ESPECIFICAÇÃO E IMPLEMENTAÇÃO DO PROTOCOLO DE ACESSO A BASE DE DADOS REMOTA

ANA MARIA MOTTA GARCIA LOPES MANUEL DE JESUS MENDES

DCA - FEE - UNICAMP

CAIXA POSTAL 6101 - FAX: (0192) 391395

CEP: 13081 - CAMPINAS - SP - BRASIL

e-mail: anamaria@leblon.fee.unicamp.ansp.br

RESUMO

Neste artigo são introduzidos conceitos sobre acesso a bases de dados remotas. São descritos os conceitos básicos e o modelo funcional do protocolo, que inclui um conjunto de serviços genéricos que atende, no mínimo, as propriedades comuns de uma classe de serviços específicos. Uma ferramenta CASE (Computer Aided Software Engineering) conhecida como EPOS é usada na implementação do protocolo visando integrá-lo ao projeto SISDI-MAP.

ABSTRACT

This paper introduces the concepts about remote database access. The basic concepts and functional model of protocol are described. This includes a subset of generic services that attends, at least, the common properties of a class of specific services.

A CASE tool, known as EPOS, is used for implementing the protocol for integrate it to the SISDI-MAP.

1)INTRODUÇÃO

Com a crescente necessidade de compartilhamento de recursos, a cooperação entre unidades computacionais distintas obteve um considerável acréscimo de importância. A informação já não poderia ficar restrita a um único equipamento, ganhando mobilidade para satisfazer a um acesso solicitado, independente de características de software ou hardware. A comunicação entre estes sistemas heterogêneos é baseada em padrões estabelecidos no modelo de referência para interconexão de sistemas abertos (OSI - Open System Interconnection) da entidade ISO (International Standards Organization). O modelo OSI/ISO consta de 7 camadas de funcionalidades distintas [04], dispostas hierarquicamente de modo a prover comunicação nó a nó. Nosso trabalho é uma implementação do protocolo de alto nível localizado na camada 7, camada de aplicação, denominado **RDA** (Remote Database Access) conforme padrão ISO 9579 [01].

Na camada de aplicação, identificamos dois tipos de elemento de serviço: os comuns (CASE's) e os específicos (SASE's). Como CASE's podemos citar: ACSE (Association Control Service Element), ROSE (Remote Operations Service Element), RTSE (Reliable Transfer Service Element) e CCR (Commitment, Concurrency and Recovery). Além do RDA, podemos citar como SASE's: FTAM (File Transfer, Access and Management), DS (Directory Services), NM (Network Management) e TP (Transaction Processing).

O protocolo RDA tem como finalidade facilitar o acesso a base de dados localizadas em estações de trabalho remotas. O padrão define elementos que serão mapeados nos CASE's que formarem o seu contexto de aplicação. Para tal, é usada a notação de operação remota (ro-notation) do protocolo ROSE, definido no documento ISO 9072-1 [03]. O padrão provê independência entre os sistemas componentes, tal que uma estação possa ser capaz de interagir com qualquer base de dados que suporte o padrão. Analogamente, uma base de dados que suporte o padrão será capaz de interagir com um usuário de qualquer estação que suporte o padrão.

Em nossa implementação buscamos nos aproximar de um desenvolvimento orientado a objeto [08], devido à aplicação permitir uma certa independência dos serviços de comunicação OSI. Os objetos serão os ASE's, que contém o encapsulamento de seus atributos e serviços exclusivos. Um serviço de um objeto é definido por sua função, com parâmetros associados. As mensagens são as interfaces entre objetos.

O padrão RDA será parte integrante do projeto SISDI-MAP (Sistema Didático para MAP) que vem sendo desenvolvido no Departamento de Engenharia da Computação e Automação da FEE-UNICAMP, com o objetivo de simular a comunicação entre dois nós ligados em rede num ambiente industrial [06]. Na seção seguinte apresentaremos o protocolo dentro de seu modelo e funcionalidade. Na seção 3, será feita uma breve descrição de seus serviços genéricos. Na seção 4, discutiremos aspectos da especificação da

implementação do RDA utilizando a ferramenta EPOS.Finalmente, na seção 5, apresentaremos algumas conclusões.

2)MODELO E FUNCIONALIDADE

No modelamento do RDA destacam-se dois componentes: **cliente e servidor** [01]. Por cliente entende-se a entidade de aplicação local que requisita o início da associação e os serviços da base de dados. Ao servidor, que é a entidade de aplicação remota, cabe a tarefa de responder a requisição do estabelecimento da associação e prover os serviços solicitados da base de dados. O nó servidor consta de uma base de dados com suas características e um sistema gerenciador , que irá prover os recursos de dados solicitados pelo nó cliente (requisitante). Na figura 1 temos o modelo RDA no ambiente OSI. A comunicação entre o AP cliente e o RDA será realizada por uma instanciação do programa de aplicação que traduzirá as chamadas de manipulação de dados em primitivas RDA. Do lado remoto, o RDA receberá as primitivas dos CASE's e decodificará para o gerenciador da base de dados as solicitações recebidas. A comunicação entre o RDA e os CASE's componentes do contexto de aplicação, se fará via primitivas com a passagem de parâmetros a serem mapeados nas PDU's dos CASE's.

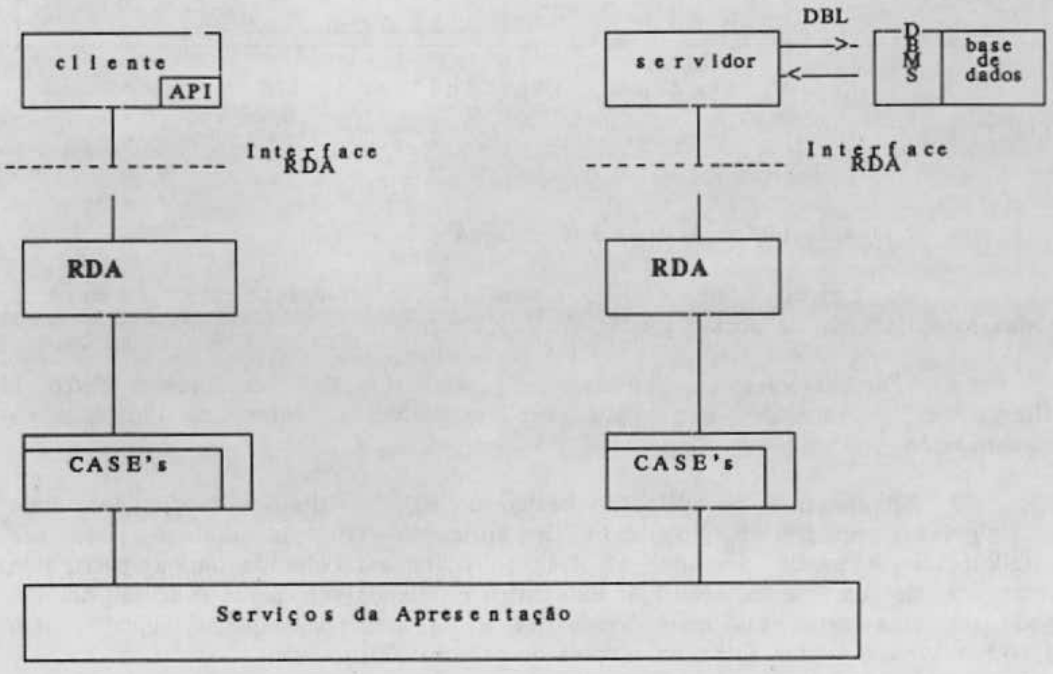

Figura 1 - Modelo do RDA

O protocolo RDA define serviços genéricos, com características comuns a serviços específicos diversos. Já as classes de serviços específicos serão definidas por especializações [02], que introduzem semânticas extra ou restringem a faixa de possibilidades do uso dos serviços genéricos. O próprio RDA pode ser considerado como uma especialização dos serviços de operações remotas, pois restringe as operações permitidas (o modelo de operações remotas define 4 classes de operações; o RDA só utiliza uma: assíncrona, retornando sucesso ou falha), além de introduzir sua própria semântica. Como exemplo de uma especialização temos a linguagem SQL que é mostrada com o uso do comando R-ExecuteDBL na figura 2.

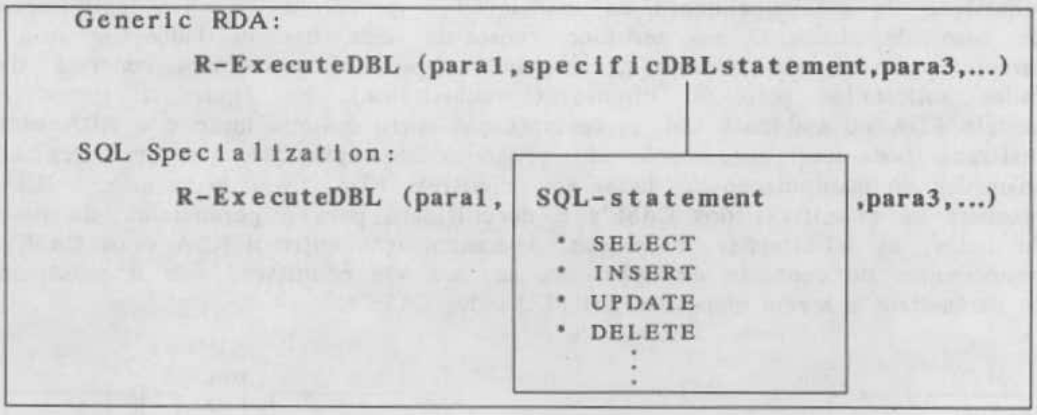

Figura 2 - Especialização SQL

2.1) Associação, Diálogo e Transação

Os conceitos de diálogo, associação e transação são inerentes a todas as aplicações de acesso a base de dados remotas.

Por associação, entende-se a relação contínua entre o cliente e o servidor que deve ser estabelecida antes de iniciada a comunicação.

No contexto de aplicação básico do RDA, o diálogo confunde-se com a própria associação. No contexto de aplicação TP, o diálogo pode ser estabelecido a partir de uma associação recém estabelecida ou a partir da retomada de uma associação já existente e disponível para uso. O diálogo pode ter uma existência mais longa que a da associação que o suporta, isto é, sobrevivendo a uma falha na própria associação [09].

A transação é uma unidade logicamente completa de processamento

que se torna atômica aos processos não envolvidos. A qualquer tempo, ao menos uma transação estará sendo processada dentro de um diálogo, ainda que do lado do servidor várias transações, pertencentes a diálogos distintos, estarão sendo processadas concorrentemente, com a garantia de serialização. Dependendo do contexto de aplicação utilizado, o gerenciamento de transações é realizado pelos serviços RDA (contexto de aplicação básico) ou pelos serviços TP (contexto de aplicação de TP).

Na figura 3 pode-se comparar os 2 tipos de contexto de aplicação RDA.

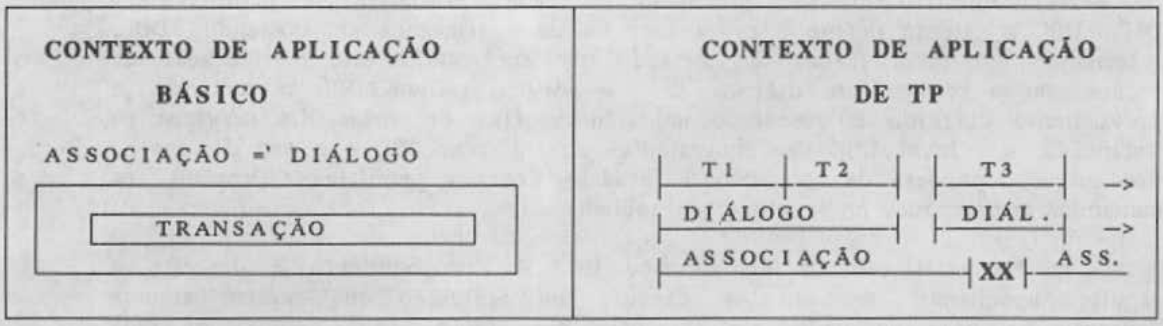

Figura 3 - Transações, Diálogos e Associações

3)DESCRIÇÃO DOS SERVIÇOS GENÉRICOS

No protocolo RDA os serviços genéricos são descritos formalmente como macros ASN.1. Os serviços RDA são agrupados em unidades funcionais de acordo com suas funções. Assim, temos:

3.1)Unidade Funcional de gerenciamento do diálogo

Onde estão agrupados os serviços responsáveis pelo estabelecimento (r-Associate) e pela liberação da associação (r-Release), assim como a retomada do diálogo. Uma primitiva r-Associate é enviada e retorna trazendo como resultado, entre outros, o identificador do diálogo, que será usado como argumento na retomada de um diálogo suspenso. Através da primitiva r-Release é negociado, entre os componentes, o tipo da especialização (rDAspecialization) que o gerenciador entenderá como a linguagem de manipulação da base de dados. Os serviços do ACSE [05], A-Abort e A-P-Abort, que provocam o término abrupto da associação tanto pelo usuário como pelo provedor, também estão contidos nesta unidade funcional.

3.2)Unidade Funcional de tratamento de recursos

Neste grupo encontram-se os serviços que controlam a disponibilidade dos recursos de dados cujos conteúdos serão acessados e manipulados pelas operações realizadas na base de dados do servidor. Uma

primitiva r-Open, que associa um recurso de dado para acesso, é enviada e pode retornar um identificador de recurso dado que permitirá a estruturação de forma hierárquica dos recursos associados. A primitiva r-Close libera um ou mais recursos que haviam sido previamente requeridos. Um recurso de dado é o meio pelo qual o cliente tem acesso a base de dados do servidor. Assim, o serviço r-Open deve preceder qualquer operação na base de dados.

3.3)Unidade Funcional de Operações na Base de Dados

Neste grupo são especificadas as operações que definem e/ou executam comandos da linguagem da base de dados (DBL). Na primitiva r-Execute um comando DBL é definido e executado imediatamente um número fixo de vezes, sendo o cliente informado de cada resultado. Na primitiva r-DefineDBL o cliente define e o servidor valida e armazena um comando DBL, retornando um identificador do comando que será válido até a liberação do recurso ou o término do diálogo. Com o serviço r-InvokeDBL o comando já previamente definido é executado um número fixo de vezes. Os serviços r-DefineDBL e r-InvokeDBL são equivalentes ao r-ExecuteDBL e o uso de um ou dos outros dependerá da aplicação do usuário. Com a primitiva r-DropDBL, os comandos armazenados no servidor são anulados.

Na utilização de um serviço RDA, o procedimento é descrito a seguir. Suponhamos que um AP execute um aplicação que realize alguma operação na base de dados do servidor. Para tanto, deve estabelecer uma associação através do envio da primitiva r-Associate.req. Com a confirmação desta, deve ser enviada a primitiva r-Open.req que o provedor do RDA mapeará num serviço RO-INVOKE.req para o ROSE, que retornará sucesso (RESULT) ou falha (ERROR). O mapeamento se dá através do preenchimento dos parâmetros do RO-INVOKE. Aí então o cliente envia uma primitiva que trate de uma operação na base de dados do servidor, cujo processo é semelhante ao descrito acima.

4)ASPECTOS DA ESPECIFICAÇÃO DA IMPLEMENTAÇÃO

Na especificação de um protocolo , existem alguns pontos em aberto que devem ser tratados antes da implementação propriamente dita : os "local implementation matters ". No protocolo RDA a implementação das interfaces, a definição dos valores das constantes, o tratamento das prioridades dos serviços são alguns dos problemas que tivemos que resolver. Com a adoção do EPOS, que já foi utilizado na implementação da camada de sessão, da camada de apresentação e do ROSE [07], os problemas locais, quando não solucionados com o uso da própria linguagem de especificação formal, são solucionados com a codificação direta em C, no programa fonte gerado pelo EPOS.

4.1) A Ferramenta EPOS

O pacote EPOS("Engineering and Project-management Oriented Support System) [10] foi desenvolvido pelo Instituto de Engenharia de Controle e Automação de Processos da Universidade de Stuttgart, Alemanha, para dar suporte a projetistas e engenheiros no planejamento, desenvolvimento e

manutenção de sistemas de hardware e de software. Para gerar o projeto, o EPOS dispõe de módulos e ferramentas que permitem criar uma Base de Dados, que é gerenciada em todo o ciclo de vida do projeto. O acompanhamento é possível através de análises entre requisitos e especificações [13] com documentação atualizada.

4.2) Especificação dos Requisitos no EPOS-R

Na especificação dos requisitos em EPOS-R, que se trata de uma linguagem de especificação para prover formalidade entre os diversos grupos de profissionais que atuem no projeto, foi considerada a descrição do problema e sua solução conceitual [11].

Na descrição do problema, descrevemos a estrutura do problema, o ambiente em se encontra, as ferramentas utilizadas no desenvolvimento, as interfaces com os CASE's e com o AP.

Na solução conceitual foram descritos os requisitos dependentes da solução, os componentes lógicos da solução e as restrições. Foram utilizados elementos funcionais da linguagem que permitissem conexões lógicas entre EPOS-R e EPOS-S (REQUIREMENT, CONSTRAINT e FULFILS).

4.3) Especificação da Implementação em EPOS-S

EPOS-S é uma linguagem de especificação formal, com sintaxe e semântica próprias, usada para descrever o projeto [12].

A especificação em EPOS-S é formada por um grupo de objetos estruturados de forma hierárquica com interações bem definidas entre si. A especificação é iniciada de forma mais abstrata e à medida que avançamos na especificação, chegamos ao nível de folha, que representa o código da linguagem. Os objetos de projeto utilizados foram : Action, Data, Condition, e Device.

Uma ACTION descreve atividades nas quais os dados são processados. Pode ser de quatro tipos: Module, Task, Procedure e Macro. Nossa estrutura resultante da especificação no EPOS-S (figura 4) inicia-se com ACTION MODULE SASE_RDA, que é o nó raiz da árvore hierárquica. A seguir, temos ACTION MODULE RDA_PROTOCOL, que constitui-se de uma ACTION TASK PROC_RDA que consiste de um loop eterno de verificação das portas que comunicam o RDA a um AP e aos CASE's associados: ACSE e ROSE. O outro nó ligado diretamente ao nó raiz é o RDA_INCLUDES, composto pelos arquivos externos que serão incluídos para a geração do código final. No uso de uma Action Procedure, descrevemos uma subrotina com função específica, como por exemplo o preenchimento dos parâmetros de uma primitiva. Uma Action Macro representa uma especificação que é substituída pelo nome da action macro no corpo da especificação. A única Action Macro do nosso projeto foi a de inicialização, que tratou das declarações de tipos e inicialização da tabela de estados.

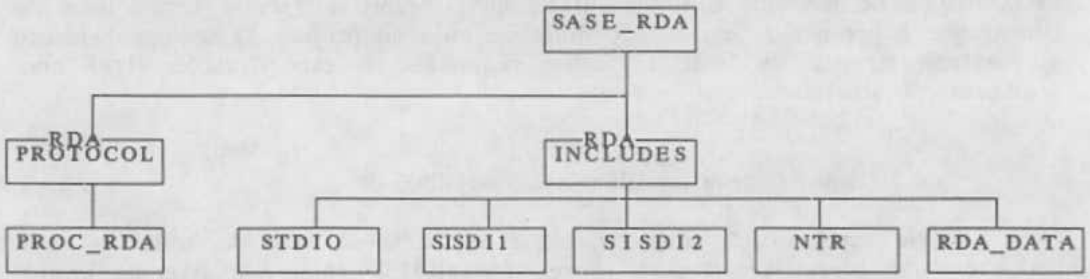

Figura 4 - Diagrama Hierárquico Principal da Especificação EPOS-S

No objeto DATA descrevemos as variáveis pertencentes ao projeto, como, por exemplo, as variáveis que definem os dados internos, as variáveis globais e as variáveis que definem as portas de comunicação. No CONDITION descrevemos condições que influenciam o fluxo do sistema; são variáveis lógicas. No objeto DEVICE, definimos o dispositivo utilizado, no nosso caso o microcomputador.

O EPOS permite sete filosofias de implementação, que privilegiam o objeto escolhido. Decidimos por um projeto misto, que não privilegiaria um objeto em relação a outro. A escolha se deu devido a características intrínsecas da implementação do protocolo.

Após a especificação completa EPOS-S, utilizamos uma ferramenta do EPOS, o COMPOSER que, a partir da especificação EPOS-S, executa a geração automática de código, em linguagens como ADA, PASCAL, FORTRAN e C. Nesta fase, podemos alterar o programa fonte gerado automaticamente pelo Composer e retroalimentar o banco de dados com as alterações efetuadas.

No decorrer do projeto, foi utilizada ferramenta EPOS-A [13] para testar as inconsistências descritas pelo EPOS-R e EPOS-S, gerando relatórios de análise.

Em nosso projeto do RDA, optamos pelo uso da linguagem C devido a sua portabilidade e a fácil migração para o UNIX das workstations, onde o projeto SISDI-MAP vêm sendo implantado.

5)CONCLUSÕES

Com o crescente uso das especializações, o RDA permite que ambientes heterogêneos suportem uma considerável quantidade de linguagens de manipulação de base de dados, apenas com uma negociação prévia.

No futuro, uma extensão do RDA deverá permitir a inclusão de objetos complexos de banco de dados multimeios (voz, imagem, vídeo, etc) com

a atuação das redes de alto desempenho com taxas acima de 100 Mbits/s.

Em nosso trabalho foi considerado o contexto de aplicação básico do RDA, até porque o elemento da aplicação TP não estava implementado. Um contexto de aplicação que use o TP, provê uma infraestrutura na qual aplicações distribuídas em ambientes heterogêneos possam processar transações confiáveis entre um ou mais sistemas abertos.

AGRADECIMENTOS

Agradecemos ao Serviço de Processamento de Dados do Estado de Ceará - SEPROCE, pela liberação da autora para cursar mestrado na FEE-UNICAMP, onde o trabalho vem sendo desenvolvido.

REFERÊNCIAS BIBLIOGRÁFICAS

[01] ISO DP 9579 - Information Processing Systems - Remote Database Access - nov.,1988.

[02] ISO/IEC 9579-2 - Information Processing Systems - Open Systems Interconnection - Remote Database Access - Part 2: SQL Specialization - fev.,1990.

[03] ISO 9072-1 - Information Processing Systems - Text Communication - Remote Operation - Model, Notation and Service Definition - 1988.

[04] ISO DIS 7498 - Information Processing Systems - Open Systems Interconnection - Basic Reference Model - 1984.

[05] ISO 8649 - Information Processing Systems - Open Systems Interconnection - Service Definition for the Association Control Service Element - dez.1988.

[06] Mendes, M. J. - Comunicação Fabril e o Projeto MAP/TOP - IV EBAE - jan.,1989.

[07] Silva, C.R.M. - Implementação do Elemento de Serviço para Operações Remotas Utilizando uma Ferramenta de CASE conhecida por EPOS - Tese de Mestrado - FEE/UNICAMP - mar.,1991.

[08] Coad, P., Yourdon, E. - Object-Oriented Analysis - Prentice-Hall - 1990.

[09] Johannsen, W., Lamersdorf W. - An Open System Architecture for Transaction Supported Distributed Database Applications - IBM Deutschland European Networking Center - 1990.

[10] Lauber, R.J., Lemp, P.R. - Epos Overview - 1986.

[11] GPP - Specific. Languge EPOS-R - EPOS Manual - ver 5.0.

[12] GPP - Specific. Languge EPOS-S - EPOS Manual - ver 5.0.

[13] GPP - Analysis System EPOS-A - EPOS Manual - ver 5.0.