Implementação de Serviços Multimídia em CORBA

Rodrigo C. M. Prado*
DCA - FEEC - UNICAMP
Campinas, Brazil, 13083-970

Luiz A. Guedes†
DEE - CT - UFPA
Belém, Brazil, 66000-000

Luís F. Faina‡
DEINF - CETEC - UFU
Uberlândia, Brazil, 38400-902

Eleri Cardozo§
DCA - FEEC - UNICAMP
Campinas, Brazil, 13083-970

Abstract

Distributed multimedia systems (DMS) are receiving strong attention from the software industry. However, even considering such strong interest, there are few experiences accumulated regarding the methodological design and implementation of DMS. This lack of experience is explained by the complexity related to the management of information when multiple forms of media are integrated in a distributed computing infrastructure. The large amount of standards in the field of multimedia computing also contributes to the difficulties mentioned above.

With the aim of providing multimedia services in a standard and modular way, this paper proposes a model for distributed multimedia services based on the CORBA architecture. In order to evaluate the proposed model two scenarios derived from the videoconferencing domain are evaluated.

Keywords: Distributed Systems, Multimidia Systems, Multimidia Services, CORBA.

Resumo

Sistemas multimídia distribuídos vem recebendo grande atenção por parte da indústria de software. Porém, apesar desse interesse, ainda há pouca experiência quanto ao projeto e implementação de forma sistemática de tais sistemas. Esta carência deve-se basicamente à dificuldade de gerência da informação imposta pela integração de várias formas de dados em uma infra-estrutura distribuída de computação, aliada à grande variedade de padrões de informação existente na área.

Com o objetivo de prover serviços multimídia de forma modular e padronizada, é apresentado neste artigo um modelo de serviços multimídia distribuídos baseado na arquitetura CORBA. Para validar a eficiencia desse modelo são apresentados alguns cenários oriundos de um sistema de teleconferência.

Palavras-chave: Sistemas Distribuídos, Sistemas Multimídia, Serviços Multimídia, CORBA.

1 Introdução

Sistemas multimídia se caracterizam pela capacidade de manipulação de vários tipos de meios, desde os mais simples como textos e gráficos (meios estáticos), até os mais ricos, como animação, áudio e vídeo (meios contínuos). O interesse por tais tipos de sistemas decorre da grande variedade de possíveis aplicações, que vai desde teleconferência e documentos multimídia até educação à distância e trabalho cooperativo em grupo. Entretanto, o uso de meios contínuos em sistemas computacionais, diferentemente dos meios estáticos, implica em capturar, processar, transmitir e exibir grandes volumes de dados continuamente e por longos períodos de tempo. Além disso, estes dados devem ser capturados e exibidos de forma cadenciada ao longo do tempo.

*prado@dca.fee.unicamp.br
†affonso@dca.fee.unicamp.br
‡lffaina@dca.fee.unicamp.br
§eleri@dca.fee.unicamp.br

Sistemas multimídia podem ser divididos em duas classes básicas. Uma diz respeito aos sistemas que operam em uma única máquina ou sistemas multimídia centralizados (SMC); e a outra aos sistemas que operam em um ambiente distribuído ou sistemas multimídia distribuídos (SMD). Sendo que SMD são uma generalização de SMC, neste trabalho estamos particularmente interessados em SMD.

Apesar da relevância, ainda há pouca experiência relacionada ao projeto e implementação de SMD. Esta carência deve-se basicamente à dificuldade de manipulação dos meios contínuos com as atuais tecnologias de computação e comunicação, somada à grande variedade de padrões existentes para capturar, codificar, transmitir e exibir esses tipos de meios.

E é justamente visando prover serviços padronizados para desenvolvimento de aplicações multimídia distribuídas em ambientes heterogêneos que apresentamos nossa proposta de implementação para esses serviços. No nosso modelo, os serviços multimídia são implementados como objetos em conformidade com a padronização *Common Object Resquest Broker Architecture* (CORBA) da *Object Management Group* (OMG) [1]. Neste contexto os serviços multimídia se dividem basicamente em serviços de dispositivos, serviços de transporte e serviços de configuração e controle.

O restante deste trabalho está dividido da seguinte forma: nas seções 2 e 3 é apresentada uma visão geral sobre sistemas e serviços multimídia, em seguida (seção 4) são descritas as características fundamentais da arquitetura CORBA. Na seção 5, é apresentado o nosso modelo de implementação de serviços multimídia sobre uma implementação CORBA. Também são exibidos alguns resultados provenientes da implementação de um protótipo de uma aplicação de teleconferência. Finalmente, são apresentadas as conclusões sobre o trabalho, além de indicações de desenvolvimentos futuros.

2 Sistemas Multimídia

Sistema multimídia em geral se caracterizam pela integração de diversos tipo de meios em uma única infra-estrutura computacional [2]. Notadamente, os meios mais relevantes são os contínuos, como vídeo e áudio. Já os sistemas multimídia distribuídos (SMD), em particular, necessitam, além da infra-estrutura computacional, de um suporte de comunicação, dado que os serviços estão geograficamente dispersos. Neste caso, o tipo de comunicação possui característica ponto-multiponto, como é o caso de teleconferencia [2].

Devido às características temporais dos meios contínuos, Qualidade de Serviço (QoS) é uma noção fundamental em sistemas multimídia e expressa o quanto são satisfatórios os serviços oferecidos pelo sistema. A qualidade de serviço de um determinado fluxo é função direta da quantidade de recursos destinada a ele (CPU, largura de banda de transmissão, capacidade de armazenamento, tipos de periféricos e sistemas de software). Usualmente, se caracteriza qualidade de serviço através de parâmetros objetivos, onde esses parâmetros são geralmente descritos a partir do ponto de vista do sistema. Um conjunto de parametros bem estabelecido na literatura é o seguinte [3]:

- atraso fim-a-fim: é o tempo transcorrido desde a captura ou acesso a uma base de dados de um segmento de fluxo contínuo até a sua apresentação;
- jitter fim-a-fim: é a variação do atraso fim-a-fim;
- taxa de pacotes perdidos (Packet Error Rate - PER): é a porcentagem de pacotes descartados durante o processo de transmissão;
- taxa de bits errados (Bit Error Rate - BER): é a porcentagem de bits que sofreram inversão de valor devido ao processo de transmissão.

De maneira geral, o atraso fim-a-fim afeta a interatividade da aplicação; o jitter fim-a-fim afeta o nível de continuidade do meio contínuo e BER/PER têm impacto no grau de distorção do dados. Limites usuais desses parâmetros em teleconferência são [3] [4]: 100-250 ms para o atraso fim-a-fim; 5-10 ms para o jitter fim-a-fim; 0,001-0,01% para o PER e 0,01-0,1% para o BER. Quanto menores forem os valores destes parâmetros maiores serão os índices de qualidade de serviço da aplicação.

Os fluxos contínuos necessitam passar por várias etapas desde sua captura até a exibição (a figura 1 ilustra o caso de um fluxo de vídeo). Tais fluxos são digitalizados e tratados como uma seqüência de segmentos, cada uma consistindo de uma saída de um dispositivo (câmera, microfone, etc.) amostrada a uma razão

constante. Estes segmentos demandam algum processamento antes e depois de serem transmitidos e/ou armazenados, como codificação/descodificação, filtragem e compactação/descompactação. Estas etapas tem influência direta nos parâmetros de QoS.

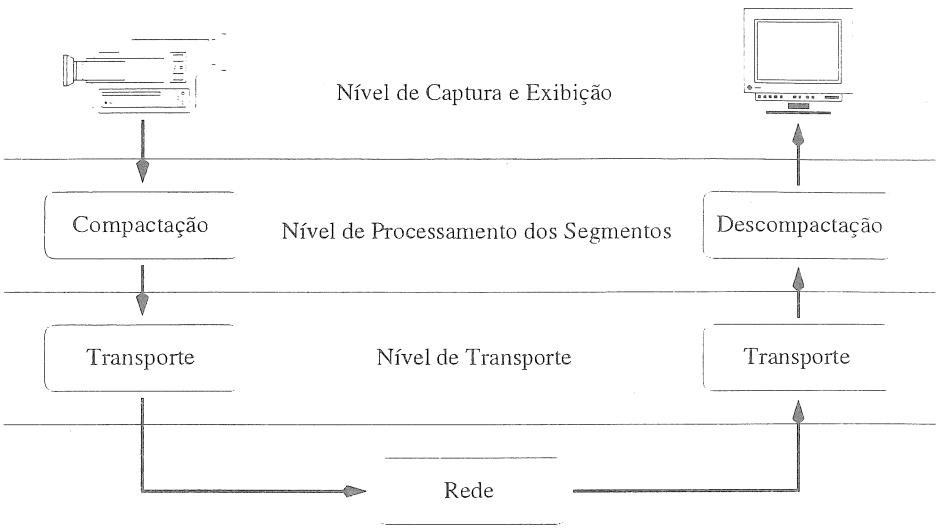

Fig. 1: Cenário das etapas de um fluxo de vídeo.

Como podemos observar a partir de figura 1, para que os fluxos contínuos possam ser entregues e exibidos de forma cadenciada no tempo, tem-se a necessidade de garantir níveis de qualidade de serviço ao longo de todas as etapas, ou seja, desde a captura até a sua exibição. Porém, estes requisitos usualmente são satisfeitos parcialmente, o que tem um impacto direto na qualidade do serviço final oferecida pelo sistema. De maneira geral, quanto maior o nível de satisfação destes requisitos parciais maior será a qualidade do serviço oferecido pelo sistema.

3 Serviços Multimídia

Considera-se serviço multimídia aqueles que operam meios contínuos. Podemos classificar os tipos de serviços multimídia em três grandes classes: serviços de dispositivos; serviços de transporte ou transmissão e serviços de configuração e controle.

3.1 Serviços de Dispositivos

Os serviços de dispositivos são responsáveis basicamente pelas atividades de captura, exibição e tratamento dos dados de meios contínuos. A captura pode ser proveniente de dispositivos físicos, como camera e microfone, ou de base de dados, para o caso de informação previamente armazenada. O primeiro caso é típico em aplicações como teleconferência, enquanto o segundo é mais próprio para video sob-demanda. Exibições de fluxos de vídeo e áudio são efetuados por periféricos como tela gráfica e alto falante.

Porém, entre a captura e sua subsequente exibição, um fluxo contínuo necessita sofrer uma série de processamento. Deste modo, um sinal de fluxo contínuo depois de ser capturado e antes de ser transportado via rede (no lado do produtor de informação), deve ser filtrado, para retirada de componentes de frequências espúrias, e sofrer um processo de compactação, para otimizar os recursos de rede. Finalmente, no lado do receptor, após ser transportado via rede e antes de ser exibido, esse sinal deve passar por um procedimento de descompactação, que realiza uma atividade funcional inversa àquela realizada na compactação.

3.2 Serviços de Transporte

Esses serviços têm como responsabilidade transportar os dados de meios contínuos desde sua fonte até seus receptores. Os serviços de transporte podem ser orientados a conexão (com garantia de entrega) ou sem conexão (sem garantia de entrega). Particularmente para os fluxos de meios contínuos o processo de

retransmissão, típicos dos serviços orientados a conexão, pode acarretar efeitos indesejáveis na qualidade do serviço de transporte, devido ao aumento do jitter e atraso. Além disto, pode adicionar uma carga desnecessária ao transporte de dados de tempo crítico se os dados estão sendo transmitidos por um sistema de transporte de alta confialibilidade.

Como é aceitável um certo nível de perda de informação nos dados de meios contínuos, pois tais dados possuem alto grau de redundância (por exemplo, vídeo), e a periodicidade na entrega dos dados é um fator determinante na caracterização da qualidade do serviço, sendo portanto mais apropriado utilizar mecanismos de transporte leves com serviços sem conexão.

3.3 Serviços de Configuração e Controle

Os serviços de configuração e controle se caracterizam por tratarem com os aspectos de gerência dos outros dois tipos de serviços previamente abordados.

Por configuração, entende-se a seleção de padrões compatíveis para todas as etapas por que percorrem os fluxos contínuos. Por exemplo, um vídeo compactado num padrão MPEG [5] não pode ser descompactado via algoritmo CellB [6]. Num outro exemplo, um fluxo de áudio capturado a uma taxa de 8 KHz e com 16 bits de precisão não pode ser exibido a 16 Khz, mesmo mantendo-se o nível de precisão, pois suascaracterísticas sonoras seriam totalmente deturpadas.

Já os serviços de controle correspondem a atividade de monitoramento dos níveis de qualidade de serviço dos fluxos, medidos a partir dos parâmetros de QoS, além das atividades que atuam propriamente sobre as características dos fluxos contínuos. É importante ressaltar que estas atividades podem ser desencadeadas pelo processo de monitoramento.

Podemos citar como atividade de atuação o serviço de sincronização entre fluxos de áudio e vídeo originados de fontes localizadas numa mesma estação. É usual que o procedimento de sincronização seja realizado através do paradigma de fluxos mestre e escravos, proposto em [7], onde um fluxo escravo, (vídeo) tenta seguir o fluxo mestre (áudio). Para tal, o sincronizador atua no fluxo escravo atrasando-o ou adiantando-o, conforme o caso.

Outra atividade de atuação seria o processo de adaptação do padrão de um determinado fluxo, visando manter índices de QoS previamente estabelecidos. Um cenário seria o caso de detecção de altos índices de perda de pacotes na transmissão de um fluxo de vídeo por motivo de sobrecarga na rede, que levaria o sistema a diminuir a taxa de captura dos quadros de vídeo para tentar diminuir o nível de perda de informação.

Para que se possa utilizar esses serviços multimídia eficientemente, é necessário que eles apresentem interfaces bem especificadas e padronizadas, além de facilidade de distribuição, de tal modo que aplicações multimídia tenham seus custos de desenvolvimento reduzidos consideravelmente.

Os serviços multimídia devem ser especificados de forma clara e para isto as características de abstração e encapsulação, típicas da orientação a objetos, são bastante convenientes.

Já a padronização dos serviços é necessária para que aplicações multimídia possam executar sobre diversos domínios, tornando-se mais independente possível da infra-estrutura de implementação.

O suporte a serviços distribuídos é fundamental em aplicações multimídia, pois boa parte de tais aplicações são de natureza distribuída, como teleconferência e vídeo sob-demanda.

Então, nessa perspectiva, o modelo CORBA de objetos distribuídos se apresenta como uma arquitetura bastante apropriada para suportar esses tipos de serviços multimídia distribuídos. Na próxima seção serão apresentadas as noções básicas dessa arquiterura.

4 O Modelo CORBA de Objetos Distribuídos

A especificação CORBA [8] do consórcio OMG tem sua origem na necessidade de uma padronização de sistemas distribuídos orientados a objeto. A arquitetura distribuída orientada a objeto surgiu como uma solução para os problemas relacionados à complexidade e diversidade do software. Este paradigma possibilita a diversas aplicações, escritas em diferentes linguagens, executar sobre múltiplos sistemas operacionais e tecnologias de rede. Sistemas distribuídos oferecem a infra-estrutura básica que permite abstrair as camadas de comunicação enquanto a orientação a objeto fornece um *framework* para encapsulamento e reuso necessários à integração de aplicações. A especificação *Object Management Architecture* (OMA) é a visão completa da OMG de um ambiente distribuído. Esta visão é composta de cinco componentes principais, conforme mostra a figura 2.

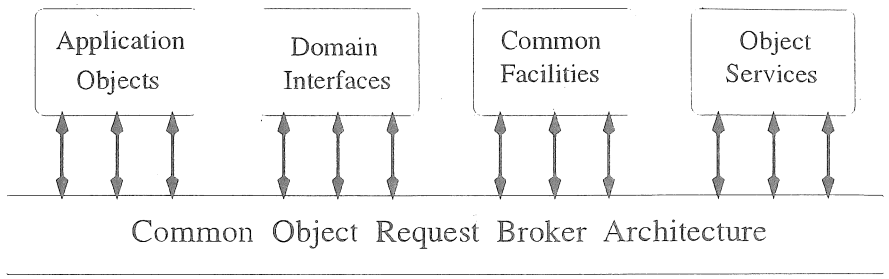

Fig. 2: Especificação da *Object Management Architecture*.

Object Request Broker (ORB) é o mecanismo de comunicação da especificação. O ORB fornece uma infra-estrutura que permite a interação entre objetos independente da plataforma e técnicas utilizadas na implementação destes objetos. A conformidade com a especificação CORBA garante portabilidade e interoperabilidade de objetos sobre uma rede heterogenea.

Object Services definem a gerência do ciclo de vida de objetos. Interfaces são fornecidas para criar objetos, controlar o acesso a objetos, manter o mapeamento de objetos relocados e controlar a relação entre tipos de objetos (gerência de classe). Também são providos ambientes genéricos nos quais os objetos podem executar suas tarefas. Exemplos de *Object Services* são: *Object Naming Service*, *Event Notification Service*, *Persistent Object Service* e *Transaction Service*. Os *Object Services* fornecem consistência às aplicações e auxiliam o aumento da produtividade do programador.

Common Facilities correspondem a um conjunto de aplicações genéricas que podem ser configuradas de acordo com os requisitos específicos de uma determinada aplicação. Estas facilidades se situam no nível de usuário, como por exemplo, facilidades de impressão, correio eletrônico e gerencia de documentos e banco de dados.

Domain Interfaces representam áreas verticais que fornecem funcionalidades de interesse do usuário final em domínios de aplicação particulares. As interfaces podem combinar algumas Common Facilities e Object Services, mas são projetadas para realizar tarefas particulares dentro de um determinado mercado ou indústria específicos.

Application Objects representam componentes (objetos) que realizam tarefas particulares ao usuário final. Uma aplicação é tipicamente construída a partir de um grande número de objetos básicos - alguns específicos à aplicação, outros ao domínio, alguns construídos a partir de Object Services e outros de um conjunto de Common Facilities.

A especificação CORBA utiliza para construção e integração de aplicações distribuídas os paradigmas de orientação a objetos e cliente-servidor. Neste modelo, os objetos são os componentes de distribuição e podem desempenhar o papel de cliente, quando requisitam uma operação, ou de servidor, quando recebem, executam e respondem à requisições (figura 3).

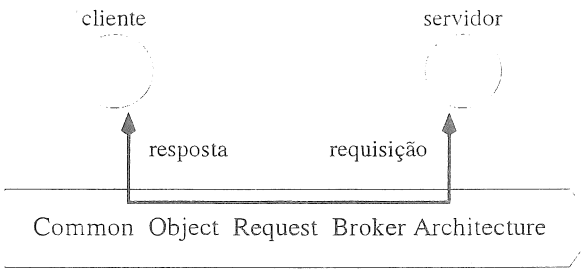

Fig. 3: Interação Cliente/Servidor via CORBA.

Para permitir interações entre objetos, cada um deve notificar aos seus potenciais clientes quais operações são possíveis e como elas devem ser invocadas. Em outra palavras, devem definir suas interfaces. A linguagem *Interface Definition Language* (IDL), também especificada pela OMG, define o tipo do objeto, seus atributos, métodos e os parâmetros dos métodos.

Cada objeto está associado a um servidor, podendo o servidor gerenciar um conjunto de objetos com diferentes interfaces ou não. Os servidores proveem objetos para serem usados por um ou mais clientes. Se um servidor não estiver ativo ele pode ser ativado quando um de seus objetos recebe uma requisição. Para que isto seja possível todos os servidores devem ser registrados num repositório de implementações.

Quando o servidor não está no mesmo espaço de endereçamento do cliente, ele é considerado remoto, mesmo localizado na mesma máquina. O termo remoto é definido desta maneira porque muitas das regras se aplicam a operações entre espaços de endereçamento e entre máquinas. O termo serviço é empregado para designar uma funcionalidade provida ao resto do sistema por objetos de um ou mais servidores.

Visando transpor esta limitação atual, na próxima seção é apresentada nossa proposta de suporte de aplicações multimídia sobre CORBA. Nessa solução são projetados objetos de implementação voltados para aplicações multimídia, porém o fluxo contínuo é transportado por fora do ORB, utilizando-se para isto um sistema de transporte não orientado a conexão.

A atual versão da especificação CORBA não provê suporte a interfaces do tipo fluxo contínuo. Assim, não é possível a utilização de IDL para definir uma interface para o estabelecimento, transmissão e controle de um fluxo contínuo de áudio ou vídeo.

5 Arquitetura de Implementação

Nesta seção é apresentada a arquitetura de implementação para serviços multimídia em CORBA. Esses serviços são divididos em três categorias básicas: serviços de dispositivos, serviços de transporte e serviços de configuração e controle. São apresentadas também as interfaces em IDL de cada um dos objetos (figura 4) que disponibilizam estes serviços.

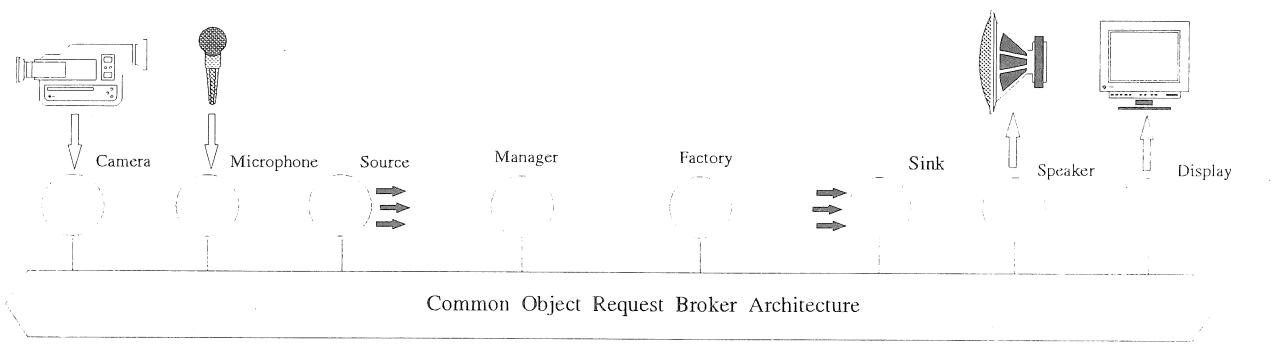

Fig. 4: Serviços Multimídia em CORBA.

5.1 Serviços de Dispositivos

Estes serviços estão associados aos componentes de *hardware* do sistema que são fontes ou destinos de fluxo multimídia. A abstração destes dispositivos é feita através dos seguintes objetos: `Microphone`, `Speaker`, `Camera` e `Display`, que são especializações do objeto `VirtualDevice`.

As interfaces descritas em IDL são:

```
interface VirtualDevice {
  long  startDevice();
  long  pauseDevice();
  long  continueDevice();
};
```

```
interface Microphone : VirtualDevice {
  attribute  AudioParam parameters;
};

interface Camera : VirtualDevice {
   attribute  VideoSourceParam parameters;
};

interface Speaker : VirtualDevice {
  attribute  AudioParam parameters;
};

interface Display : VirtualDevice {
  attribute  VideoSinkParam parameters;
};
```

A interface `VirtualDevice` apresenta métodos responsáveis por criar uma `thread` que repetidamente envia ou recebe do dispositivo (startDevice), suspende a execução da thread (pauseDevice) e retoma sua execução normal (continueDevice). Esta interface é herdada pelas outras quatro interfaces.

As interfaces `Microphone`, `Camera`, `Speaker` e `Display` apresentam o atributo *parameters*, que é uma estrutura contendo os parâmetros relacionados às características intrínsecas do dispositivo. Para áudio os parametros (`AudioParam`) são: taxa de amostragem do sinal, bits por amostra, tipo da codificação, volume, tamanho do buffer e porta de acesso. Para vídeo, do lado consumidor de fluxo os parametros (`VideoSinkParam`) são: tipo compressão, tipo de codificação, número de quadros por segundo, dimensão do quadro, número de bits por pixel. Do lado produtor de fluxo os parâmetros (`VideoSourceParam`) são os mesmos do lado consumidor acrescido de um seletor de transporte (port no caso da arquitetura TCP/IP).

5.2 Serviços de Transporte

Como a comunicação entre os objetos CORBA é tipicamente orientada a conexão e para o caso de fluxo contínuo, como foi ressaltado na Seção 3.2, o uso deste tipo de serviço não é o mais indicado, optou-se por fazer o transporte dos dados multimídia "por fora" do ORB, utilizando o protocolo RTP [9] sobre UDP/IP. Este protocolo tem como principal característica ser um protocolo leve, pois não é orientado a conexão e possui baixa sobrecarga adicional.

Os serviços de transporte são responsáveis pelo transporte de dados multimídia do objeto produtor, `Microphone` ou `Camera`, até o(s) objeto(s) consumidor(es), `Speaker` ou `Display`. Tais serviços são providos, respectivamente, pelos objetos `Source` e `Sink`, cujas interfaces de controle são:

```
interface Source {
  long  startSource();
  long  connectSink(in string sink);
  long  desconnectSink(in string sink);
};


interface Sink {
  readonly attribute Status qos;
  long  startSink();
};
```

Atualmente, o atributo *qos* incorpora os parametros jitter, banda utilizada e taxa de pacotes perdidos.

5.3 Serviços de Configuração e Controle

As funcionalidades desta categoria de serviço são providas pelos objetos `Factory` e `Manager`, com interfaces IDL descritas abaixo:

```
interface Factory {
  long  createMMObj(in string type, in string name);
  long  destroyMMObj(in string name);
};

interface Manager {
  attribute string sourceDevice;
  attribute string source;
  readonly attribute strSeq sinkDeviceList;
  readonly attribute strSeq sinkList;
  long  acceptDevice(in string device);
  long  removeDevice(in string device);
  long  acceptSink(in string sink);
  long  removeSink(in string sink);
  long  pauseAll();
  long  continueAll();
};
```

O objeto `Factory` é responsável pela criação (`createMMObj`) e destruição (`destroyMMObj`) de objetos do tipo `Microphone`, `Camera`, `Speaker`, `Display`, `Source`, `Sink` e `Manager`. O objeto `Manager` tem por responsabilidade agrupar informações sobre todos os objetos envolvidos na transmissão de um determinado fluxo multimídia (deste modo há um objeto `Manager` para cada fluxo da aplicação). O gerenciamento é realizado através da identificação do objeto produtor de fluxo (`sourceDevice`), do objeto de transporte associado ao produtor (`source`), de uma lista dos objetos consumidores de fluxo (`sinkDeviceList`) e uma lista com respectivos objetos de transporte associados (`sinkList`). Estão disponíveis métodos para adicionar e remover elementos destas listas (`acceptDevice` e `removeDevice` para `sinkDeviceList` e `acceptSink` e `removeSink` para `sinkList`. Existe, ainda, métodos para suspender e retomar o fluxo (`pauseAll` e `continueAll`).

5.4 Cenário de Aplicação em Teleconferência

A implementação da especificação CORBA utilizada foi ORBIX 2.1 da Iona Technologies [10] para o sistema operacional Solaris 2.5.1. O serviço de áudio foi implementado sobre o dispositivo nativo de audio da estação (/dev/audio), enquanto o serviço de vídeo foi construído através de funções de processamento de imagem da biblioteca *xil* [11]. As estações multimídia utilizadas foram plataformas SUN SPARCstation 5, configuradas com camera de vídeo, microfone e alto-falante e interligadas por uma rede ethernet. Para validação desses serviços foram gerados dois cenários de teleconferencia.

No cenário I (figura 5), uma única fonte emite para duas estações o fluxo de vídeo e áudio. Assim nas estações 2 e 3 são instanciados os objetos `Speaker` e `Display`, enquanto que na estação 1 os objetos `Microphone` e `Camera`. Cada um dos objetos `Microphone` e `Camera` está acompanhado de um objeto `Source`, enquanto cada objeto `Speaker` e `Display` está acompanhado de um objeto `Sink`.

No cenário II (figura 6), duas fonte de vídeo e áudio emitem para uma única estação os respectivos fluxos. Neste cenário, nas estações 1 e 2, objetos `Microphone` e `Camera` são instanciados, enquanto na estação 3 dois objetos `Display` e um `Speaker` são instanciados. Para cada objeto `Display` há um objeto `Sink`, enquanto que há um único `Speaker` e dois `Sink's`. Neste cenário, o dispositivo físico alto-falante é único no sistema.

A alta modularidade da arquitetura possibilitou a construção rápida e eficiente destes dois cenários de teleconferência. De fato, para geração dos cenários, foram apenas necessários a implementação de objetos clientes, com o propósito de instanciar os objetos de aplicação nas configurações adequadas. Além disso, os métodos `acceptDevice()`, `removeDevice()`, `acceptSink()` e `removeSink()` da interface `Manager` possibilitam a reconfiguração da aplicação de forma imediata. Estas características demonstram a versatilidade

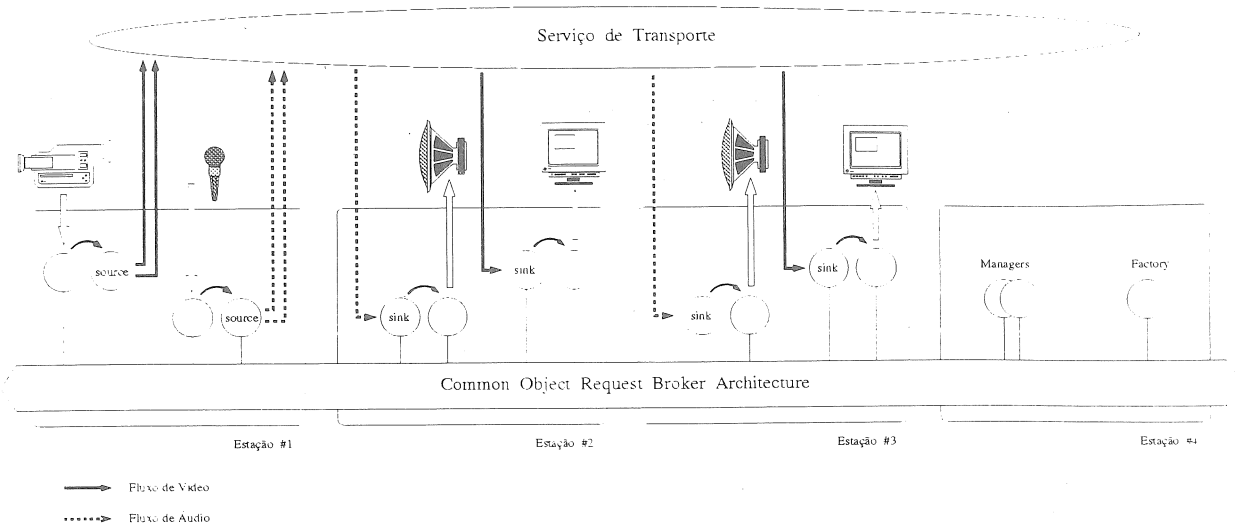

Fig. 5: Cenário de uma Teleconferencia I.

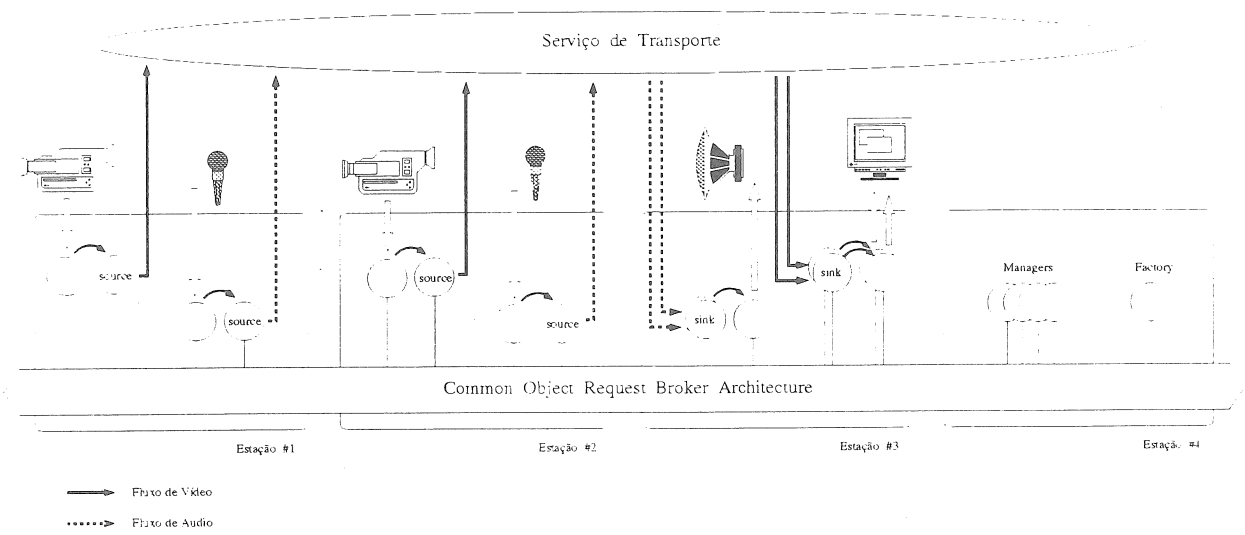

Fig. 6: Cenário de uma Teleconferencia II.

dessa arquitetura de serviços enquanto camada intermediária para desenvolvimento de aplicações multimídia distribuídas.

Os objetos de aplicação foram codificados em C++ como implementações de objetos CORBA. Aproximadamente 1500 linhas de código foram necessárias para descrição dos objetos do canal (ou seja, `Source` e `Sink`). Os parâmetros utilizados para video são número de quadros/seg. (10,15,30), tamanho da tela (640x320, 320x180) e tipo de compactação (JPEG, CELLB, MPEG). Para o audio foram utilizados taxa de amostragem (8000 Hz, 44100 Hz), e precisão (8 e 16 bits).

Para os cenários de teste, a infra-estrutura de comunicação (rede ethernet com cerca de 30 máquinas SUN SPARCstation) apresentou resultados de desempenho satisfatórios para suporte de serviços multimídia. Porém, devido esta tecnologia de rede não garantir reserva de recursos (por exemplo, banda de transmissão), ela não é uma solução factível para aplicações multimída de maior porte.

6 Conclusões e Futuros Trabalhos

Neste trabalho foi apresentado um modelo de implementação orientado a objetos distribuídos CORBA para a provisão de serviços multimídia distribuídos. A construção de um protótipo de um sistema de teleconferência, utilizando esses serviços multimídia, validou o modelo.

Devido à característica modular e distribuída do modelo, o desenvolvimento do protótipo de teleconferencia se mostrou extremamente simples e rápido, caracterizando assim, a eficiência desse modelo.

Atualmente, outros serviços multimídia estão em desenvolvimento. Dentre essses serviços podemos citar sincronização entre fluxos multimídia e adaptação de qualidade de serviço.

Agradecimentos

Esta pesquisa é financiada em parte pelas seguintes agencias brasileiras de fomento à pesquisa: FAPESP (processo 92/3507-0), CNPq (processo 300723/93-8) e CAPES.

References

[1] OMG Document, *The Common Object Request Broker: Architecture and Specification*, revision 2.0, July, 1995. URL http://www.omg.org/

[2] Williams, N., G.S. Blair, *Distributed Multimedia Application Survey* Internal Report N. MPG-91-11. Computing Department, Lancaster University, Bailrigg, Lancaster LA1 4YR, UK, April 1991.

[3] Vogel, A.; Kerhervé, B.; Bochmann, G. e Gecsei, J., *Distributed Multimedia and QoS: Survey*, IEEE Multimedia, Summer, pp. 10-18, 1995.

[4] Furht, B., *Multimedia Systems: An Overview*, IEEE Multimedia, pp. 47-59, Spring 1994.

[5] Le Gall, D., *MPEG: A Video Compression Standard for Multimedia Applications* , Communications of The ACM, April, 1991, Vol. 34, N. 4.

[6] *Cell Image Compression Byte Stream Description*, ftp://playground.sun.com/pub/multimedia / video / cellbytestream.ps.Z.

[7] Anderson, D.P. e Homsy, G., *A Continuous Media I/O Server and Its Synchronization Mechanism*, IEEE Computer, pp. 51-57, Outubro, 1991.

[8] Ben-Natan, R., *CORBA: A Guide to Common Object Request Broker Architecture*, McGraw Hill, 1995.

[9] *RTP: A Transport Protocol for Real-Time Aplications*, RFC 1889, URL http://ds.internet.net/rfc1889.txt

[10] IONA Tecnologies Ltda., *Orbix Distributed Object Technology - Programmer's Guide*, Release 2.x, October 1996.

[11] *Solaris XIL Imaging Library*, http://www.sun.co.jp/sunsoft/Products/Solaris-datasheets/XILTech-Spec.html