. Para fazer isso, há diferentes opções. A especificação final de uma arquitetura de software é também denominada **arquitetura de sistema**.

Para nossa discussão, a noção de um **estilo arquitetônico** é importante, Tal estilo é formulado em termos de componentes, do modo como esses componentes estão conectados uns aos outros, dos dados trocados entre componentes e, por fim, da maneira como esses elementos são configurados em conjunto para formar um sistema. Um componente é uma unidade modular com interfaces requeridas e fornecidas bem definidas que é substituível dentro de seu ambiente (OMG, 2004b).

Apesar da falta de consenso sobre muitas questões de sistemas distribuídos, há uma delas com a qual muitos pesquisadores e praticantes concordam: pensar em termos de clientes que requisitam serviços de servidores nos ajuda a entender e gerenciar a complexidade de sistemas distribuídos, e isso é bom.

No modelo cliente-servidor básico, processos em um sistema distribuído são divididos em dois grupos, com possível sobreposição. Um servidor é um processo que implementa um serviço específico. Um cliente é um processo que requisita um serviço de um servidor enviando-lhe uma requisição e, na sequência, esperando pela resposta do servidor. Essa interação cliente-servidor, também conhecida como **comportamento de requisição-resposta**.

A distinção entre três níveis lógicos, como discutimos até aqui, sugere várias possibilidades para a distribuição física de uma aplicação cliente-servidor por várias máquinas. A organização mais simples é ter só dois tipos de máquinas: Uma máquina cliente que contém apenas os programas que implementam o nível (parte do nível) de interface de usuário. Uma máquina do servidor que contém o resto, ou seja, os programas que implementam o nível de processamento e de dados.

Arquiteturas cliente-servidor multidividida são uma consequência direta da divisão de aplicações em uma interface de usuário em componentes de processamento e em um nível de dados. As diferentes divisões correspondem diretamente à organização lógica das aplicações. Em muitos ambientes de negócios, o processamento distribuído equivale a organizar uma aplicação cliente-servidor como uma arquitetura multidivididas. Esse tipo de distribuição é denominado **distribuição vertical**.

Uma classe importante de sistemas distribuídos organizada segundo uma arquitetura híbrida é formada por **sistemas de servidor de borda**. Esses sistemas são disponibilizados na Internet onde servidores são colocados *na borda' da rede. Essa borda é formada pela fronteira entre as redes corporativas e a Internet propriamente dita.