Sessões de Acesso e Serviço TINA Baseadas em Web

Eduardo J. Oliveira, Luís F. Faina e Eleri Cardozo
DCA - FEEC - UNICAMP - Caixa Postal 6101
Campinas, SP, Brasil, CEP 13083-970
Email: {ejo,lffaina,eleri}@dca.fee.unicamp.br

Sumário

O consórcio TINA (*Telecommunications Information Networking Architecture*) define uma arquitetura comum de *software* para serviços de telecomunicações e de informação, norteada por conceitos de "re-uso" de componentes de *software*, transparência de distribuição e utilização de padrões abertos, visando o desenvolvimento e gerência de serviços de qualquer complexidade.

Por sua vez, o crescimento da *World Wide Web* (WWW) na Internet a partir de meados desta década popularizou o uso dos Web *browsers* no acesso aos serviços de comunicação e informação, tornando-os padrão para a interface e interação com o usuário, assim como peça chave no desenvolvimento de aplicações.

Este artigo trata da utilização da tecnologia Web como mecanismo de acesso para serviços TINA. É apresentado um modelo de tecnologia contemplando a implementação do modelo computacional da Arquitetura de Serviços TINA no domínio do usuário e a interação dos componentes deste domínio com os componentes no domínio do provedor.

Palavras-chave: TINA-C, WWW, CORBA, Arquitetura de Serviços

Abstract

The Telecommunications Information Networking Architecture (TINA) Consortium defines a common software architecture to provide telecommunications and information services, using concepts of "re-use" of software components, distribution transparency, and open standards, aiming at the implementation and management of services of any complexity.

The explosion of the World Wide Web (WWW) in the Internet since the middle of the 90's has popularized the Web browsers as access mechanism to communication and information services. They has become the *de facto* standard for interfacing and interacting with the user and a key tool for the development of applications.

This paper concerns the use of Web technology as access mechanism to TINA services. It shows a Web-based technological model for realizing the TINA Service Architecture in the user domain and the interaction between the components in the user domain with those in the TINA provider domain.

Keywords: TINA-C, WWW, CORBA, Service Architecture

1 Introdução

Nos últimos 15 anos o mercado de telecomunicações tem passado por mudanças profundas devido a fatores como desregulamentação, aumento da competição entre as operadoras e aumento da demanda para o oferecimento de novos serviços contemplando, principalmente, comunicação multimídia e interatividade. Entretanto, a introdução desses novos serviços no âmbito de telecomunicações encontra uma forte barreira em uma infra-estrutura que tem sido voltada para a conexão de usuários, e não para o oferecimento de serviços. Assim, o futuro do mercado de telecomunicações aponta para um ambiente competitivo e globalizado em que as operadoras terão que diferenciar-se umas das outras não só no provimento e estabelecimento de conexões de transporte de rede, mas também, e principalmente, no oferecimento de novos serviços [1].

Nesse cenário, o TINA-C (*Telecommunications Information Networking Architecture Consortium*) [2] foi formado por operadoras de telecomunicações, fabricantes de computadores, *software* e equipamentos de rede, para definir uma arquitetura comum de *software* para serviços multimídia e de informação, no âmbito de telecomunicações. A arquitetura TINA, norteada por conceitos de "re-uso" de componentes de *software*, transparência de distribuição e utilização de padrões abertos, visa o desenvolvimento e gerência de serviços de qualquer complexidade, com maior rapidez e confiabilidade.

Por sua vez, a recente explosão da *World Wide Web* (WWW, ou simplesmente Web) na Internet, tem levado à disseminação dos serviços de comunicação e informação e não só popularizou o uso dos Web *browsers* como os tornou padrão para a interface e interação com o usuário na rede. Os Web *browsers* (e *servers*) têm evoluído e hoje, com a incorporação de novas tecnologias como JavaScript, Plug-ins e, principalmente, Java, mais do que simples ferramentas de apresentação de páginas, imagens e formulários, podem ser utilizados como uma poderosa ferramenta para o desenvolvimento de aplicações. Desta forma, sistemas envolvendo comunicação multimídia e interatividade estão se tornando cada vez mais e mais populares, fazendo da Internet uma verdadeira rede multi-serviço.

Na primeira fase do consórcio TINA (1993 a 1997), várias especificações da arquitetura foram produzidas. A segunda fase (1998 a 2000) tem por objetivo demonstrar os benefícios de TINA como uma plataforma para a provisão de serviços avançados [3] e, para tal, pretende validar seus modelos e especificações utilizando experimentos de campo. Um dos cenários imediatos para esses experimentos é a integração TINA-Internet. TINA contempla a provisão de serviços sobre redes de faixa larga com esquemas flexíveis de Qualidade de Serviço (QoS) e, através da sua Arquitetura de Serviços, pode ser utilizada como plataforma para integrar os dois tipos de rede: Internet e a rede de telecomunicações tradicional. A tecnologia Web, por sua vez, pode ser uma opção bastante interessante como mecanismo de acesso para aplicações TINA, pois permite a implementação do modelo computacional descrito por sua Arquitetura de Serviços no domínio do usuário.

Neste trabalho apresentamos a implementação de sessões de acesso e de serviço da arquitetura TINA utilizando-se a tecnologia Web. Este artigo está organizado da seguinte forma: a Seção 2 apresenta uma visão geral da arquitetura TINA, sendo sua Arquitetura de Serviços enfocada na Seção 3. Na Seção 4 é apresentado o modelo de tecnologia utilizado para implementação da Arquitetura de Serviços TINA e na Seção 5 descreve-se a implementação de sessões TINA utilizando-se as tecnologias apresentadas nas seções anteriores. Na Seção 6 é feita uma avaliação do modelo utilizado e, finalmente, na Seção 7 apresentamos as conclusões e alguns direcionamentos deste trabalho.

2 A Arquitetura TINA

A arquitetura TINA tem como objetivo prover uma arquitetura de *software* comum e coerente para oferecer serviços de telecomunicações e de informação [4], garantindo interoperabilidade, portabilidade e reusabilidade de componentes de *software*, independência com relação a tecnologias específicas, além de ajudar na criação e gerência de um sistema complexo. Para tal, baseia-se em quatro princípios básicos: análise orientada a objetos, distribuição, desacoplamento de componentes de *software* e separação lógica.

A análise orientada a objetos permite o particionamento da complexidade do sistema em um conjunto de modelos e seus pontos de referência. A distribuição dos componentes de *software* dos serviços acomoda características de tráfego, carga de rede e demanda de um consumidor específico. A distribuição em TINA é suportada pelo *Distributed Processing Environment* (DPE). As funções básicas do DPE TINA são proporcionar uma visão independente da tecnologia das camadas inferiores e esconder das aplicações a natureza distribuída do sistema, além de fornecer suporte à localização e interação remota dos objetos de aplicação. O desacoplamento dos componentes de *software* assegura que a mudança de um componente devido à mudança nos padrões, linguagens, programas ou redes em que ele se baseia, não afete os outros componentes.

TINA provê duas grandes separações lógicas. A primeira é entre as aplicações e o ambiente (isto é, o DPE) no qual elas executam. A segunda é a separação das aplicações em "parte específica do serviço" e "parte genérica de gerência e controle". TINA integra todas as funções de gerência e controle em uma arquitetura de *software* unificada e lógica, suportada por uma única plataforma de computação distribuída. Assim, ao invés de ser forçada a residir em um sistema em particular, estas funções podem ser disponibilizadas em qualquer ponto da rede.

Baseado no conceito de separação lógica, a arquitetura TINA [5] é estruturada em quatro (sub)arquiteturas: Arquitetura de Serviços, Arquitetura de Rede, Arquitetura de Gerência e Arquitetura Computacional (Figura 1). A Arquitetura de Serviços fornece conceitos e princípios para o projeto, especificação, implementação e gerência de serviços (tradicionais ou telemídia) de telecomunicações. A Arquitetura de Gerência fornece conceitos e princípios para o desenvolvimento de sistemas de *software* empregados na gerência dos serviços e da infra-estrutura que os suportam. A Arquitetura de Rede fornece conceitos e princípios para o projeto, especificação, implementação e gerência de redes de comunicação. Finalmente, a Arquitetura Computacional fornece conceitos e princípios para o projeto, especificação e implementação de sistemas de *software* distribuídos.

TINA utiliza o RM-ODP (*Reference Model for Open Distributed Processing*) [6] para descrever a organização de um sistema. TINA define, também, um conjunto de pontos de referência derivados do seu modelo de negócio. A conformidade com esses pontos de referência garante a interoperabilidade entre os produtos desenvolvidos em TINA.

O modelo de negócios TINA representa um paradigma universal, aplicável a qualquer situação em que um usuário faz uso de um serviço oferecido por um provedor, estabelecendo-se assim um relacionamento usuário-provedor que pode ser mapeado de maneira flexível em cada ponto de referência TINA. São estes pontos de referência que fornecem as especificações das interações que acontecem entre os diferentes domínios [4]. O modelo de negócios TINA descreve as diferentes partes envolvidas na provisão do serviço e o relacionamento entre elas. Os principais domínios definidos dentro desse modelo são: Consumidor, *Retailer*, Provedor de Serviço e Provedor de Conectividade (Figura 2).

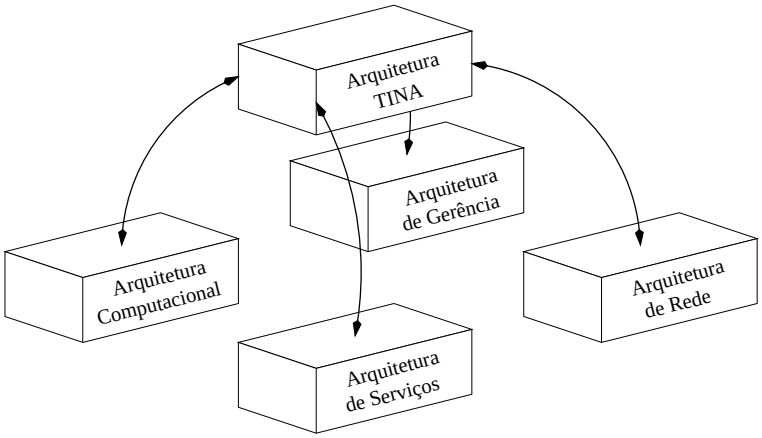

Figura 1: Subdivisões da Arquitetura TINA.

Consumidores compram serviços dos *Retailers*, e esses serviços podem, na realidade, ser providos por um terceiro (*3rd party*), o Provedor de Serviço. Os Provedores de Conectividade fornecem o transporte de rede e serviços de conectividade às demais partes envolvidas.

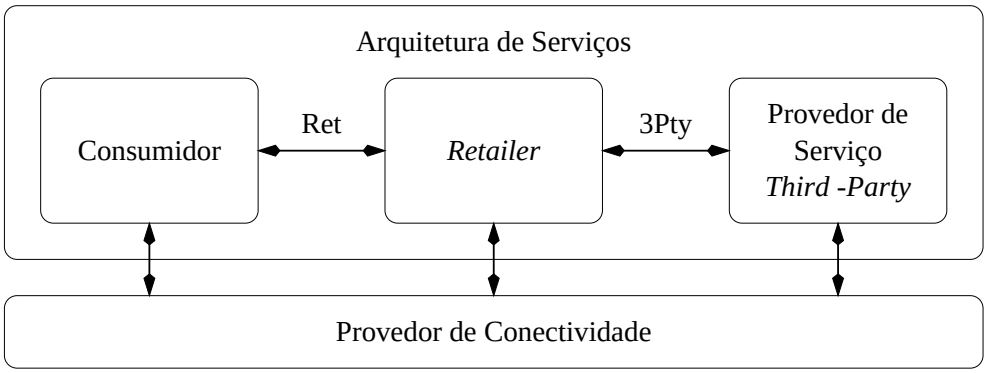

Figura 2: Modelo de Negócios da Arquitetura TINA.

Há de se notar que qualquer empresa pode se encarregar de algum ou todos esses papéis do modelo de negócios. Por exemplo, uma empresa pode fornecer um serviço em que desempenha tanto o papel de *Retailer* quanto de Provedor de Serviço, ou seja, os dois papéis podem se encontrar sob o mesmo domínio.

Os papéis de Consumidor, *Retailer*, Provedor de Serviços e seus respectivos pontos de referência (Ret e 3Pty na Figura 2) estão no escopo da Arquitetura de Serviços (Seção 3). O relacionamento desses com o Provedor de Conectividade estão no escopo da Arquitetura de Recursos de Rede [7].

3 Arquitetura de Serviços TINA

A Arquitetura de Serviços TINA [8] define conceitos, princípios e regras para se construir, estabelecer e operar serviços de telecomunicações e de informação. Para tal busca definir um conjunto de componentes de serviço, reusáveis e interoperáveis, e princípios e mecanismos para suporte mais efetivo na rede a serviços multimídia (com múltiplos

participantes e com suporte à mobilidade pessoal[1] e mobilidade de sessão[2]).

Na Arquitetura de Serviços TINA, o modelo tradicional de chamada (*call model*) de telecomunicações é substituído por um modelo mais flexível: o modelo de sessão. Uma sessão é uma relação temporária entre um grupo de recursos que são designados a efetuar, de forma conjunta, uma tarefa por um período de tempo. O conceito de sessão da Arquitetura de Serviços TINA visa separar os diferentes aspectos dos serviços e promover a distribuição de suas funções. Três tipos principais de sessões são identificados: Sessão de Acesso, Sessão de Serviço e Sessão de Comunicação.

A Figura 3 mostra como se processa a interconexão entre os componentes de serviço, com a separação nos diversos domínios, mostrando o caso simples em que dois usuários usam o serviço de um mesmo provedor que, neste caso, desempenha este papel tanto na parte de acesso quanto de uso do serviço.

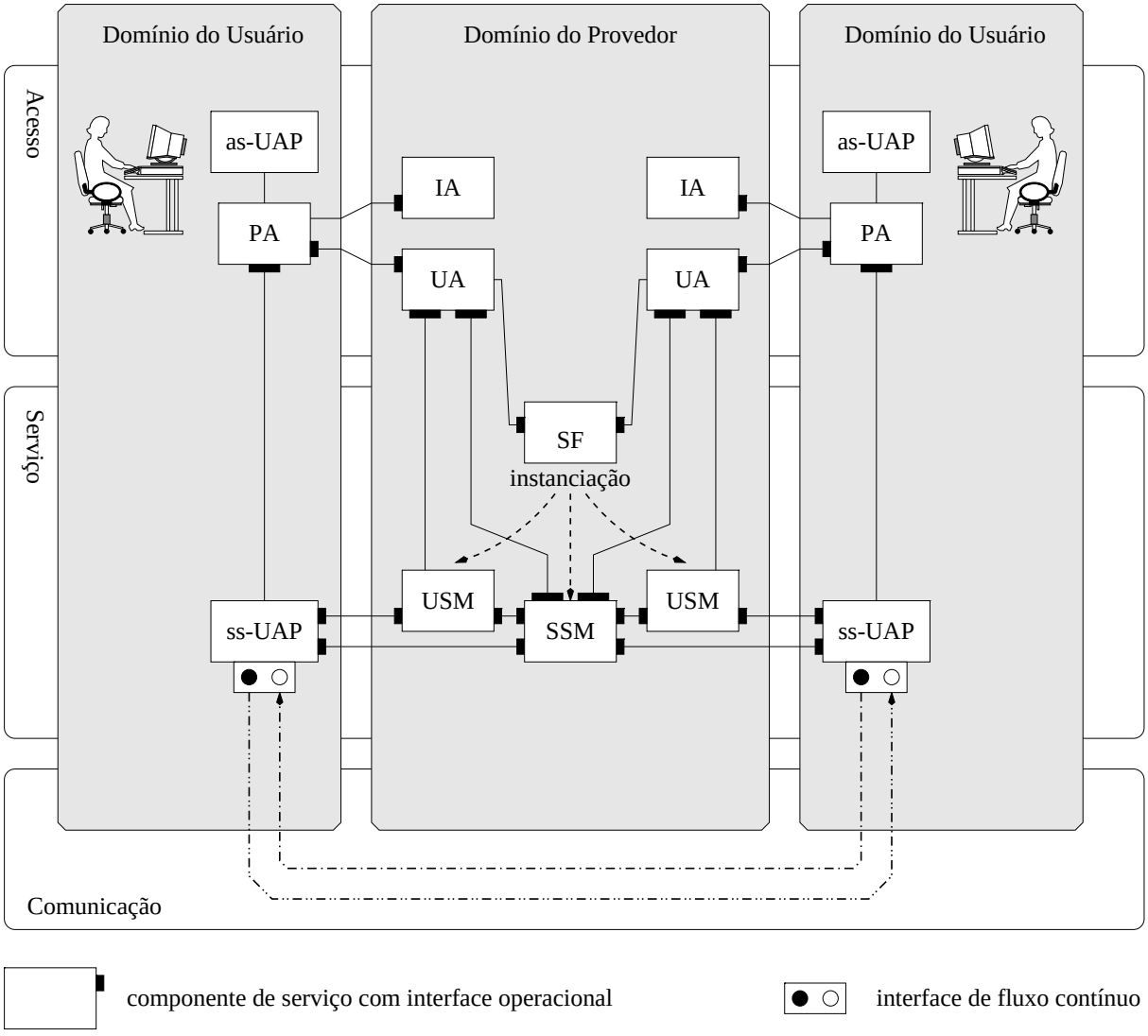

Figura 3: Componentes da Arquitetura de Serviço TINA.

[1]Por exemplo, um usuário pode fazer acesso ao provedor independentemente de sua localização física e do tipo do terminal sendo utilizado.

[2]Suspensão do uso de um serviço e sua retomada a partir de um outro equipamento.

3.1 Sessão de Acesso

A sessão de acesso promove o suporte necessário para que o usuário possa fazer acesso aos serviços TINA oferecidos pelo provedor. Este suporte se refere à identificação e autorização do usuário, bem como personalização do serviço. No domínio do usuário, a sessão de acesso representa as configurações e capacidades empregadas no relacionamento do usuário final com o provedor e, uma vez estabelecida, o usuário poderá, por exemplo, invocar serviços e receber convites para se unir a uma determinada sessão de serviço. No domínio do provedor, este deve manter informação sobre a identificação do usuário e demais capacidades a ele associadas.

Os principais componentes da sessão de acesso são mostrados na Figura 3. O agente inicial (**IA** - *Initial Agent*) é o ponto de contato inicial no domínio do provedor para o estabelecimento de uma sessão de acesso. A aplicação do usuário para sessão de acesso (**as-UAP** - *access session User Aplication*) corresponde à interface homem-máquina da aplicação provendo ao usuário mecanismo para interagir com o provedor. Esta interação é feita através do agente do provedor (**PA** - *Provider Agent*), que é o "representante" do provedor no domínio do usuário, e com o agente do usuário (**UA** - *User Agent*), que tem a função de "representar" o usuário no domínio do provedor, suportando mecanismos para o acesso personalizado aos serviços e determinando, por exemplo, se as capacidades associadas ao terminal utilizado permitem a execução de um determinado serviço. Um usuário pode ter mais de uma sessão de acesso com o provedor; entretanto, existe um único UA representando o usuário em um mesmo provedor.

Os componentes relativos à parte de acesso são independentes dos serviços oferecidos.

3.2 Sessão de Serviço

A sessão de serviço corresponde à provisão e uso do serviço propriamente dito. Ela é uma instância de um determinado tipo de serviço e inclui as informações necessárias para seu uso, bem como recursos de comunicação e negociação de qualidade de serviço e controla, ainda, os relacionamentos entre os membros participantes da sessão de serviço.

A visão global do serviço é representada pela sessão de serviço do provedor, enquanto a sessão de serviço do usuário corresponde à visão local de cada participante do serviço. A sessão de serviço é representada computacionalmente pelo gerente de sessão de serviço (**SSM** - *Service Session Manager*) e pelo gerente de sessão de usuário (**USM** - *User Session Manager*), que são instanciados a partir de uma fábrica de serviços (**SF** - *Service Factory*). Algumas das capacidades suportadas pelo SSM incluem guardar o estado da sessão de serviço e suportar ações de suspensão e retomada bem como a adição/remoção de usuários a/de uma sessão de serviço e o suporte a capacidades de gerência associadas à sessão de serviço (por exemplo, *accounting*). O USM também está relacionado às capacidades de manter o estado da sessão e às ações de suspensão/retomada em uma sessão, no que diz respeito ao lado do usuário.

Outro componente na sessão de serviço é o **ss-UAP** (*service session User Aplication*). Ele corresponde à aplicação do usuário, relacionada à sessão de serviço, que permite ao usuário fazer uso das capacidades de um serviço (por exemplo, iniciar e encerrar uma sessão, convidar outros usuários, suspender e retomar uma sessão, etc.).

3.3 Sessão de Comunicação

A sessão de comunicação abstrai do serviço os recursos de comunicação necessários para o estabelecimento e gerência das conexões de fluxo fim-a-fim, de maneira independente da tecnologia de rede.

Uma sessão de comunicação pode lidar com múltiplas conexões. Cabe notar porém que, a um dado momento, a uma sessão de comunicação deve estar associada apenas uma sessão de serviço que a controla.

A descrição da sessão de comunicação encontra-se no documento "TINA Network Resource Architecture" [7].

3.4 Dependência entre Sessões

Há um relacionamento e até mesmo, em alguns casos, uma dependência entre as sessões definidas. Essas dependências e relacionamentos são importantes para a compreensão da divisão e ordem de criação da sessão TINA e estão resumidas a seguir:

- uma sessão de serviço não pode existir sem uma sessão de acesso no lado do participante que a criou;
- várias sessões de serviço podem ser criadas a partir de uma única sessão de acesso;
- várias sessões de comunicação podem ser criadas a partir de uma única sessão de serviço;
- uma sessão de comunicação não pode existir sem uma sessão de serviço associada.

4 Modelo de Tecnologia para Sessões TINA

A especificação TINA firma-se mais na definição de pontos de referência e não cobre os aspectos de como implementar sua arquitetura. Assim, ainda hoje não existem sistemas TINA disponíveis publicamente e que sirvam de vitrine ilustrando a transição dos conceitos para a realidade. A tecnologia Web com o devido suporte à tecnologia de distribuição e, com quesitos como segurança suficientemente equacionados, pode ser utilizada principalmente na interação entre os domínios consumidor e provedor, viabilizando a experimentação e validação da Arquitetura de Serviços TINA.

A Figura 4 ilustra o modelo de tecnologia empregado na implementação de sessões TINA utilizando a tecnologia Web. Este modelo baseia-se nos conceitos da Arquitetura de Serviços TINA e nas especificações e interfaces definidas no ponto de referência Ret-RP (*Ret Reference Point*) de seu modelo de negócios (Figura 2). O nosso provedor TINA desempenha os papéis tanto de *Retailer* quanto de Provedor de Serviço. Cabe notar, ainda, que o servidor Web e o provedor TINA podem estar localizados em máquinas distintas ou na mesma máquina.

4.1 Tecnologia Web

A motivação do uso da tecnologia Web para a implementação de sessões TINA vem da constatação de que, além de ser uma tecnologia amplamente difundida e em evolução, ela apresenta algumas importantes características que viabilizam e facilitam a implementação de sessões TINA. Essas principais características são: uma interface gráfica padrão, sua

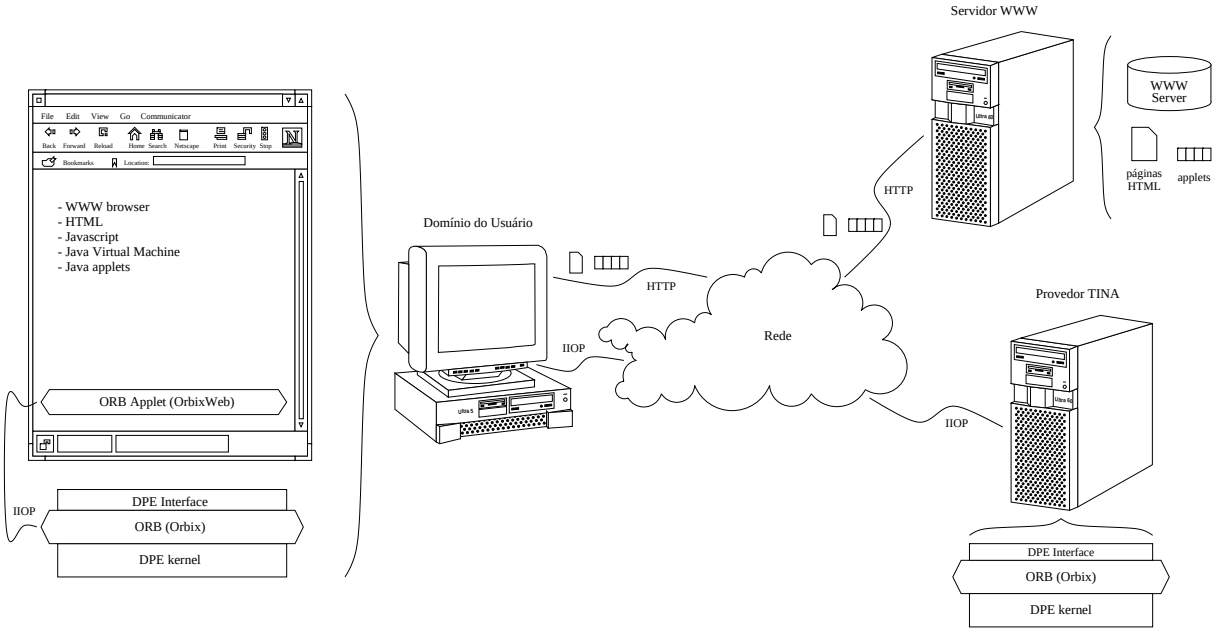

Figura 4: Modelo de tecnologia da implementação de sessões TINA.

característica multi-plataforma e, notadamente, a capacidade de extensão das funcionalidades do Web *browser* através da incorporação de tecnologias como *Javascript*, *Plug-in* e, principalmente, *applets* Java. Além disso, o Web *browser* implementa um mecanismo transparente e automático de *download* de *applets* a partir do servidor. Esse mecanismo vem ao encontro de uma premissa da Arquitetura de Serviços TINA: a independência de terminal, ou seja, o usuário não precisa fazer acesso ao provedor a partir de um terminal específico. Assim, TINA prevê que os componentes necessários para o estabelecimento do acesso ao provedor (UAPs e PA) sejam descarregados no domínio do usuário quando do estabelecimento do acesso.

Um aspecto também de interesse, decorrente do suporte a Java, é o suporte a *threads*. Em Java, o suporte a *threads* faz parte da própria linguagem, o que facilita sobremaneira a programação utilizando esse conceito.

4.2 Tecnologia para Processamento Distribuído

A arquitetura TINA prevê que as interações entre os domínios sejam suportadas por um ambiente de processamento distribuído (DPE). Essas interações são realizadas através de interfaces operacionais e demandam um protocolo que mantenha o estado da conexão durante invocações sucessivas. O protocolo HTTP utilizado pelos Web *browsers* foi desenvolvido primariamente para a requisição de documentos hipertexto e não mantém o estado da conexão durante invocações sucessivas. Desta forma, mecanismos auxiliares como *Cookies* [9] ou a geração de páginas dinâmicas com identificador de sessão (mapeando para um controle de sessão implementado no servidor) são necessários para se manter o estado de um serviço. Assim, os serviços desenvolvidos no contexto Web são, com frequência, desenvolvidos de forma independente uns dos outros e cada serviço possui o seu próprio sistema de acesso e gerenciamento. Além disso, os paradigmas utilizados nessas aplicações dificultam o "re-uso" de componentes de *software* e escalabilidade, pois são desenvolvidos em um nível de abstração que não é natural aos desenvolvedores de serviços [10].

O DPE TINA é baseado na arquitetura CORBA (*Common Object Request Broker Architecture*) [11], do OMG (*Object Management Group*), com modificações para requisitos de telecomunicações, como *streams* (fluxo contínuos de informação) e composição de objetos com múltiplas interfaces (por exemplo, uma para uso e outra para gerência). CORBA permite a objetos clientes invocar métodos em objetos servidores sem se preocuparem com a localização do objeto servidor, a linguagem de programação em que foram implementados e diferenças existentes entre o ambiente de execução dos objetos (*hardware*, sistemas operacionais, protocolos de comunicação). Essa funcionalidade é provida em CORBA pelo *Object Request Broker* (ORB) que, localizado no lado cliente e no lado servidor, cria e gerencia as comunicações entre os objetos. CORBA define uma linguagem neutra, *Interface Definition Language* (IDL), que permite descrever a especificação de um objeto CORBA e suas interfaces. As interfaces IDL podem ser implementadas em qualquer linguagem para a qual o mapeamento IDL esteja disponível, como Java e C++, por exemplo. Assim, aplicações CORBA escritas em linguagens diferentes podem interoperar.

CORBA também define o protocolo GIOP (*Generic Inter-ORB Protocol*) para a comunicação entre objetos remotos através da rede de comunicação e, desta forma, permite que haja interoperabilidade entre diferentes implementações de ORB. Em particular, CORBA define o IIOP (*Internet Inter-ORB Protocol*) para a interconexão de ORBs através de uma rede TCP/IP.

As implementações comerciais de CORBA em geral fornecem os ORBs também em forma de *applets* Java. Desta forma, os demais componentes (*applets*) executando no ambiente do Web *browser* podem fazer uso dessa tecnologia.

4.3 Interface de Fluxo Contínuo

Além de interfaces operacionais, o modelo computacional TINA prevê também interfaces de fluxo contínuo (*streams*). As interfaces de fluxo contínuo são utilizadas na sessão de comunicação e constituem o elemento básico para a comunicação multimídia entre as partes envolvidas.

Para o desenvolvimento de serviços multimídia distribuídos no ambiente proposto por TINA, são necessários, então, mecanismos para o estabelecimento, controle e gerência de fluxos multimídia.

Os serviços oferecidos na Internet basicamente contemplam apenas o modelo de transmissão utilizando a estratégia de "melhor esforço" (*best effort*). Assim, a Internet hoje padece de degradação no desempenho tanto em termos de disponibilidade quanto de qualidade de serviço. Ou seja, a Internet carece de suporte a *streams* de comunicação com garantia de qualidade de serviço, ainda que soluções para esse problema estejam sendo buscadas, como por exemplo o uso de RSVP [12].

Vários *frameworks* de *streams* multimídia estão disponíveis na Internet (RealAudio e RealVideo, Netshow, etc.). Entretanto esses *frameworks* tipicamente utilizam mecanismos proprietários para o estabelecimento e controle de *streams*, o que dificulta a sua adoção como solução para o ambiente proposto por TINA.

Para facilitar o desenvolvimento de aplicações multimídia distribuídas, o OMG definiu a especificação *Control and Management of Audio/Video Streams* (A/V Streams) [13] para uma infra-estrutura baseada em CORBA composta de um conjunto de objetos distribuídos, com operações padronizadas, que implementam um *framework* de *streams* multimídia. Algumas características de destaque desse *framework* são: suporte a conexões

"ponto-a-ponto" e "ponto-multiponto", suporte a múltiplos protocolos (incluindo TCP, UDP, RTP/UDP e ATM/AAL5) e fluxos com requisitos variados, como de qualidade de serviço, por exemplo. O suporte a *A/V Streams* e uma extensão a esse serviço integrando-o ao serviço de ciclo de vida foi incorporado ao nosso DPE TINA como descrito em [14].

Os componentes mais importantes do *A/V Streams* para as aplicações TINA são: dispositivo multimídia (`MMDevice`) e controlador de *stream* (`StreamCtrl`). Os dispositivos multimídia representam dispositivos lógicos ou físicos que geram ou consomem fluxos de mídia. Estes dispositivos comunicam-se via *streams* que podem agregar um conjunto de fluxos. Todas as operações de controle e sinalização são realizadas através do ORB, enquanto segmentos de mídia em um fluxo são transportados fora do ORB [15]. Os *streams* são gerenciados por um objeto de controle, `StreamCtrl`, que provê operações para estabelecimento e controle de fluxo (`start`, `stop`, `destroy`).

Como descrito em [14] o DPE, e consequentemente o serviço de *streams*, foi implementado em C++. Portanto essas funcionalidades estão disponíveis fora do ambiente do Web *browser*, como mostra a Figura 4. Uma vez que o controle e a sinalização são baseados em CORBA, esses componentes podem ser controlados pelo `StreamCtrl` no provedor através de invocações CORBA/IIOP. Uma implementação do serviço de *streams* baseada em Java e JMF (*Java Media Framework* [16]) já está pronta, porém ainda não foi integrada ao DPE.

5 Implementação de Sessões TINA

A implementação de sessões TINA utilizando-se a tecnologia Web é detalhada na Figura 5. No domínio do usuário, os componentes dos serviços TINA são disponibilizados através de um Web *browser*, Netscape Communicator, que é utilizado para interface gráfica com o usuário e ambiente para a execução de *applets* Java. A interface gráfica é implementada em Java juntamente com HTML e *Javascript*, através da tecnologia *LiveConnect* [17] da Netscape. No ambiente do Web *browser*, o DPE TINA é representado pela presença de um *applet* CORBA 2.0 que possui mapeamento IDL/Java (OrbixWeb [18]).

No domínio do provedor, um servidor Web é utilizado para armazenamento das páginas HTML e *download* de *applets* Java. Para a implementação dos componentes das sessões TINA, é utilizada a ferramenta Orbix 2.3 [19], que é uma implementação CORBA 2.0 com mapeamento IDL/C++.

O protocolo HTTP, nativo do *browser*, é utilizado nas primeiras etapas de acesso à página inicial do nosso provedor TINA e para o *download* dos *applets* Java que implementam os componentes de acesso e serviço no domínio do usuário. As interações entre estes componentes e os componentes no provedor são baseadas em CORBA/IIOP. A comunicação entre os componentes no domínio do usuário, ou seja, entre os *applets* que implementam as funcionalidades de PA e UAPs, são realizadas através de comunicação *inter-applet* provida pelo ambiente do Web *browser*.

5.1 Estabelecendo uma Sessão de Acesso TINA

O contato inicial com o provedor TINA é feito através de uma URL do servidor Web do provedor. A página inicial contém a *tag* HTML para buscar o *applet* com o componente as-UAP (*access session User Application*). Este inicialmente apresenta ao usuário apenas uma interface para a sua autenticação através de nome e senha e também

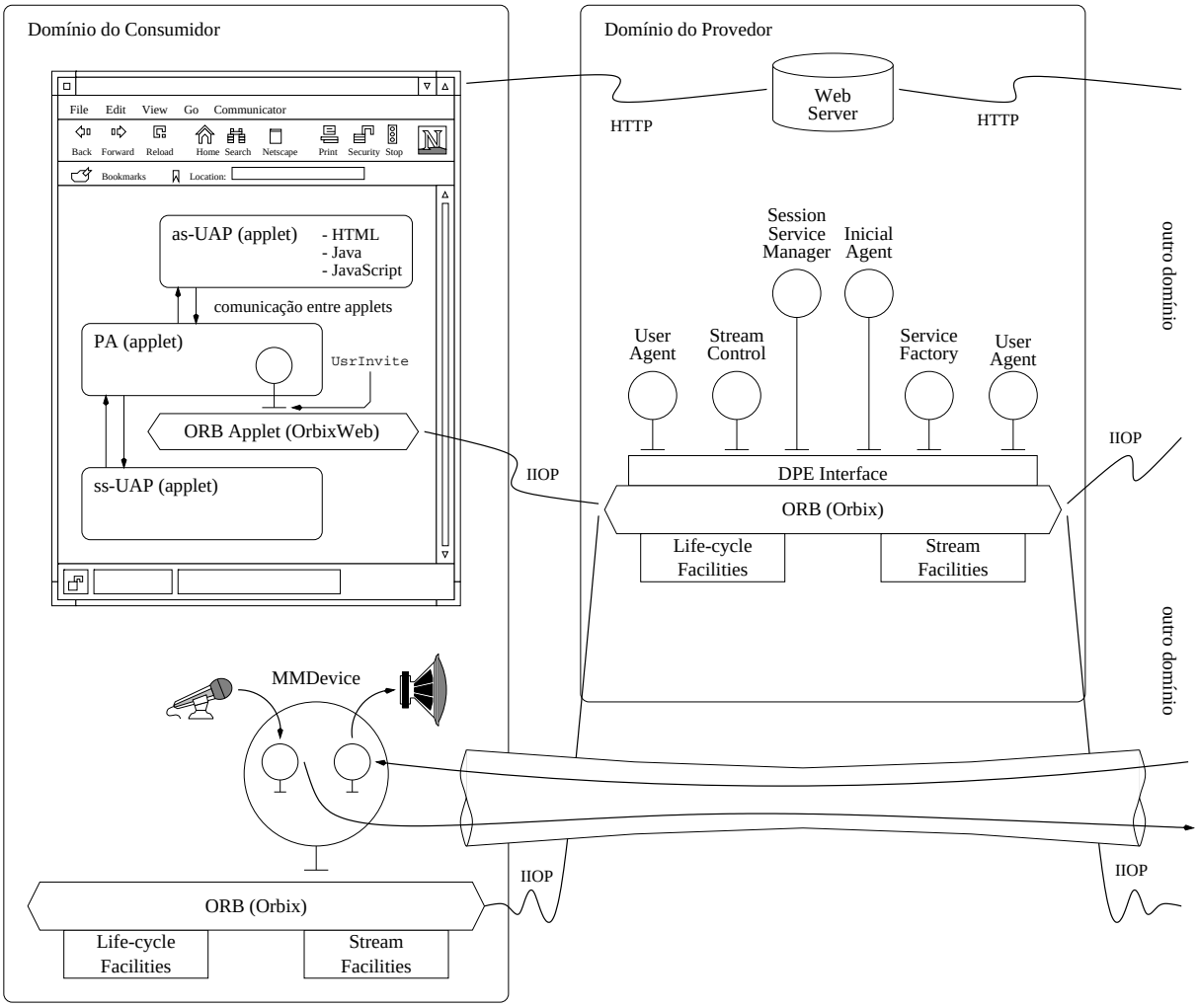

Figura 5: Implementação de Sessões TINA.

contém a interface gráfica para as demais interações relativas ao acesso. Uma outra *thread*, iniciada pelo as-UAP, encarrega-se de fazer o *download* do agente do provedor (PA). O PA representa o provedor TINA no domínio do usuário e é responsável por traduzir as requisições dos componentes que fazem a interação com o usuário nas invocações aos componentes no domínio do provedor TINA.

Após receber nome e senha do as-UAP, o PA os envia ao IA[3] para serem autenticados[4]. Uma vez autenticados com sucesso, o IA retorna ao PA um identificador de sessão (utilizado em todas as interações entre os componentes do domínio do usuário com os do provedor) e a interface do agente do usuário (UA), representando o usuário no domínio do provedor. O PA, então, registra junto ao UA a sua interface de convite (`UsrInvite`) de modo que possa, assincronamente, receber notificações do provedor como, por exemplo, convites de outros usuários para se unir a determinadas sessões de serviço. O PA também registra junto ao UA informações sobre o contexto do domínio do usuário, ou seja, informações sobre as capacidades de seu terminal. Uma vez concluídos com sucesso todos esses passos, a sessão de acesso é considerada estabelecida.

[3]Um passo intermediário pode ser acrescentado onde o PA requisita ao IA os métodos de autenticação disponíveis, para então escolher o mais adequado ao ambiente em questão.

[4]Pode-se utilizar um canal seguro (CORBA + SSL) para a autenticação.

5.2 Iniciando Sessões de Serviço

Uma vez estabelecida a sessão de acesso, o usuário tem disponível a lista de serviços em que está subscrito e pode criar uma nova sessão de um determinado serviço. A Figura 6 mostra uma sessão de acesso estabelecida para um usuário subscrito em um serviço chamado Audiophone. Uma vez que o usuário selecione esse serviço, essa informação é enviada pelo PA ao UA no domínio do provedor que interage com a fábrica do serviço (SF) para a criação do gerente de sessão de serviço (SSM). O SSM é responsável pela lógica global do serviço e suporta ações para suspensão e retomada da sessão e a adição e remoção de usuários. A referência à interface do SSM e o identificador da sessão de serviço são retornados ao UA que os repassa ao PA. O PA, por sua vez, promove o *download* do componente ss-UAP (*service session User Application*), com a interface gráfica específica do serviço (Figura 7).

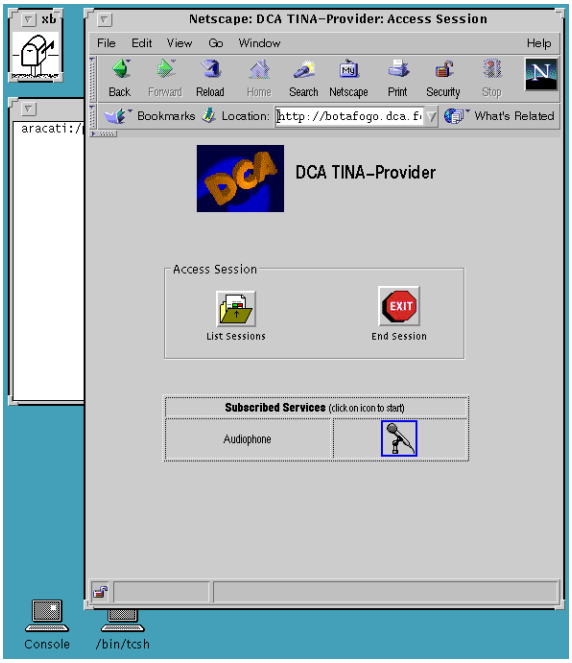

Figura 6: Sessão de Acesso.

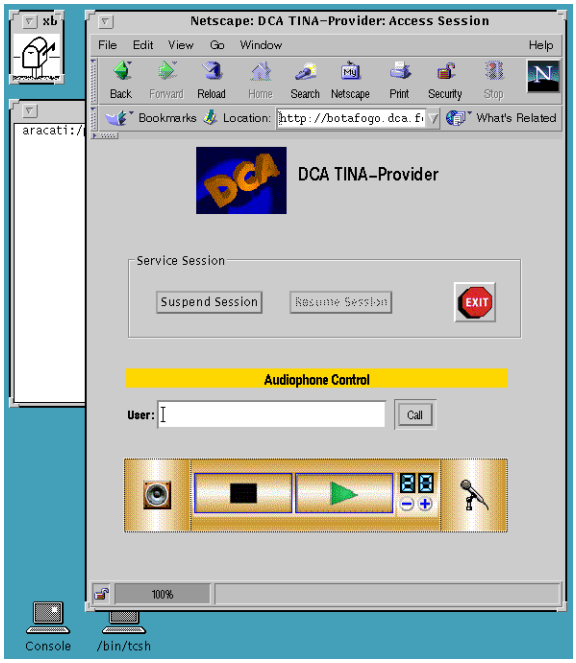

Figura 7: Sessão de Serviço Audiophone.

Uma vez que a sessão do serviço (Audiophone) é criada, o usuário pode convidar outro usuário para se unir a esta sessão fornecendo o nome do usuário destino no campo apropriado na interface gráfica do serviço (Figura 7). Esse convite é entregue ao destinatário através da interface do PA, no domínio do destinatário. Nossa implementação tem como premissa que o usuário destino deve ter pelo menos uma sessão de acesso estabelecida com o provedor para poder receber convites.

5.3 Estabelecendo o Fluxo de Comunicação

Uma vez aceito o convite de conexão, o SSM promove o estabelecimento da comunicação entre as partes, o que é feito através do controlador de *streams* (`StreamCtrl`). O `StreamCtrl` ativa os servidores de áudio (auto-falante e microfone) em cada um domínios envolvidos na comunicação (Figura 5) e estabelece os fluxos de comunicação entre eles.

Cada serviço pode demandar tipo de fluxo e qualidade de serviço diferentes. Esses requisitos são mapeados pelo SSM nos parâmetros de qualidade de serviço apropriados, utilizados pelo `StreamCtrl`. Desta forma, requisitos de áudio como "qualidade de CD", por exemplo, são convertidos nas propriedades de dispositivo tais como "*sample rate*", "*sample size*", etc.

5.4 Controle do Serviço

Através do ss-UAP, as partes podem controlar os fluxos multimídia utilizando os botões de *play* e *stop* (Figura 7). Estas ações são mapeadas em invocações ao `StreamCtrl`, que efetivamente controla os fluxos.

Outra funcionalidade incorporada ao serviço é a mobilidade de sessão de serviço, ou seja, a habilidade de suspender e retomar a participação do usuário em uma sessão de serviço. Quando o usuário, no domínio A, suspende sua participação na sessão de serviço com outro usuário no domínio B, estas ações são gerenciadas pelo SSM utilizando-se as funcionalidades de ciclo de vida (*life-cycle*) e *streams* providas pelo DPE [14]. A retomada da sessão suspensa pode ser realizada no mesmo terminal e domínio ou, mais interessantemente, através de outro terminal em um terceiro domínio (domínio C). Neste caso o SSM recebe do UA a localização do terminal no domínio C e restabelece automaticamente os fluxos entre as partes envolvidas, agora nos domínios B e C.

6 Avaliação do Modelo

Uma avaliação inicial do modelo utilizado mostra que a tecnologia Web é bastante atrativa para a implementação de sessões de acesso e serviço TINA, destacando-se o suporte à linguagem Java e o mecanismo para o *download* automático de componentes no domínio do usuário e sua execução no ambiente de segurança do Web *browser*. Entretanto essa mesma segurança acaba impondo algumas importantes restrições.

Os *applets* Java que executam nos Web *browsers* estão sujeitos às restrições do modelo de segurança para *applets*, também conhecido como "*sandbox*". Este modelo assegura que o código fruto de *download*, como no caso dos *applets*, é considerado "não-confiável" e portanto restrito. Um *applet* Java rodando nessa área restrita não pode, por exemplo, fazer conexões de rede para um servidor diferente daquele de onde foi transferido; nem tampouco para a máquina onde está sendo executado. Para estender a flexibilidade do modelo "*sandbox*" utilizamos *applets* "assinados". Com essa tecnologia, conhecida como "*object signing*" [20], *applets* podem fazer acesso aos recursos antes restritos baseado numa política definida pelo usuário. Entretanto *object signing* exige a arbitragem de uma terceira entidade, de certificação, que verifique a autenticidade do certificado que assina o *applet*.

Outra consideração diz respeito ao tempo de *download* do *applet* ORB do provedor para o Web *browser*. O tamanho do *applet* OrbixWeb na versão utilizada é de aproximadamente 800 KBytes. Somado ao tamanho das classes com os tipos básicos usados no desenvolvimento dos componentes TINA mais os próprios componentes, podemos chegar a um *download* inicial de mais de 900 Kbytes no estabelecimento da sessão de acesso. Dependendo da velocidade da rede sendo utilizada, o tempo de *download* inicial pode levar a um desconforto para o usuário. Uma alternativa seria a utilização do ORB nativo da distribuição do Web *browser*. Netscape Communicator possui um ORB pré-instalado

e IIOP-*compliant*, entretanto essa versão não tem sido atualizada pela *Netscape*. Java 1.2 [21] também já conta com um ORB em sua distribuição. Entretanto essa estratégia fica dependende da versão do Web *browser* e/ou do ORB nativo. Outra alternativa é a instalação do *applet* ORB mais classes básicas no ambiente (Web *browser*) do usuário.

A necessidade de execução de *applets* Java e a presença de um ORB executando no Web *browser* faz com que o terminal no domínio do usuário seja um cliente de maior poder e infra-estrutura computacional. Isso pode limitar o acesso aos serviços do provedor TINA na medida em que o usuário somente pode utilizar terminais com poder computacional para abrigar um Web *browser* com suporte a Java e objetos distribuídos. Terminais mais "leves", nesse contexto, poderiam ser contemplados disponibilizando-se *gateways* no servidor que intermediassem o acesso desses clientes. Por exemplo, um Web *browser* mais "leve", com suporte apenas para HTTP, poderia fazer acesso a um *gateway* HTTP/TINA, implementado através de um *servlet* [22]. Do ponto de vista do cliente HTTP, o *servlet* seria visto como um servidor Web; do ponto de vista do provedor TINA, ele seria visto como o componente PA do domínio do usuário.

A sessão de comunicação utiliza o suporte a *streams* (A/V Streams) incorporado ao DPE TINA como descrito na Seção 4.3. Por ser um padrão aberto e baseado em CORBA, A/V Streams oferece interoperabilidade com qualquer componente de *software* capaz de fazer acesso a um ORB. Além disso, o suporte a múltiplos protocolos permite a utilização de fluxos com diferentes requisitos de qualidade de serviço, o que dá à sessão de comunicação alta flexibilidade com relação à tecnologia da rede de comunicação. Por outro lado, a implementação utilizada é dependente de plataforma e precisa estar pré-instalada no ambiente do usuário. Uma alternativa para esse problema é a integração da implementação do serviço de *streams* utilizando Java e *Java Media Framework* (Seção 4.3).

7 Conclusões

Neste trabalho foi apresentada a implementação de um modelo para disponibilização de serviços multimídia baseado na Arquitetura de Serviços TINA. Em particular, foi apresentada a utilização da tecnologia Web para a implementação e disponibilização de componentes da Arquitetura de Serviços TINA no domínio do usuário, mais especificamente dos componentes das sessões de acesso e serviço, e a utilização dessa tecnologia na interação usuário-provedor. Também foi avaliado o modelo utilizado, levantando-se alguns problemas e suas possíveis soluções.

Com os resultados obtidos concluímos que a tecnologia Web é bastante atrativa para a implementação de acesso aos serviços TINA, no domínio do usuário. A partir dessa infra-estrutura (sessão de acesso e de serviço), novos serviços podem ser oferecidos com um sistema de acesso e gerência unificados. Como extensão deste trabalho estão sendo realizados a integração da implementação do serviço de *streams* utilizando Java e JMF, assim como o desenvolvimento de outros serviços.

Agradecimentos

Esta pesquisa é suportada pelas seguintes agências: FINEP(1588/96) e CNPQ (que provê bolsa de estudos para o primeiro autor). O Departamento de Informática da Universidade Federal de Uberlândia suporta o segundo autor.

Referências

[1] L. Lehman, M. Cadorin, and C.E. Wurgler. "Service Creation on a TINA Plataform: an Experience Report". In *TINA'97 Conference, Chile, Nov. 1997*, Nov. 1997.

[2] F. Dupuy, G. Nilsson, and Y. Inoue. "Tina consortium: toward networking telecommunications information services". *IEEE Communications Magazine*, 33(11):78–83, Nov. 1995.

[3] TINA-C. "Business Objectives for 1998-2000", 1998. `http://www.tinac.com/TINA2000/objectives.html`.

[4] Y. Inoue, D. Guha, and H. Berndt. "The TINA Consortium". *IEEE Communications Magazine*, 36(9), Set. 1998.

[5] M. Chapman and S. Montesi. "Overall Concepts and Principles of TINA". Technical Report Doc. TB_MDC_018_1.0_94 Version 1.0, TINA Consortium, `http://www.tinac.com/`, Fev. 1995.

[6] ISO/IEC 10746-1 / ITU-T Draft Recommendation X.901. "ODP Reference Model Part 1: Overview". Technical report, International Organization for Standardization and International Electrotechnical Committee, Jun. 1995.

[7] C. Abarca et al. "Network Resource Architecture". Technical report, TINA-C, `http://www.tinac.com`, Fev. 1997. Versão 3.0.

[8] C. Abarca et al. "Service Architecture". Technical Report Version 5.0, TINA Consortium, Jun. 1997.

[9] JavaScript Guide - Appendix C. *"Netscape Cookies"*. `http://developer1.netscape.com:80/docs/manuals/communicator/jsguide4/cookies.htm`.

[10] C. Smith. "Applying TINA-C Service Architecture to the Internet and Intranets". In *TINA'97 Conference*, Santiago, Chile, Nov. 1997.

[11] OMG. "The Common Object Request Broker: Architecture and Specification". Technical Report Version 2.2, Object Management Group, Mar. 1998. `http://www.omg.org`.

[12] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. "RSVP - A New Resource ReSerVation Protocol". *IEEE Network*, 7(5), Sep./Oct. 1993.

[13] Inc. IONA Technologies, Inc. Lucent Technologies, and AG Siemens-Nixdorf. "Control and Management of A/V Streams". Technical Report telecom/97-05-07, Object Management Group, `http://www.omg.org/`, Out. 1997.

[14] A.S. Pinto, E.J. Oliveira, L.F. Faina, and E. Cardozo. "TINA-based Environment for Mobile Multimedia Services". In *TINA'99 Conference*, Abr. 1999.

[15] L.F. Faina, E.J. Oliveira, R.C.M. Prado, and E. Cardozo. "Developing Multimedia Applications with the OMG Streaming Framework". In *ICC'99 - International Conference on Communications, Mini Conference on Multimedia Applications, Services and Technologies*, Jun. 1999.

[16] Sun Microsystems. *"Java Media Framework Programmer's Guide (v 0.5)"*, Dez. 1998. http://java.sun.com/products/java-media/jmf.

[17] JavaScript Guide - Chapter 5. *"LiveConnect"*. http://developer1.netscape.com:80/docs/manuals/communicator/jsguide4/livecon.htm.

[18] IONA Technologies Ltd. *"OrbixWeb 2.0.1 - Programming Guide"*, Out. 1996.

[19] IONA Technologies Ltd. *"Orbix 2.3c MT - Programming Guide"*, Nov. 1997.

[20] *"Netscape Object Signing: Establishing Trust for Downloaded Software"*. http://developer.netscape.com/docs/manuals/signedobj/trust/index.htm.

[21] Sun Microsystems. *"Java(tm) Development Kit, version 1.2"*, Dez. 1998. http://www.java.sun.com/products/jdk/1.2/.

[22] *"Servlets: A Technical Discussion"*. http://www.servlet.com/srvpres/index.html.