Implementando Aplicações Distribuídas Utilizando CORBA: Um Estudo de Caso Voltado à Área de Banco de Dados

ANAMÉLIA CONTENTE DE SOUZA
VITÓRIO BRUNO MAZZOLA

Centro de Ensino Superior do Pará - CESUPA
Área de Ciências Exatas e Tecnologia
Av. Governador José Malcher, 1963
66060-230 – Belém – PA
(91)246-7172/ (91)2465494
E-mail: anamelia@inf.ufsc.br
cesupa@cesupa.br

INE – UFSC – Campus Universitário
88040-900 – Florianópolis – SC
Tel.: (048) 331.9498 / (048) 331.9566
E-mail: mazzola@inf.ufsc.br

Resumo: o presente artigo trata do estudo de plataformas para o desenvolvimento de aplicações distribuídas, em particular a arquitetura CORBA. O objetivo perseguido neste estudo é o de apresentar esta plataforma através de seus principais componentes, seu funcionamento e principais vantagens e problemas. Além de um estudo teórico, baseado na documentação disponível sobre CORBA, é descrita também a experiência de implementação conduzida sobre uma aplicação na área de Bancos de Dados.

Palavras-chave: CORBA, DCOM, Bancos de Dados Distribuídos.

1 Introdução

O desenvolvimento de software segundo a tecnologia de Objetos tem se revelado uma abordagem com inúmeros benefícios. Dentre estes, pode-se destacar a modularidade, flexibilidade e reusabilidade, características essenciais para a qualidade e a produtividade do desenvolvimento de software.

Considerando a diversidade de plataformas computacionais e de linguagens de programação disponíveis atualmente, a concepção de uma aplicação deve levar em conta a possibilidade de utilização de componentes (ou Objetos) concebidos nas diferentes linguagens e capazes de executar em diferentes ambientes.

Visando proporcionar um ambiente de desenvolvimento capaz de manipular a heterogeneidade e a distribuição das aplicações atuais, que o OMG (*Object Management Group*) definiu a arquitetura CORBA (*Common Object Request Broker Architecture*), uma arquitetura de suporte ao desenvolvimento voltada para aplicações distribuídas heterogêneas orientadas a Objetos. Desde o início dos anos 90, quando foram publicados os primeiros documentos relativos à arquitetura CORBA, um grande número de trabalhos foi iniciado no sentido de aprimorar a arquitetura e de apresentar implementações compatíveis com sua especificação.

Mais recentemente, a Microsoft apresentou a especificação de um ambiente com a mesma orientação do CORBA, a arquitetura DCOM, que tem sido vista como uma proposta concorrente àquela proposta pelo OMG.

O presente artigo apresenta os principais conceitos relativos à arquitetura CORBA e os resultados obtidos a partir de sua utilização no desenvolvimento de uma aplicação na área de Bancos de Dados.

Na seção 2, é introduzida a arquitetura CORBA e os principais elementos que a compõem. Na seção 3, é descrita a experiência de implementação

envolvendo o uso de CORBA e, finalmente, na seção 4, as conclusões sobre os resultados obtidos nesta experiência.

2 A Arquitetura CORBA

CORBA é uma arquitetura que foi definida pelo OMG e tem sua base no modelo Cliente/Servidor e no paradigma de Orientação a Objetos. Segundo este modelo, a execução de uma aplicação distribuída segue um esquema de interação entre Clientes e Servidores, de modo que um Cliente encaminha solicitações de serviços a Objetos CORBA.

As solicitações de serviços, assim como as respostas a estes são gerenciadas por um ambiente de comunicação, o ORB (*Object Request Broker*). Para cada Objeto de uma aplicação, existe a especificação dos serviços que ele pode oferecer. O Cliente, de posse desta informação, é capaz de encaminhar sua solicitação ao Objeto que poderá atendê-la.

Objetos CORBA podem assumir o papel de Clientes de outros Objetos CORBA. Isto é feito quando, para que ele possa executar um de seus serviços (que tenha sido anteriormente solicitado por um Cliente), deva fazer uso de um ou diversos serviços providos por outros Objetos.

O Cliente pode obter informações dos serviços do Objeto, nas interfaces dos mesmos. Estas interfaces tanto podem fornecer estas informações ao Cliente, quanto à infra-estrutura de comunicação. O formato das mensagens recebidas e enviadas pelos Objetos, não fica visível, de forma que seja assegurada a transparência na comunicação entre o Cliente e o Objeto. Estes Objetos recebem um identificador único, para diferenciá-los um do outro, independente do local onde se encontrem, chamada "Referência do Objeto".

2.1 Object Management Architecture

A **OMA**, foi criada pelo OMG, com o objetivo de integrar as aplicações baseadas em Objetos distribuídos. Sua estrutura é baseada essencialmente em dois elementos: o ***Modelo de Objeto***, que define o Objeto que será distribuído pelo sistema heterogêneo e o ***Modelo de Referência***, que define as características da integração destes Objetos.

O RFP (*Request for Proposal*) é utilizado para que os Modelos de Objetos e de Referência, possam ser compatíveis com especificações anteriormente utilizadas e, com isso tornar possível a construção de sistemas de objetos distribuídos interoperáveis em sistemas heterogêneos.

CORBA possui toda sua estrutura baseada na arquitetura OMA. Esta define a comunicação entre os Objetos distribuídos através de quatro elementos:

- **Objetos de Aplicação**: São os Objetos criados pelo usuário, que possuem características definidas de acordo com os seus objetivos. Eles também são tidos como os usuários finais de toda a infraestrutura CORBA.
- **Facilidades Comuns**: São componentes definidos em IDL (*Interface Definition Language*) que fornecem serviços para o uso direto das aplicações de Objetos. Estes estão divididos em duas categorias que são, horizontal e vertical. As Facilidades Comuns definem regras de integração para que os componentes possam colaborar efetivamente.
- **Serviços Comuns**: São serviços que possibilitam a implementação e utilização de Objetos. Eles definem uma estrutura de Objetos de baixo nível, que ampliam as funcionalidades do ORB, sendo utilizados para criar um componente, nomeá-lo e introduzi-lo no ambiente.
- **ORB (Object Broker Request)**: É o elemento que permite que Objetos emitam solicitações em um ambiente distribuído e heterogêneo, de forma transparente. Os principais aspectos de operação deste elemento serão descritos a seguir.

2.2 Operação do ORB

Clientes e Objetos que se comunicam no CORBA, podem ser definidos em diferentes linguagens de programação, ordenação de bytes interna da máquina, localização e outros aspectos que compõem um sistema heterogêneo distribuído. Portanto, é necessário que a interface com a qual o Cliente tem contato direto, e esta com o Objeto, seja independente de tais fatores. O ORB cumpre este papel.

O ORB, além de garantir, a transparência dos fatores acima citados, é também responsável pela ativação de um Objeto, para o qual uma solicitação é encaminhada. Esta ativação é realizada de forma transparente, não importando se o Objeto está situado a nível local ou remotamente.

Como já foi mencionado, um Cliente precisa informar a que Objeto se destina sua solicitação. Isto é feito através da **Referência do Objeto**. Existem diversas maneiras de um cliente obter as referências sobre outros objetos do sistema. Uma delas é no momento da criação do objeto. No CORBA, este

processo ocorre como se fosse uma requisição como outra qualquer, feita pelo cliente, ao qual retorna a referência a objeto.

Isto faz com que a estrutura do ORB permaneça bem simples, que é uma das características interessantes do CORBA, se preocupando somente com a comunicação e a infra-estrutura de ativação para as aplicações de Objetos Distribuídos, deixando para os demais componentes do OMA, o máximo de funcionalidade possível.

2.3 *Stubs* e *Skeletons*

Os *stubs* e os *skeletons* realizam funções correspondentes, sendo que o *stub* fica situado na extremidade do Cliente e o *skeleton*, na extremidade do Objeto. São eles que implementam a arquitetura Cliente/Servidor no CORBA.

Tanto o *stub* Quanto o *skeleton* são criados baseados na compilação da interface IDL do Cliente e do Objeto, respectivamente. Quando um Cliente emite uma solicitação, ele só precisa dizer ao *stub* para qual Objeto ela é destinada, não devendo importar se o Objeto está situado a nível local ou não. Isto é o que garante a transparência nos sistemas distribuídos. O *stub* localizará o objeto destino e fará junto com o ORB o empacotamento da mensagem, de forma que seja possível enviá-la pela rede (se necessário).

Devido a esta capacidade de localizar Objetos-destinos, os *stubs* também são denominados de *proxies*.

Os *skeletons* têm função parecida, com a diferença que eles irão receber solicitações do ORB. A partir de então, deverão desempacotar a mensagem e entregá-la à implementação do Objeto. Caso haja resposta, eles farão o processo inverso.

Uma vez que as mensagens são construídas na aplicação Cliente e na implementação do Objeto, *stubs* e *skeletons*, precisam ter conhecimento completo das interfaces OMG IDL dos Objetos CORBA que são invocados. Isto é o que se chama de **Invocação Estática**.

Nesta modalidade de invocação, o *stub* deve transformar da linguagem de programação que é passada da aplicação do Cliente, para uma forma que possa ser trafegada pela rede. E o *skeleton* faz o inverso. Ele transforma o que recebe para a linguagem de programação da implementação do Objeto.

A invocação e envio dinâmico surgiu para solucionar o problema da inflexibilidade da invocação estática, que não permitia que Objetos criados após a compilação do *stub* ou *skeleton*, fossem acessados. Na invocação estática, para que novos Objetos pudessem ser acessados, era necessário parar o sistema ou realizar nova compilação. Isto já não é mais necessário na invocação dinâmica.

As duas interfaces suportadas pelo CORBA para a invocação dinâmica são o DII e o DSI. O DII (Interface de Invocação Dinâmica), corresponde ao *stub* e o DSI (Interface *Skeleton* Dinâmica), ao *skeleton*. Aqui, estas interfaces, diferentemente da invocação estática, são fornecidas diretamente pelo ORB, portanto não há necessidade de cada Cliente ou Objeto possuir a sua. A vantagem da invocação dinâmica, é a transparência, pois a implementação nunca sabe se a sua requisição está sendo feito a um *stub* estático ou dinâmico.

3 Utilizando CORBA

Como forma de avaliar a utilização da plataforma CORBA e de seus componentes, foi desenvolvida uma aplicação de banco de dados distribuídos. Como ferramenta de implementação, foi adotado o ambiente Delphi 4, na versão Cliente/Servidor.

O exemplo escolhido corresponde à base de dados de um hotel, o **PitAnita's Hotel**. O programa utilizado para escrever e armazenar as tabelas da base de dados, foi o *Database Desktop*.

3.1 Servidor CORBA

Tanto o servidor quanto o cliente foram implementados utilizando-se o ambiente orientado a objetos Delphi 4. Esta versão deste ambiente, já traz ferramentas que facilitam a interação entre clientes e servidores CORBA.

O primeiro que deve ser criado naturalmente é o servidor. Ele conterá os componentes que terão acesso às tabelas. No caso desta aplicação, foram criados dois servidores. Um deles contendo as tabelas de Lotação e Ocupação, e o outro, as de Reserva e Cadastro de Hóspedes. Na verdade, bastaria que fosse criado um único servidor contendo todas estas tabelas. Contudo, com o intuito de prevenir falhas em caso de haver algum problema com um destes servidores, e também para a demonstração do uso de mais de um servidor por um mesmo cliente simultaneamente, concluiu-se que seria mais conveniente a repartição das tabelas em mais de um servidor.

Os dois servidores criados foram o PitAnitaCorbaServ1 e o PitAnitaCorbaServ2. Ambos contém um Módulo de Dados Remoto CORBA, com duas tabelas específicas cada.

O próximo passo é acrescentar ao módulo tantos componentes *TTable* quanto forem o número de tabelas requeridas, no caso duas, conectar este componente ao BD e tabela requeridos, ativá-lo e exportá-lo de forma que sua tabela possa estar visível aos usuários.

3.2 Cliente CORBA

A aplicação cliente é conectada ao servidor através de um componente MIDAS do Delphi, denominado

CorbaConnection.

Este possui algumas propriedades que deverão ser preenchidas, tal como o nome do computador em que o servidor está localizado; o identificador do repositório, que é o nome do servidor mais o da interface do objeto que foi adicionada ao repositório de interfaces; e o nome do objeto, que é o nome da instância do objeto. No caso de uma aplicação cliente/servidor local, não é necessário fornecer nem o nome do objeto e nem o do *host*.

Para executar uma aplicação CORBA desenvolvida no Delphi, é necessário primeiramente abrir o ORB na máquina servidora, que no caso é o SMART AGENTda VisiBroker que já vem com esta versão do Delphi. Posteriormente, o servidor deverá ser executado. Quando o servidor já estiver rodando, então é a vez do cliente. Basta executá-lo no próprio Delphi.

No caso da nossa aplicação, dois ORBs deverão ser abertos, um em cada máquina servidora, depois os dois servidores deverão ser executados ou através do DOS ou do Explorando do Windows, e somente então os clientes.

4 Resultados

Para que se pudesse obter resultados significativos, a análise foi realizada em dois níveis: uma análise de cunho teórico, com base na documentação disponível; uma análise experimental, envolvendo uma implementação que utilizou como exemplo de reflexão uma aplicação na área de bancos de dados.

Embora não tenha sido descrito neste artigo, os estudos realizados consideraram também a plataforma DCOM da Microsoft, no contexto de um estudo comparativo.

No primeiro estudo, pôde-se observar, uma maior abertura do CORBA para o desenvolvimento de aplicações, uma vez que existem implementações desta plataforma disponíveis para as diversas arquiteturas e sistemas operacionais do mercado; já o DCOM, embora tenha as mesmas orientações, impõe uma arquitetura onde o servidor (ou servidores) do sistema devam ser baseados em Windows NT, o que, de certo modo, não foge à estratégia da Microsoft em fazer do seu sistema operacional a plataforma líder no desenvolvimento de sistemas corporativos.

No segundo estudo, algumas dificuldades foram, por outro lado, encontradas, desde a seleção da implementação até a obtenção de sucesso na operação da aplicação selecionada nos ambientes CORBA e DCOM. Conforme relatado na sessão 3, dentre as diversas opções possíveis em termos de ambiente de desenvolvimento, foi escolhido o ambiente Delphi (versão 4.0), o que foi determinado pela disponibilidade de uma versão que apresentava suporte ao desenvolvimento de aplicações em CORBA e DCOM. Para realizar a análise, foi escolhido um exemplo simples de aplicação de base de dados, considerando basicamente dois perfis de implementação.

O primeiro perfil de implementação considerava todos os processos da aplicação localizados no mesmo posto de trabalho, esta decisão tendo sido tomada como estratégia de ajuste da aplicação no ambiente de desenvolvimento. O segundo perfil de implementação foi projetado considerando os processos distribuídos ao longo de uma subrede dentro do ambiente computacional da rede do Departamento de Informática e de Estatística da Universidade Federal de Santa Catarina.

Para a obtenção de resultados desta segunda etapa de análise, os critérios considerados foram a facilidade de implementação, a adequação ao perfil pretendido da aplicação, a portabilidade e a taxa de sucesso na operação da aplicação.

No que diz respeito à facilidade de utilização, observou-se que este critério está mais vinculado ao ambiente de desenvolvimento adotado do que propriamente da proposta de arquitetura considerada. No caso deste estudo, o desenvolvimento da aplicação foi facilitado pela existência de bibliotecas previamente construídas disponíveis no ambiente Delphi 4, o que tornou a implementação possível de ser realizada num prazo relativamente curto.

No tocante à adequação ao perfil da aplicação escolhida, no caso, uma aplicação de bancos de dados, também foi observado um nível de adequação bastante satisfatório no caso das duas plataformas, não tendo sido observadas diferenças significativas. É importante ressaltar, ainda, que o ambiente de desenvolvimento escolhido oferece facilidades importantes para o desenvolvimento de aplicação que fazem uso do modelo Cliente/Servidor, caso do exemplo escolhido no contexto deste trabalho.

No aspecto portabilidade, sem dúvida o desenvolvimento da aplicação sobre CORBA apresentou melhores características, uma vez que ela oferece uma maior flexibilidade, principalmente no que diz respeito à localização do servidor ou servidores, o que não ocorre no caso de aplicações envolvendo o DCOM, que impõe o posicionamento de servidores em máquinas sob Windows NT, com o MTS instalado.

Finalmente, no aspecto sucesso operacional da aplicação, o DCOM, embora tenha permitido implementar rapidamente a aplicação com o perfil local (todos os processos localizados na mesma máquina), ofereceu dificuldades para o funcionamento no perfil distribuído.

Finalizando esta parte de apresentação de resultados, é importante ressaltar o interesse cada vez maior no desenvolvimento de aplicações distribuídas com base em plataformas voltadas a aplicações distribuídas orientadas a objetos, como CORBA e DCOM. As vantagens da adoção de tal tecnologia são muitas, podendo-se resumir como principal benefício a concentração dos esforços do desenvolvedor sobre as funcionalidades e requisitos da aplicação, podendo este abstrair-se dos problemas que podem surgir da distribuição dos processos e das eventuais conseqüências advindas da heterogeneidade do sistema computacional, uma vez que estas podem ser resolvidas pela adoção de uma tal plataforma.

5 Bibliografia

[1] BERTINI, Luciano. **Apresentação de Objetos Multimídia e Hipermídia em um Ambiente Distribuído Heterogêneo Utilizando os Padrões MHEG-5 e CORBA**. Florianópolis, 1998.

[2] ORFALI, Robert; HARKEY, Dan. **Client/Server Programming with Java and Corba.** 2.ed. Canada: John Wiley & Sons Inc., 1998.

[3] CANTÚ, Marco. **Dominando o Delphi 4: "A Bíblia".** São Paulo: Makron Books, 1998.

[4] SOUZA, Anamélia. **Tecnologia para Ambientes Distribuídos – CORBA.** Monografia. Belém, 1998.

[5] TALLMAN, Owen; KAIN, J. Bradford. **COM versus CORBA: A Decision Framework.** [on line]. Disponível na Internet via WWW: URL: http://www.quoininc.com/quoininc/ COM_ CORBA.html

[6] **CORBA VS. DCOM.** [on line]. Disponível na Internet via WWW: URL: http://www.intraware.com/ms/itwr/snws/ 980605.html