PLATAFORMA MULTIWARE: PROJETO E DESENVOLVIMENTO DA CAMADA MIDDLEWARE

Mendes, M.J. - Unicamp-FEE-DCA
Madeira, E.R.M. - Unicamp-IMECC-DCC
e-mail: mendes@dca.fee.unicamp.br
edmundo@dcc.unicamp.br

Sumário

As plataformas hoje disponíveis estão sendo expandidas para suporte ao processamento aberto distribuído de informação multimídia. A plataforma Multiware em desenvolvimento na UNICAMP adota o modelo básico RM-ODP ("Reference Model - Open Distributed Processing") da ISO/CCITT ("International Organization for Standardization / Consultative Committee on International Telephony and Telegraphy") como modelo funcional e incorpora os conceitos e produtos já existentes no mercado: ANSAware, DCE/DME-OSF ("Distributed Computing Environment / Distributed Management Environment - Open Software Foundation") e CORBA-OMG ("Common Object Request Broker Architecture - Object Management Group"), entre outros. Descreve-se, no trabalho, o modelo de implementação do software correspondente à camada Middleware, com a adoção de metodologia iterativa incremental orientada a objetos.

1. INTRODUÇÃO

Com a evolução das tecnologias de comunicação de redes privadas de Alto Desempenho e das redes públicas B-ISDN ("Broadband - Integrated Service Digital Network"), baseadas em fibra ótica e em protocolos "light-weight" de transporte, abriu-se, na corrente década, a possibilidade de expansão do processamento e comunicação de informação em rede para novas áreas e campos de aplicação como Teleensino, Telemedicina, Telepublicação, Telecontrole e Telediagnose, entre outros.

Por um lado é fundamental o suporte ao processamento e comunicação multimídia (dados, texto, voz, imagens, vídeo, televisão,...). Localmente as estações de trabalho devem possuir hardware e software adequados ao processamento com parâmetros contratuados de qualidade de serviço, por exemplo, observando-se as restrições temporais típicas da representação sincronizada dessa informação. As Bases de Informação devem ser apropriadas à consulta e armazenamento eficientes de documentos complexos multimídia/hipermídia e os protocolos de comunicação, em particular de transporte e de aplicação, devem observar a nova realidade de capacidades de transmissão de Gbits/segundo, e as redes baseadas nas técnicas ATM ("Asynchronous Transfer Mode").

Por outro lado, as novas aplicações apresentam requisitos crescentes de cooperação em grupo, com a garantia de autonomia dos agentes isolados, e a realização de ambientes abertos, onde qualquer um possa entrar, de forma dinâmica, ou sair, sem prejuízo da operação. Entre outros citam-se os ambientes CSCW ("Computer Support Cooperative Work")(Groupware) como "Computer Conferencing" e DAI ("Decentralized Artificial Intelligence") [1] em Organizações Inteligentes, como representativos de uma parcela significativa do mercado futuro. Mas outras aplicações no ambiente de INs ("Intelligent Networks") e "Telecommunications Network Management" (TMN) deverão ser de igual importância e volume nos mercados público e privado da teleinformática.

Um dos aspetos mais importantes de desenvolvimento diz respeito à adoção de técnicas Orientadas a Objetos como forma mais adequada de domínio da complexidade dos sistemas, de garantia de reusabilidade dos sistemas em desenvolvimento ou já desenvolvidos, de implementação das bases heterogêneas e distribuídas da informação (OODBMS - "Object Oriented Data Base Management System") e da incorporação de ambientes de desenvolvimento (OOA - "Object Oriented Analysis", OOD - "Object Oriented Design", OOP - "Object Oriented Programming") que viabilizem processos iterativos e incrementais de engenharia de sistemas.

O trabalho enquadra-se num projeto em desenvolvimento na UNICAMP, que visa esta área tecnológica. A arquitetura básica das plataformas (Figura 1) agrega idéias defendidas na literatura e procura adotar produtos já existentes no mercado. A especificação funcional básica segue os princípios do modelo RM-ODP ("Reference Model - Open Distributed Processing") da ISO/CCITT. As várias camadas consideradas, - hardware/software básicos, middleware, groupware -, formam uma arquitetura aqui denominada de MULTIWARE.

Os temas mais importantes em desenvolvimento no projeto são:

- Especificação dos requisitos de aplicações CSCW em ambientes abertos e desenvolvimento de software de suporte a estas classes de aplicações;
- Adoção de sistemas e especificações existentes no mercado como ANSAware,

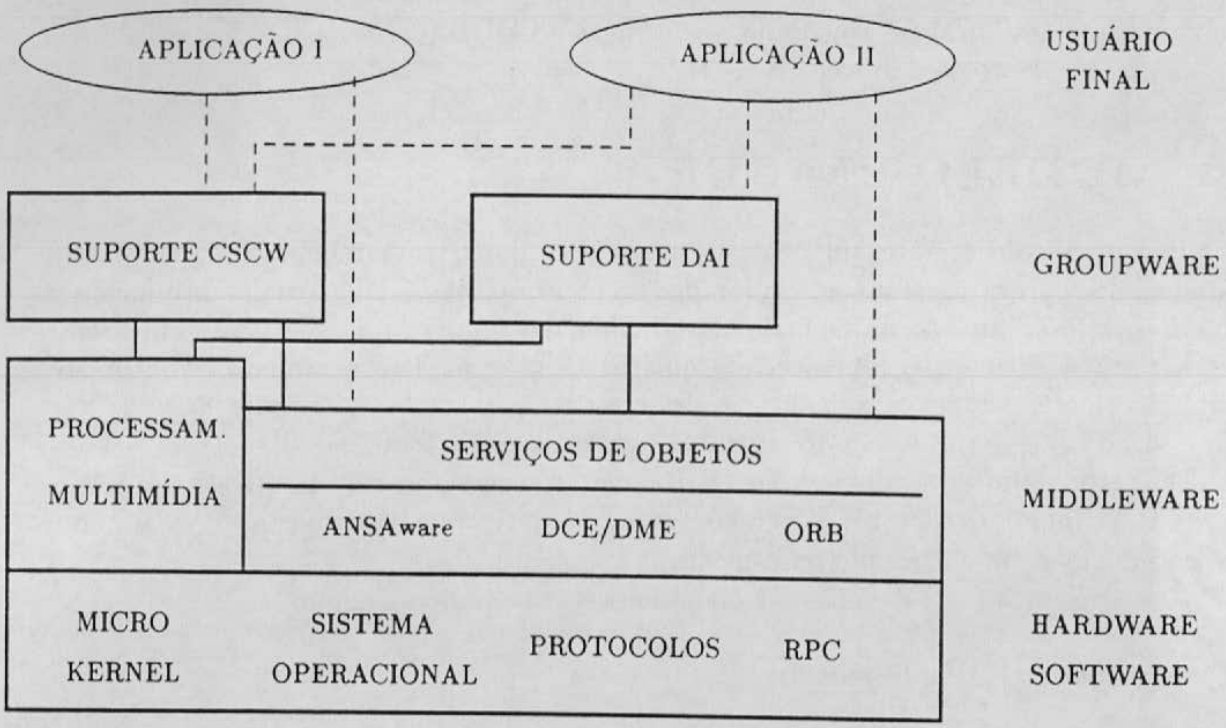

Figura 1

Figura 1: Plataforma MULTIWARE

DCE/DME -OSF ("Distributed Computing Environment / Distributed Management Environment - Open Software Foundation"), CORBA-OMG ("Common Object Request Broker Architecture - Object Management Group"),...;

- Desenvolvimento de funções da camada Middleware segundo o modelo RM-ODP (segurança, transparências, gerenciamento,...);
- Desenvolvimento do software cliente-servidor para acesso a bases de dados multimídia, orientadas a objetos (OODBMS);
- Desenvolvimento de software/hardware para processamento e apresentação, compressão e transmissão de documentos multimídia, com restrições de tempo-real e sincronização.

Neste artigo são apresentados alguns dos resultados referentes à camada Middleware. Na secção 2 resumem-se os aspectos mais importantes do padrão ODP a serem considerados na plataforma e na secção 3 discutem-se os principais conceitos envolvidos

na especificação OMG-CORBA. Na secção 4 desenvolve-se o modelo de implementação da camada Middleware, baseado nas especificações ORB ("Object Request Broker") e na análise de serviços e funções RM-ODP.

2 MODELO RM-ODP

A plataforma MultiWare apresentada neste trabalho deverá incluir o maior número possível dos conceitos e funções elaborados na padronização ODP. Atualmente encontra-se em estado adiantado de especificação o RM-ODP [2], considerado como um Framework para a futura padronização de outros padrões de Processamento Aberto Distribuído. Prevê-se que este documento esteja pronto durante os anos de 95 e 96. Para uma visão mais geral desses documentos recomenda-se o trabalho [3].

Dois tipos diferentes de padrões ODP deverão surgir posteriormente [4], um respeitante a componentes (funções) como o Trader e o outro relacionado com procedimentos de composição (por exemplo para alcançar transparências).

Nos próximos anos deverão ser elaborados vários padrões como:

- **funções ODP** (Tabela T1)
- **padrões de notações de linguagens dos pontos de vista ODP** com destaque para:
  - linguagem **IDL="Interface Description Language"** para especificação de interfaces computacionais;
  - **templates de bindings** para reusabilidade **e de configuração**;
  - padrões de modelamento do conteúdo de informação das referências de interfaces e de sua representação para transferência através de RPCs ("Remote Procedure Calls") padronizados (por ex. RPC-ISO);
- diversos **padrões de composição de componentes** por exemplo para a realização de:
  - transparências;
  - segurança;
  - gerenciamento de sistemas distribuídos e de distribuição do gerenciamento de sistemas.

FUNÇÃO	DESCRIÇÃO
Gerenciamento:	
Núcleo	controla alocação de recursos nos nós
Objeto	gerencia objetos de engenharia básicos - OEBs -(constituem as aplicações ODP)
Cluster	gerencia clusters (unidade de ativação, desativação e migração)
Cápsula	gerencia cápsulas (unidade de alocação de recurso)
Domínio Comunic.	gerencia os recursos de comunicação num domínio
Interface de Refer.	gerencia interf. de refer. (inform. necessária para estabelecimento de binding com uma interface de um OEB)
Transação:	garante propriedades transacionais para invocações
Grupo:	provê operações para suportar grupos
Repositório:	
Armazenamento	provê um repositório para dados, incluindo clusters desativados
Relocação	armazena inform. sobre interfaces que mudaram de localização
Repos. de Tipos	mantém especificação de tipos e relações entre tipos
Trading	armazena ofertas de serviços e os divulga quando requisitado
Segurança:	
Contr. de Acesso	previne o uso não autorizado de um agente (usuário)
Auditoria	provê monitoração e coleta de informação sobre as interações
Autenticação	provê a corroboração de que um agente tem a identidade reivindicada
Confidencial.	previne a descoberta de informação por agentes não autorizados
Integridade	previne a alteração ou destruição não autorizada de dados
Não-Repúdio	previne de um agente envolvido numa interação recusar o fato de ter participado
Transparência:	
Acesso	provê interoperabilidade entre arquit. de comp. e ling. de program. heterogêneas
Localização	requer a propag. de inform. sobre mudanças na localiz. de objetos
Transação	mascara a coordenação de operações transacionais
Falha	mascara a falha e a possível recuperação de objetos
Federação	suporta interoperabilidade entre diferentes domínios administrativos
Migração	habilita um cluster mover de uma cápsula para outra
Grupo	permite a construção transparente de grupos de interfaces
Recurso	mascara a desativação e reativação de clusters

Tabela 1: Funções ODP

Os ítens vistos acima são escolhidos pela sua importância para o projeto descrito neste trabalho. Por outro lado [4], será intensificada a interação do trabalho ODP com diversos consórcios de empresas, em particular com OMG. Esta organização publicou as especificações CORBA, que serão descritas na secção seguinte.

3 ESPECIFICAÇÃO OMG/CORBA

Para a realização do sistema em questão, propõe-se o uso da especificação **CORBA= "Common ORB Architechture"**, em que **ORB="Object Request Broker"** do Modelo desenvolvido pela OMG é chamado de **OMA= "Object Management Architecture"** [5] (Figura 2). Os Serviços de Objetos são a coleção de serviços (interfaces e objetos) que providenciam as funções básicas para uso e implementação de objetos. As Facilidades Comuns são coleções de serviços que providenciam funções úteis a diversas aplicações e os Objetos de Aplicação, aqueles que são específicos das aplicações do usuário final.

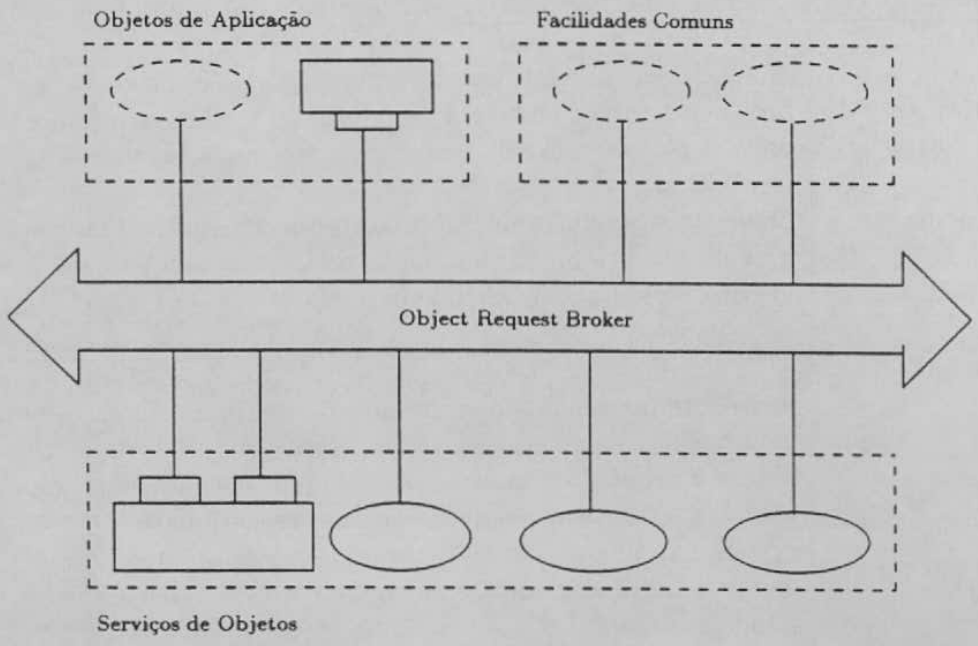

Figura 2: Arquitetura de Gerenciamento de Objetos

3.1 CONCEITOS BÁSICOS

Como é apresentado no documento [6] o ORB providencia a interoperabilidade entre aplicações residentes em máquinas diferentes de ambientes heterogêneos distribuídos.

O ORB é responsável por todos os mecanismos que levam uma requisição de um objeto cliente até um outro executor. O cliente (objeto que deseja que seja executada uma certa função) emite um request específico captado pelo ORB através de uma interface. O ORB responsabiliza-se por encontrar o objeto destino executor da operação solicitada (chamado em **OMA de "Object Implementation"**), e de prepará-lo para a sua recepção e execução, comunicando-lhe os parâmetros do request, e devolvendo ao cliente, sendo o caso, os resultados da execução. As operações anteriores ocorrem através de interfaces diferentes, definidas por uma linguagem IDL="Interface Definition Language":

- **do lado do cliente:**
  - **interfaces de invocação dinâmica**, permitindo a construção de invocações de objetos: o cliente pode especificar o objeto a ser invocado, a operação a ser executada, e o seu conjunto de parâmetros, através de uma chamada ou sequência delas;
  - **IDL-stubs** dependentes da interface do objeto-alvo, para a chamada de uma rotina específica de uma operação do objeto escolhido: os stubs permitem a programadores nas linguagens da aplicação (por ex. C++) fazer chamadas às operações definidas em IDL e em geral são desenvolvidos de forma específica e otimizada para cada implementação ORB;
- **do lado do objeto implementado:**
  - **IDL-skeletons**, para recepção das requisições dos clientes e chamada dos métodos correspondentes: trata-se de uma interface "up-call" na qual a implementação de objeto escreve rotinas conformes, a serem chamadas posteriormente pelo ORB;
  - **adaptadores de objeto**, como recurso primário de acesso da implementação de objeto aos serviços providenciados pelo ORB. Dada a grande diversidade de objetos, os adaptadores permitem ao ORB atender grupos particulares com requisitos semelhantes.

O sistema prevê interfaces ORB genéricas, idênticas para qualquer ORB, com algumas poucas operações comuns a todos os objetos: estas interfaces poderão ser usadas por clientes e implementações de objetos.

O ORB deverá também fornecer serviços respeitantes a:

- **repositório de interfaces** providenciando objetos persistentes para representação da informação IDL disponível em "run-time". Assim poderá localizar-se, de forma dinâmica, um objeto cuja interface não estava disponível no instante de compilação do programa;
- **repositório de implementações**, com a informação necessária para que o ORB localize e ative implementações de objetos.

LINGUAGEM IDL

A linguagem [6] é usada para descrever as interfaces e obedece às mesmas regras léxicas que C++. A sua gramática é um subconjunto de ANSI C++, com comandos adicionais para suporte dos mecanismos de invocação de operações. No documento CORBA é feita de forma completa a definição da sintaxe e semântica da linguagem bem como dos mecanismos de binding para a linguagem C. Em seguida descrevem-se em IDL todos os mecanismos das interfaces de invocação dinâmica, de repositório de interfaces, de interfaces ORBs e de adaptadores básicos de implementações de objetos.

3.2 DESCRIÇÃO DAS PRINCIPAIS INTERFACES

INTERFACE DE INVOCAÇÃO DINÂMICA (IID)

Esta interface permite através da operação "create-request" que programas-clientes criem e invoquem requisições (pseudo-objetos) a implementações de objetos que as executem. Uma requisição é formada por uma referência de objeto (alvo), uma operação (a ser executada) e uma lista de parâmetros in-out, repassados na sua forma nativa. A operação "invoke" é então chamada, passando o ORB a procurar o método adequado à sua execução. No caso do cliente chamar "send" após "invoke" ele será reativado imediatamente (IO assíncrono). Poderá ainda proceder-se, em qualquer instante, através de "create-list", à criação de listas de parâmetros, para futuro uso. O pseudo-objeto requisição repassado pelo ORB à implementação-destino contém um campo de informação chamado de "context" que corresponde à informação adicional do cliente e ambiente desejado de execução, e que poderá ser útil ao servidor durante execução. A atualização dinâmica do contexto é feita através de operações próprias (get-default, set-one-value, set/get/delete-values, ...).

INTERFACES DE REPOSITÓRIOS [5]

Um ORB necessita de informações sobre os objetos que manipula, disponíveis nos "stubs" (código que mapeia as rotinas C nos protocolos de comunicação do Core), ou, de forma mais geral, num **Repositório de Interfaces**. Prevêem-se várias interfaces de consulta para o acesso dos clientes, das implementações e do próprio ORB: não

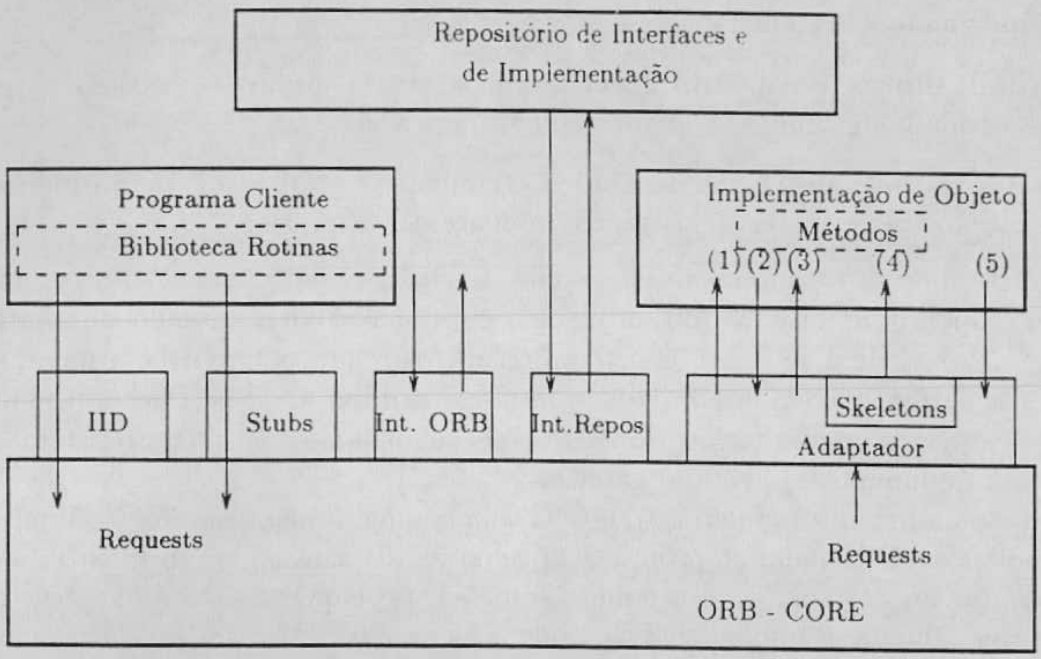

Figura 3

Figura 3: Exemplo de utilização do ORB

se especificaram mecanismos de inserção, pois esses são dependentes dos cenários de gerenciamento e de desenvolvimento dos sistemas (compilação de referências IDL, construção de objetos pela Interface Dinâmica, cópia de objetos entre repositórios,...). O repositório pressupõe memória persistente: no caso de um sistema de arquivos, haverá uma única cópia de cada interface, mas num OODB cópias múltiplas poderão ser distribuídas em diversas máquinas, para maior eficiência de acesso e disponibilidade.

ADAPTADORES DE OBJETOS [6]

Os adaptadores assumem importância especial nas implementações, sendo responsáveis pela geração e interpretação de referências de objetos, pela invocação dos métodos, segurança de interação, ativação e desativação de implementações, mapeamento de referências com os objetos correspondentes e pelo seu registro.

Diferentes ORBs providenciarão níveis diferentes de serviços, podendo-se, por ex., através do adaptador, permitir-se a uma implementação acesso a serviços mesmo quando não implementados no Core. Prevê-se um limite razoável do número de adaptadores, que deverão ser suficientemente genéricos para atender conjuntos de implementações com funções semelhantes. Por ex. consideram-se os seguintes tipos:

- **BOAs="Basic Object Adapter"** a ser usado pela maioria de objetos com

implementações tradicionais:

- **Adaptadores para Bibliotecas**, com acesso a arquivos persistentes, e sem necessidade de suporte a ativação ou autenticação;
- **Adaptadores de Bases de Dados Orientadas a Objetos**, permitindo acesso a objetos persistentes, registrados implicitamente na base.

CORBA permite a interação de grande diversidade de sistemas. Por exemplo o sistema mapeia seus objetos (ou entidades) em objetos ORB (usando uma interface BOA), que os poderá invocar: o sistema recebe os requests do ORB, atuando como um cliente na repassagem aos objetos propriamente ditos. Caso seja necessário poderá desenvolver-se um adaptador de propósito especial, ou ainda considerar o sistema como ORB remoto que interaja via um gateway.

No caso de um BOA (Figura 3), o ORB inicia uma implementação de Objetos (1), esta sinaliza ao ORB quando pronta para requests (2), chegando o primeiro request a implementação é ativada (3), e os requests repassados pelo ORB através dos "skeletons" (4). Posteriormente a implementação poderá acessar o BOA para criação de objetos, ativações, etc (5).

3.3 IMPLEMENTAÇÕES ORB E SUA INTERAÇÃO

Diferentes ORBs poderão fazer escolhas diferentes de implementação de acordo com as qualidades e propriedades das famílias especifícas de programas a que atendem. Poderão assim existir implementações ORB múltiplas, com representações distintas das referências de objetos (nome de objeto para sua identificação segura) e formas diferentes de execução das invocações: por exemplo deve ser possível a um cliente ter acesso simultâneo a duas referências de objetos gerenciadas por duas implementações ORB diferentes. O ORB-Core providencia a representação básica dos objetos e da comunicação dos requests. Acima do Core as interfaces mascaram as diferenças entre ORB Cores distintos.

Prevêem-se diversos tipos de implementações de ORBs que devem existir e cooperar entre si:

- **ORBs residentes no cliente ou na implementação**: desde que haja um mecanismo adequado de comunicação (mecanismos IPC transparentes a localização, por ex. RPCs) stubs implementados diretamente no cliente acessam as implementações;
- **ORBs baseados em servidores**: para centralização do gerenciamento do ORB todos os clientes e implementações podem comunicar-se com um ou mais servidores, cuja função será a de rotear as requisições dos clientes até as implementações.

No caso o ORB pode até ser um programa normal, em relação ao Sistema Operacional;

- **ORBs incluídos no S.O.**: para maior segurança, robustez e desempenho o ORB passa a ser considerado como um serviço básico do Sistema Operacional. O S.O. pode conhecer locais e estruturas dos clientes e implementações, por ex. evitando-se "marshalling" quando ambos estejam na mesma máquina;

- **ORBs baseados em Bibliotecas**: no caso de objetos "leves" e compartilháveis, a sua implementação em bibliotecas pode ser conveniente, passando os stubs a serem os próprios métodos que permitem o acesso aos dados a serem compartilhados.

Dada a existência de uma grande diversidade de técnicas de implementação de ORBs, é um requisito primário o uso de um modelo de alto nível que torne essas diferenças transparentes: esse modelo é em ORB o modelo OO ("Object Oriented") de invocação definido em IDL! Assim um request poderá atravessar diversos ORBs preservando a semântica de invocação. Para interligar dois ORBs existem basicamente dois grupos distintos de técnicas:

- **por referências embutidas**: por exemplo, um objeto implementado usando ORB1 poderá estar disponível no ORB2, criando-se um objeto em ORB2 que, invocado, simplesmente repasse as invocações aos objetos correspondentes em ORB1;

- **por tradução de protocolos**: sendo os ORBs funcionalmente semelhantes, contudo com diferenças nos detalhes de implementação, poderá simplesmente traduzir-se as requisições de um para outro.

3.4 ARQUITETURA DOS SERVIÇOS DE OBJETOS

Os Serviços de Objetos definem Interfaces IDL e a semântica de sequenciamento disponíveis no sistema para suporte das aplicações. Em [7] dá-se uma lista preliminar de serviços identificados (Tabela 2).

Para os serviços marcados com "X" na Tabela, já foram feitas requisições [8,9] de propostas pela OMG, devendo-se assim contar brevemente no mercado com produtos correspondentes. Planeja-se o término de propostas da maioria dos serviços até 1995.

SERVIÇOS		FUNÇÕES
Arquivo		Suporta mapeamento entre objetos armazenados de back-up e objetos ativos
Back-up/Restore		Suporta back-up e recuperação de objetos
Gerenc. de mudanças		Suporta identificação, evolução consistente e gerenciamento de versões e de configuração
Troca de dados		Suporta troca entre dois objetos de todo ou parte do estado visível de um deles
Contr. concorrência	X	Acesso concorrente a um ou mais objetos por um ou mais objetos
Notificação de eventos	X	Notificação de eventos a objetos interessados
Externalização	X	Transformação de objetos para representações adequadas ao armazenamento ou transferência
Repositório de Impl.		Gerenciamento de implementações de objetos
Instalação e Ativação		Distribuição, ativação, desativação e realocação de objetos gerenciados
Repositório de Interf.		Gerenciamento de definições de interfaces
Licenciamento		Gerenciamento de licenças (controle da remuneração de fornecedores de objetos)
Ciclo de Vida	X	Criação, deleção, cópia e equivalência de objetos
Nomeação	X	Mapeamento entre nomes e objetos
Controle Operacional		Controle do comportamento dinâmico de objetos gerenciados
Persistência	X	Persistência de objetos independentemente do tempo de vida do cliente e da implementação
Propriedades		Providencia atributos nomeados dinâmicos associados com um objeto
Query		Operações em conjuntos e coleções, suporte a indexamento
Relações	X	Suporte a associações entre dois ou mais objetos
Réplicas		Réplica explícita de objetos no ambiente distribuído e gerenciamento de consistência de cópias
Segurança		Controle de Acesso a objetos
Startup		Bootstraping e término de serviços de objetos
Threads		Suporte a threads múltiplos de controle de execução
Tempo	X	Suporte à sincronização de relógios no sistema distribuído
Trading		Mapeamento de serviços existentes com os solicitados pelos clientes
Transações	X	Execução atômica de uma ou mais operações

Tabela 2: Serviços de Objetos

4 MODELO DE IMPLEMENTAÇÃO DA CAMADA MIDDLEWARE

A camada **MIDDLEWARE** na plataforma **MULTIWARE** é a responsável por prover o ambiente de processamento aberto distribuído às aplicações, independentemente do tipo de aplicação. O modelo de implementação desta camada proposto neste trabalho é apresentado na Figura 4 e possui duas subcamadas: uma inferiora que é composta por plataformas comerciais já desenvolvidas (ANSAware, DCE e ORB) e outra superiora que oferece suporte aos seviços ODP definidos na padronização RM-ODP/ISO [2]. Portanto, a definição e a implementação deste modelo adotam o RM-ODP como base.

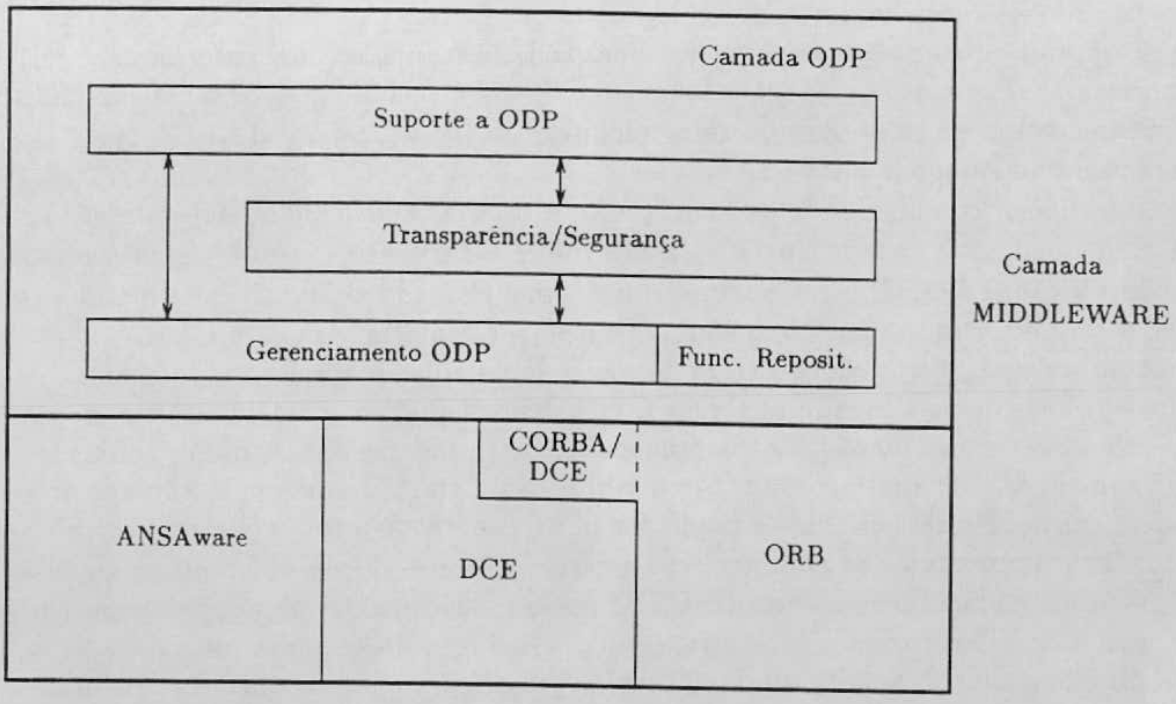

Figura 4

Figura 4: Camada MIDDLEWARE

Todas as plataformas citadas suportam o desenvolvimento, uso e manutenção de aplicações distribuídas. ANSAware, que segue a arquitetura ANSA ("Advanced Networked Systems Architecture") [10], é uma plataforma orientada a objetos que foi

desenvolvida em conformidade com o RM-ODP/ISO, DCE [11] é a plataforma orientada a processos da OSF e ORB é a plataforma orientada a objetos da OMG que foi analisada na secção 3.

Neste modelo de implementação, a subcamada superiora, chamada de **Camada ODP**, agrega abertura às plataformas, providenciando transparência de plataforma independentemente da orientação (processo ou objeto) e dos serviços disponíveis. A **Camada ODP** pode utilizar as seguintes configurações de plataformas: ANSAware, DCE, ORB e ORB sobre DCE. A incorporação destas plataformas na camada **MIDDLEWARE** será realizada de forma incremental.

A **Camada ODP** é por sua vez composta de três subcamadas: **Gerenciamento ODP**, **Transparência/Segurança** e **Suporte a ODP**. A primeira subcamada oferece os serviços básicos de gerenciamento permitindo a utilização de objetos de engenharia básicos, clusters e cápsulas. Este gerenciamento é auxiliado por funções de repositório da especificação ODP.

A segunda subcamada oferece as transparências e as funções de segurança da especificação ODP, enquanto que a última subcamada provê às aplicações as funcionalidades ODP, tais como serviços de suporte a comunicação, Trading, Suporte a Grupos e Comunicação multimídia, entre outros.

Note que a complexidade de cada uma destas três subcamadas depende da plataforma que provê serviços para esta **Camada ODP**. Por exemplo, a subcamada **Gerenciamento ODP** torna-se mais simples se a Camada ODP utilizar a plataforma ANSAware e torna-se mais complexa se utilizar a plataforma DCE ou ORB.

A subcamada **Suporte a ODP** interage com a subcamada **Transparência/Segurança** ou com a **Gerenciamento ODP** dependendo se, para uma comunicação específica, é ou não necessário as transparências e funções de segurança oferecidas pela subcamada **Transparência/Segurança**. Note que para diferentes comunicações podem ser necessários diferentes requisitos de transparência e/ou segurança. A subcamada **Transparência/Segurança** é responsável por prover estes diferentes requisitos.

A subcamada **Gerenciamento ODP** realiza basicamente as funções de gerenciamento e de repositório (exceto a função de Trading) apresentadas na Tabela 1, enquanto que a subcamada **Transparência/Segurança** realiza as funções de segurança e de transparência da Tabela.

Na fase atual do projeto, está sendo realizado o mapeamento dos conceitos ODP para a arquitetura CORBA, levando em consideração os aspectos apresentados na secção 3, com a finalidade de definir precisamente os serviços necessários na **Camada ODP** e em cada uma das subcamadas. Procedimento idêntico está sendo desenvolvido para a plataforma ANSAware. Como resultados iniciais, têm-se o refinamento da **Camada ODP** comentado anteriormente (Figura 4) e o refinamento da subcamada **Suporte a ODP** que será analisado em seguida (Figura 5).

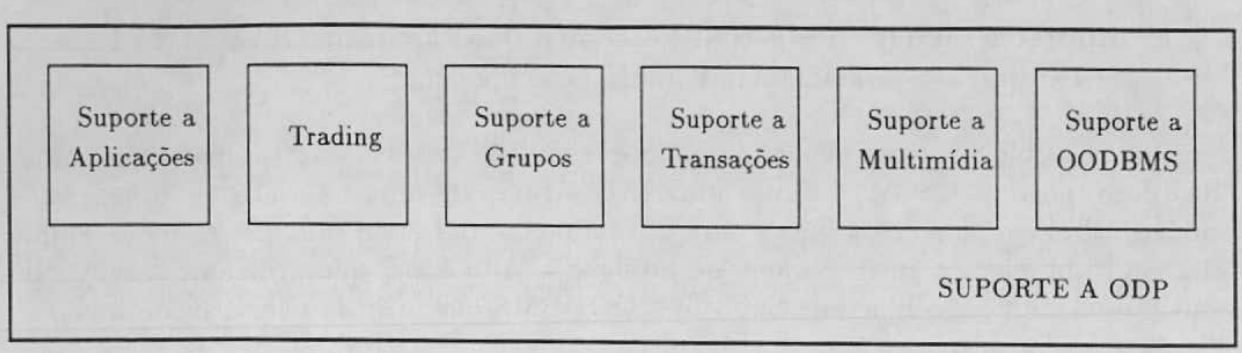

Figura 5

Figura 5: Subcamada Suporte a ODP

Os blocos funcionais que compõem a subcamada **Suporte a ODP** são apresentados na Figura 5. Estes blocos oferecem serviços da especificação ODP diretamente às aplicações ou através das camadas de Suporte a CSCW e Suporte a DAI como mostrado na Figura 1. Os blocos são os seguintes:

- **Suporte a Aplicações**: provê as funcionalidades básicas de uma plataforma de serviços abertos, tais como a definição e instanciação dos objetos (processos) que compõem as aplicações e como estes objetos estão estruturados em subsistemas, a definição dos requisitos desejados de transparência e segurança, e a implantação das comunicações necessárias (bindings) entre estes objetos, entre outras;

- **Trading**: oferece a negociação de serviços entre servidores (exportadores) e clientes (importadores). Exemplos de serviços são: exportar um serviço, procurar por um tipo de serviço, selecionar o melhor serviço segundo critérios (estáticos - por ex., certas qualidades de serviço e custo - e dinâmicos - por ex., tamanho da fila de espera) que são passados em parâmetros apropriados, entre outros;

- **Suporte a Grupos**: provê suporte à cooperação entre membros de um mesmo grupo, como por exemplo, a transmissão de uma invocação de um cliente para membros servidores apropriados do grupo, e a garantia do envio de invocações para membros do grupo numa determinada ordem;

- **Suporte a Transações**: garante a uma invocação de operação transacional ter as propriedades ACID (Atomicidade, Consistência, Isolamento e Durabilidade) requeridas;

- **Suporte a Multimídia**: permite o envio e a recepção de informação multimídia e de tempo real com qualidades de serviço desejáveis;

- **Suporte a OODBMS**: permite o acesso a Bases de Dados Orientadas a Objetos para armazenar principalmente informação multimídia.

Todos os blocos descritos acima oferecem serviços aos usuários (aplicações e camadas de Suporte a CSCW e Suporte a DAI) através de uma interface de aplicação, e podem também oferecer serviços para outros blocos desta subcamada. Como exemplo deste último caso, tem-se o bloco de Suporte a Aplicações que utiliza os serviços do bloco de Trading e o bloco de Suporte a OODBMS que utiliza os serviços do bloco de Suporte a Transações.

O bloco de Suporte a Grupos realiza as funções de Grupo do RM-ODP/ISO (Tabela 1), enquanto que o bloco de Suporte a Transações realiza as funções de Transação. O bloco de Trading realiza a função de Trading das funções de Repositório e o bloco de Suporte a Aplicações tem funcionalidades próprias como visto anteriormente.

A camada **MIDDLEWARE** pretende incorporar de maneira incremental as funcionalidades ODP apresentadas na Tabela 1. A Tabela 2 apresenta apenas algumas destas funcionalidades e, portanto, devem surgir implementações de objetos para ORB completando as funções ODP. Neste Projeto, as funcionalidades ODP da Tabela 2 são inicialmente aproveitadas, e pretende-se completar estas funcionalidades com os novos objetos que serão desenvolvidos no âmbito deste Projeto e com os novos objetos que aparecerão como produtos comerciais.

O desenvolvimento da camada **MIDDLEWARE** está na fase de projeto. A implementação da versão 1.0 desta camada está prevista para o final deste ano, e esta versão terá apenas as plataformas ANSAware e ORB na subcamada inferiora e algumas das subcamadas e blocos da **Camada ODP** (subcamada superiora). Serão priorizadas as implementações da subcamada **Gerenciamento ODP** com as funções de Repositório, parte do módulo de Transparência da subcamada **Transparênsia/Segurança**, e dos blocos de Suporte a Aplicações e Suporte a Grupos da subcamada **Suporte a ODP**.

5 CONCLUSÕES E TRABALHOS FUTUROS

As plataformas abertas multimídia são de grande importância no mercado crescente de inúmeras aplicações. O projeto considerado neste trabalho descreve a camada Middleware em desenvolvimento na UNICAMP e UNESP. Atualmente os trabalhos encontram-se ainda na fase de análise de requisitos e de elaboração de um modelo de implementação iterativo e incremental orientado a objetos. Nos próximos dois anos serão realizados os trabalhos de implementação propriamente dita, com a colocação em operação de um protótipo em estações de trabalho RISC-6000 interligadas por uma rede FDDI.

A camada Middleware foi especificada segundo o RM-ODP da ISO ainda em fase de elaboração. Mostra-se no trabalho que a adoção de diversas especificações OMG como a CORBA é de grande importância, pois elas correspondem a modelos reais de implementação e já possuem no mercado produtos (por ex. ORB-RPC) que podem ser incorporados no protótipo. Espera-se que diversos produtos e outras especificações de consórcios de empresas (OSF, OMG, X-OPEN,...) surjam nos próximos anos, viabilizando um rápido amadurecimento da área de sistemas de comunicação multimídia.

Por outro lado o trabalho de padronização ODP da ISO deverá evoluir para a especificação de diversas componentes simples e compostas. Haverá assim a possibilidade, com a experiência ganha no projeto, de interagir e contribuir para esses trabalhos.

AGRADECIMENTOS

Os autores agradecem à FAPESP (Projeto Temático N. 92/3507-0), ao CNPq e CAPES (programas de Bolsas de mestrado e doutorado e Programa RHAE), bem como a IBM (cessão de equipamentos e software), pelo apoio financeiro e logístico dado aos trabalhos de desenvolvimento. Os autores também agradecem aos bolsistas que estão neste projeto pelo trabalho conceitual e prático já realizado.

6 REFERÊNCIAS

[1] Mendes, M.J.; Loyolla, W.P.D.C. e Madeira, E.R.M. - "DEMOS: A Distributed Decision-Making Open Support System" - Fourth Workshop on Future Trends of Distributed Computing Systems - Lisboa, Portugal - Setembro 1993, pp. 208-214

[2] ISO/IEC JTC1/S C21, Basic Reference Model ODP - Part 1: Overview and Guide to Use; Part 2: Descriptive Model; Part 3:Prescriptive Model - Outubro 1993

[3] Tschammer, V.; Mendes, M.J.; Souza, W.L.; Madeira, E.R.M. e Loyolla, W.P.D.C. - "Processamento Distribuído Aberto e o Modelo RM-ODP/ISO" - 11o Simpósio Brasileiro de Redes de Computadores - Campinas, SP - Maio 1993, pp. 175-195

[4] Joint ISO/ITU ODP Working Group Turin Interim Meeting Output Action, Draft 2 - Novembro 1993

[5] Object Management Architecture Guide, Revisão 2.0, OMG TC Document 92.11.1 - Setembro 1992

[6] The Common Object Request Broker: Architectures and Specification, Revisão 1.1, OMG TC Document 91.12.1. - Dezembro 1991

[7] Object Services Architecture, Revisão 6.0, OMG TC Document 92.8.4 - Outubro 1992

[8] Object Services Request for Proposal 1, OMG TC Document 92.8.6 - Outubro 1992

[9] Object Request for Proposal 2, OMG TC Document 93.6.1 - Novembro 1993

[10] ANSA, ANSA Reference Manual, APM Ltd., 24 Hills Road, Cambridge, CB2IJP, UK, Março 1989

[11] OSF, Distributed Computing Environment, Setembro 1990