Padrões de Arquitetura Distribuída

Henrique Ferreira de Paula[1], João Victor da Silva Alves[2]

{1,2}Faculdade de Computação – Universidade Federal de Uberlândia(UFU)
Av. João Naves de Ávila, 2.121 - Bairro Santa Mônica – CEP 38400-902 - Uberlândia/MG – Brazil

[1]henriquedci@gmail.com,[2]jvictor@reito.ufu.br

Abstract. *This article aims to introduce the reader to the world of distributed computing. For this is an introduction that shows concepts and definitions about the same and distributed systems, essential for understanding the development of this work. Then there are three architectural patterns for distributed computing (multi-processor architectures, client-server architectures and distributed object architectures), its key concepts and application examples and finally we will present the benefits of using each of these standards.*

Resumo. *Este artigo tem por objetivo introduzir o leitor ao universo da computação distribuída. Para isso, é feita uma introdução que mostra conceitos e definições sobre a mesma e os sistemas distribuídos, essenciais para a compreensão do desenvolvimento desse trabalho. Em seguida, são apresentados três padrões arquiteturais para computação distribuída, arquiteturas de múltiplos processadores, arquiteturas cliente-servidor e arquiteturas de objetos distribuídos,seus principais conceitos e exemplos de aplicação e por fim, serão apresentados os benefícios do uso de cada um desses padrões.*

1. Introdução

1.1 – Definições

1.1.1 – Computação Distribuída

Uma das definições mais simples de computação distribuída apresenta a mesma como uma forma de executar aplicações cooperantes em máquinas diferentes. Diversas outras definições também podem ser encontradas, como a de que ela é a adição do poder computacional de diversos computadores interligados por uma rede. A computação distribuída começou a dar seus primeiros passos com o advento das redes de computadores como intranet's, redes públicas ou privadas e a própria internet. Atualmente, os sistemas computacionais estão cada vez mais elaborados e complexos, e grande parte dos computadores do mundo está interligada por essas redes.

No entanto, esse tipo de computação apresenta uma série de dificuldades para desenvolver, gerenciar e manter o sistema. Além disso, o controle do acesso concorrente a dados e a recursos compartilhados, evitar que falhas de máquinas ou da rede comprometam o funcionamento do sistema e a garantia da segurança e do sigilo dos dados são também as principais dificuldades características da computação distribuída.

1.1.2 – Sistemas Distribuídos

O panorama descrito anteriormente, tem permitido cada vez mais o surgimento de sistemas distribuídos, que são a união de diversos computadores via rede com o objetivo de compartilhar a execução de tarefas. Isso aumenta consideravelmente o poder de processamento, comportando uma maior carga e maior número de usuários, além de proporcionar melhor tempo de resposta e maior confiabilidade. No entanto, os sistemas distribuídos apresentam algumas desvantagens, como o comportamento imprevisível, em virtude do uso da rede e a possibilidade de ocorrerem falhas nos computadores e na própria rede. Além disso, os recursos computacionais utilizados não são totalmente controlados pelo pelo sistema, uma vez que podem empregar recursos de terceiros. Por fim, é preciso citar também que os sistemas distribuídos são bastante influenciados pela velocidade das redes das quais dependem.

Para contornar esses problemas, os sistemas distribuídos são implementados seguindo diversos padrões de arquitetura, sendo que cada tipo de sistema se adequa melhor a um padrão, e pode ter suas dificuldades minimizadas se o padrão arquitetural for bem escolhido.

1.2 – Padrões de arquitetura

Como visto anteriormente, os sistemas distribuídos devem ser implementados seguindo diversos padrões de arquitetura, sendo que cada tipo de sistema distribuído pode exigir um padrão de arquitetura específico. Esses padrões são constituídos de um conjunto de politicas que fornecem a melhor maneira de se implementar um sistema distribuído. Nesse trabalho, apresentaremos três padrões de arquitetura mais comumente utilizados, que são: arquiteturas de múltiplos processadores, arquiteturas cliente-servidor e arquiteturas de objetos distribuídos .

2. Desenvolvimento

2.1 – Arquiteturas de Múltiplos Processadores

Nesse tipo de arquitetura, o sistema é composto por múltiplos processos que podem (não necessariamente) executar em processadores diferentes. Para isso, a execução de um processo pode ser predeterminada ou estar sob o controle de um escalonador. Assim, esse tipo de arquitetura é mais indicada quando se precisa de um processamento paralelo, ou seja, quando um mesmo programa é carregado por um sistema e roda simultaneamente em vários computadores. Para que isso sejá possível, o programa deve ser escrito especificamente para rodar nos vários computadores e além disso deve também possuir uma forma de comunicação entre esses esses computadores. Uma das formas mais comuns de se obter essa comunicação é o uso da biblioteca MPI – *Message Parsing Interface*, um conjunto de funções utilizado para comunicação entre computadores.

É interessante perceber, que o processamento paralelo exige a paralelização de uma atividade, para que cada parte possa ser executada por um computador independente, e todas as partes serem por fim unidas para se chegar ao resultado. Para isso, existem diversos métodos para paralelização, segundo vários critérios, como por exemplo a localização dos dados e do programa executável, a divisão do trabalho e a espera por mensagens.

Entre as vantagens dessa arquitetura, estão o ganho de velocidade de processamento que, dependendo do número de computadores, pode chegar a valores extremamente altos, da ordem de milhares de vezes o que se conseguiria com um único computador.

Atualmente, existem alguns projetos em que se é possivel emprestar o seu computador, quando estiver ocioso, para contribuir com o processamento para diversos trabalhos científicos, bastando apenas estar conectado a internet. Essa arquitetura possui algumas desvantagens, como a complexidade de se dividir o processamento e o gerenciamento de mensagens.

2.2 – Arquiteturas Cliente-Servidor

Talvez uma das mais comuns, a arquitetura cliente-servidor é modelada como um conjunto de serviços que são fornecidos pelos servidores e um conjunto de clientes que usam esses serviços. Portanto, uma definição para a arquitetura Cliente/Servidor seria a existência de uma plataforma base para que as aplicações, onde um ou mais Clientes e um ou mais Servidores, juntamente com o Sistema Operacional e o Sistema Operacional de Rede, executem um processamento distribuído .Um sistema Cliente/Servidor poderia ser, então, entendido como a interação entre Software e Hardware em diferentes níveis, implicando na composição de diferentes computadores e aplicações.

Nesse contexto, o cliente, também denominado de “front-end” e “WorkStation”, é um processo que interage com o usuário através de uma interface gráfica ou não, permitindo consultas ou comandos para recuperação de dados e análise e representando o meio pela qual os resultados são apresentados. Além disso, apresenta algumas características distintas como: é o processo ativo na relação Cliente/Servidor, inicia e termina as conversações com os Servidores, solicitando serviços distribuídos, não se comunica com outros clientes e torna a rede transparente ao usuário.

Adicionalmente, o servidor, também denominado “back-end”, fornece um determinado serviço que fica disponível para todo cliente que o necessita. A natureza e escopo do serviço são definidos pelo objetivo da aplicação Cliente/Servidor. Além disso, ele apresenta ainda algumas propriedades distintas, como: é o processo reativo na relação Cliente/Servidor; Possui uma execução contínua; Recebe e responde às solicitações dos Clientes; Não se comunica com outros servidores enquanto estiver fazendo o papel de Servidor; Presta serviços distribuídos e atende a diversos Clientes simultaneamente.

Por fim, essa arquitetura apresenta várias vantagens, como a Confiabilidade, já que se apenas um dos servidores apresentar problema, grande parte do sistema ainda continua ativa. Além disso, a arquitetura cliente-servidor permite que as tarefas sejam feitas sem a monopolização de recursos, assim, usuários finais podem trabalhar localmente. Ainda, pode-se encontrar dentro de uma arquitetura Cliente/Servidor a interoperabilidade das estações Clientes e Servidoras entre as redes de computadores, a escalabilidade da arquitetura visando o crescimento e a redução dos elementos constituintes, a adaptabilidade de novas tecnologias desenvolvidas, a performance do hardware envolvido na arquitetura, a portabilidade entre as diversas estações que compõem a arquitetura e a segurança dos dados e processos.

No entanto, algumas desvantagens podem ser observadas nesse tipo de arquitetura, como a extensa lista de itens a serem investigados quando ocorre algum

erro, a falta de ferramentas de suporte e treinamento, aumento da complexidade , tudo isso, faz com que o gerenciamento de um sistema cliente-servidor extremamente difícil.

2.1 – Arquiteturas de Objetos Distribuídos

Nas arquiteturas de Objetos Distribuídos, não existem distinções entre clientes e servidores. Assim, cada entidade é um objeto que fornece e recebe serviços de outros objetos. Para permitir isso, os objetos se comunicam por meio de um sistema de middleware (mediação) . Dessa forma, podemos definir o middleware como sendo o software que gerencia e apoia os componentes de um sistema distribuído, ou seja, ele atua literalmente no meio do sistema como um mediador. Existem diversos padrões internacionais existentes para gerenciar a comunicação entre objetos distribuídos. O mais conhecido e usado atualmente é o CORBA (*Common Object Request Broker Architecture*), um padrão de arquitetura desenvolvido pelo OMG (*Object Management Group*), uma organização internacional que aprova padrões abertos para aplicações orientadas a objetos. Juntamente com o DCOM da Microsoft, CORBA é um dos padrões de arquitetura para computação distribuída mais populares.A CORBA defini, em sua estrutura, um objeto intermediário que atua como módulo responsável por aceitar a requisição do cliente, enviá-la ao objeto competente e quando a resposta estiver disponível, entregá-la ao cliente. Ainda, é utilizada pelo padrão uma linguagem baseada em C++, declarativa, que possibilita a interoperabilidade entre os diversos sistemas.A maior vantagem da CORBA, é a sua independência de plataforma e de linguagem, além da sua licença. As maiores desvantagens do ambiente CORBA é que somente uns poucos serviços especificados foram implementados.. Por exemplo, o software Visibroker 3.0 tem implementado somente os serviços de resolução de nomes e eventos. Serviços importantes do CORBA tais como segurança, Consulta, controle de concorrência e transações ainda não estão disponíveis, apesar de serem essenciais.

3. Conclusão

Não é apenas o poder de processamento dos computadores que deve ser considerado quando pensamos em construir software para o futuro. Deve-se considerar também que os sistemas a partir de agora estarão necessariamente funcionando em rede. A evolução da largura de banda das redes é ainda mais impressionante que a evolução do poder de processamento dos computadores. Isto significa que em pouco tempo mesmo grandes programas poderão estar distribuídos pela internet e serem executados como se estivessem instalados localmente no computador do usuário. Assim, segundo esse contexto, o conhecimento dos diversos padrões de arquitetura distribuída é essencial para todos os profissionais envolvidos no desenvolvimento de sistemas de informação.

References

Fabiane Bizinella Nardon,Sérgio Furuie, Umberto Tachinardi,“Novas Tecnologias para Construção do Prontuário Eletrônico do Paciente”, http://www.tridedalo.com.br/fabiane/publications/NovasTecnologiasPEP.pdf

Ana Paula Gonçalves Serra, “O modelo de arquitetura CORBA e suas aplicações”,

ftp://ftp.usjt.br/pub/revint/157_37.pdf

Jair Cavalcanti Leite, “Arquiteturas de Sistemas Distribuídos”, http://www.dimap.ufrn.br/~jair/ES/slides/ArqSistDist.pdf