Universidade do Porto

FEUP Faculdade de Engenharia

Licenciatura em Engenharia Electrotécnica e de Computadores

PSTFC

Mecanismo de Suporte Para Reuniões de Trabalho em Ambiente Distribuído

INESC Porto

Bernardo Miguel Ribeiro Gaspar

Orientador da FEUP:

Prof. José António Ruela Simões Fernandes

Orientadores do INESC Porto:

Engº Carlos Pinho

Engº Ricardo Morla

Engº Rui Campos

Julho de 2006

Ciência.Inovação 2010 Programa Operacional Ciência e Inovação 2010

Universidade do Porto

FEUP Faculdade de Engenharia

Licenciatura em Engenharia Electrotécnica e de Computadores

PSTFC

Mecanismo de Suporte Para Reuniões de Trabalho em Ambiente Distribuído

INESC Porto

Bernardo Miguel Ribeiro Gaspar

Orientador da FEUP:

Prof. José António Ruela Simões Fernandes

Orientadores do INESC Porto:

Engº Carlos Pinho

Engº Ricardo Morla

Engº Rui Campos

Julho de 2006

Ciência.Inovação 2010 Programa Operacional Ciência e Inovação 2010

Agradecimentos

Esta secção é dedicada a todas as pessoas que contribuiram, directa ou indirectamente, para que a realização deste estágio fosse possível:

- Aos Engenheiros Carlos Pinho, Ricardo Morla e Rui Campos, pela ideia do projecto e por todo o apoio e ajuda prestados.
- Ao orientador da FEUP, o Prof. José António Ruela Simões Fernandes, por ter aceite orientar o meu estágio.
- Aos meus amigos, pelas distracções proporcionadas.
- À minha família, por estarem sempre presentes.
- À minha namorada, pela paciência que teve nos momentos mais difíceis.
- Ao POCI2010, pelo apoio financeiro.

Índice

Índice de Figuras

1 – Introdução

Neste capítulo referem-se as linhas gerais deste trabalho: a sua contextualização, os seus objectivos, a sua importância e os contributos desenvolvidos. Insere-se ainda uma curta descrição da estrutura do relatório e do que se poderá encontrar em cada capítulo.

1.1 – Enquadramento do trabalho

A penetração e adopção das novas Tecnologias de Informação trouxe alterações significativas ao quotidiano do ser humano. Ao nível empresarial, essas tecnologias facilitam a execução de inúmeras tarefas, promovem a comunicação e interactividade entre pessoas, o que se reflecte num aumento da eficiência de funcionários e empresas. As reuniões de trabalho são um exemplo marcante deste facto; passaram a desenrolar-se com suporte de meios tecnológicos avançados, contrastando com o cenário tradicional em que a interacção entre os intervenientes na reunião se fazia recorrendo a meios tradicionais, tais como quadro, papel e caneta, acetatos, etc. De facto, a utilização de computadores e projectores de vídeo (*datashows* na terminologia anglo-saxónica) para a projecção de apresentações em formato electrónico (ex. *PowerPoint*) que suportam as discussões numa reunião de trabalho tornou-se prática comum. Por outro lado, a massificação dos computadores portáteis pessoais fez com que os participantes de uma reunião se façam acompanhar do seu próprio computador, o que potencia maior interactividade entre os intervenientes. Apesar de promover a comunicação, o sistema de

datashow apresenta algumas desvantagens. Por um lado, implica a deslocação do apresentador do seu lugar até junto do projector, eventuais trocas de lugar e mudança de cabos, o que compromete o fluxo normal da reunião e o conforto dos intervenientes seus directos (i.e., oradores). Por outro lado, o sistema de *datashow* apresenta-se como uma alternativa com custos significativos.

Neste contexto importa potenciar os meios tecnológicos ao dispor dos intervenientes de uma reunião, assim como ultrapassar as limitações do sistema de *datashow*. Em vez da utilização deste último é importante tirar partido do equipamento que os próprios utilizadores transportam consigo para as reuniões, nomeadamente os seus computadores pessoais portáteis, de modo a efectuar o mesmo tipo de apresentações de forma distribuída e com base numa infra-estrutura de comunicação. No trabalho aqui apresentado pretende-se desenvolver este aspecto, propondo uma solução que ultrapasse as limitações das soluções existentes.

1.2 – Objectivos do trabalho

Com este trabalho pretende-se desenvolver uma solução capaz de promover a comunicação, interactividade e conforto dos intervenientes de uma reunião, bem como reduzir a dependência tecnológica do *datashow* que assume actualmente um papel central e determinante nas referidas reuniões.

A solução desenvolvida deverá ser inovadora e baseada em técnicas de comunicação de dados e princípios de sistemas distribuídos. Isto porque, de um modo geral, todos os participantes possuem o seu computador portátil e encontram-se frequentemente ligados a uma rede sem fios.

1.3 – Contribuições

Neste trabalho foi desenvolvida uma aplicação distribuída de suporte a apresentações de slides *PowerPoint* em tempo real. A aplicação baseia-se num modelo de sinalização que permite obter melhores desempenhos que as soluções existentes baseadas noutros modelos. De forma a atingir os objectivos acima enunciados, a aplicação permite o anúncio de sessões (reuniões de trabalho) através da infra-estrutura de comunicação (ex. rede local sem fios) de suporte à aplicação, inclui a possibilidade do utilizador se associar a uma determinada reunião/sessão anunciada e permite a transferência automática dos ficheiros *PowerPoint* que vão ser apresentados na reunião. Desta forma, a aplicação desenvolvida possibilita a realização de reuniões de trabalho em qualquer lugar sem a necessidade de um *datashow* e apresenta-se como uma ferramenta simples e confortável para efectuar apresentações em ambientes de trabalho interactivos, por exemplo, reuniões de trabalho. Além disso, através dela torna-se possível a gestão do fluxo de uma reunião com base num modelo em que existe um coordenador que decide em cada momento quem toma a posição de apresentador/orador e, dessa forma, qual a apresentação a ser partilhada entre os intervenientes. O uso da aplicação desenvolvida permite ainda, em contraste com o sistema de *datashow*, melhor visualização de pequenos detalhes presentes nos slides, por exemplo, a legenda de uma figura.

2 – Trabalho Relacionado

Embora não haja nenhuma aplicação cuja finalidade se enquadre plenamente dentro dos objectivos definidos para este trabalho existem alguns programas de partilha de aplicações ou de *e-learning*[1] através dos quais é possível fazer-se uma apresentação de *slides* distribuída. Neste capítulo são apresentadas duas dessas aplicações, sendo primeiro descrito o protocolo no qual se basearam para permitir a partilha de aplicações.

2.1– Virtual Network Computing

O *Virtual Network Computing* (VNC), ou Computação em Rede Virtual, é um sistema de ecrã remoto que permite exportar a imagem de um Ambiente de Trabalho (*Desktop*), não apenas no computador onde está a ser executado, mas também em qualquer outro computador ligado ao primeiro com base numa rede de comunicação (ex. rede local, Internet) e suporta uma variedade de arquitecturas de computador. O VNC obedece ao modelo servidor-cliente, sendo que neste caso o servidor é o computador que aloja o ambiente de trabalho e o cliente é quem acede ao mesmo. Este sistema baseia-se na transmissão de sinais do teclado e do rato do cliente para o servidor e as actualizações visíveis no monitor do servidor para o cliente através duma rede de comunicação. Para além disso, o protocolo VNC baseia-se no conceito de *remote framebuffer* (RFB), daí ser muitas vezes designado por protocolo RFB. O protocolo gere a autorização do servidor

---

[1] *E-learning* compreende todas as formas de formação/ensino através de plataformas que utilizem a Internet como meio de comunicação entre os formandos/alunos e o formador/professor.

para actualizar o *framebuffer*[2] visto por um utilizador. Como este protocolo funciona ao nível do *framebuffer* tem potencial para ser aplicado ao nível de todos os sistemas operativos e aplicações. O método de operação do protocolo assenta em qualquer método fiável de transporte, por exemplo, TCP/IP.

O subsistema que lida com o ecrã baseia-se no princípio básico de cópia de um rectângulo com dados de pixel. O VNC utiliza vários esquemas de codificação, seleccionando o esquema mais adequado para cada rectângulo enviado, explorando o melhor possível a largura de banda, a velocidade de desenho do cliente e a velocidade de processamento do servidor.

Uma sequência de rectângulos constitui uma actualização de *framebuffer*. Estas actualizações são efectuadas a pedido do cliente, o que dá uma qualidade adaptativa ao protocolo. Quanto mais lentos forem o servidor e a rede, mais lento será também o processo de actualização.

Na Figura 1 é visível um *screenshot* dum exemplo de utilização do VNC: aceder ao ambiente de trabalho de um computador em Windows através de um computador em Linux.

---

[2] O *framebuffer* é um dispositivo de saída de vídeo que constrói a imagem no ecrã a partir de um *buffer* da memória que contém um quadro (*frame*) completo de dados.

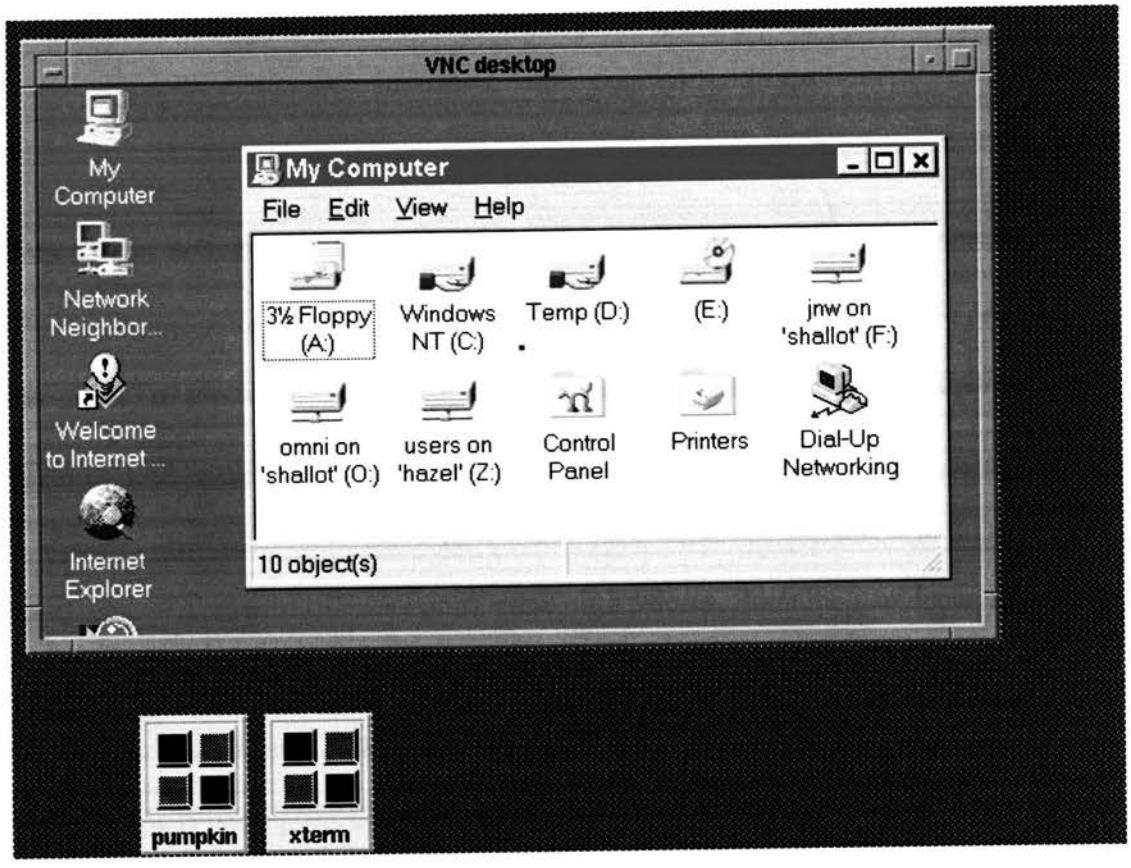

Figura 1 – Um Desktop de Windows visto em Linux com VNC

O VNC, por si só, não poderia ser usado para fazer as apresentações de *PowerPoint* numa reunião de trabalho. O protocolo permite apenas que um ecrã gráfico (neste caso, um ambiente de trabalho) seja visto noutra máquina. Na máquina onde esse ambiente está alojado não é possível visualizar a mesma informação a menos que o mesmo utilizador se ligue nos dois computadores. No entanto existem vários programas de partilha de aplicações que se basearam neste protocolo através dos quais é possível exportar o ecrã para outros computadores. Nas secções 2.2 e 2.3 apresentam-se dois exemplos.

2.2 – MSN Messenger Application Sharing

O MSN Messenger é actualmente o programa de *Instant Messaging*[3] mais usado em todo o mundo. Por meio de alterações sucessivas ao longo dos últimos anos este programa deixou de ser uma simples aplicação de mensagens de texto em tempo real. Através da integração de outras funcionalidades, tais como a capacidade de envio de voz e vídeo sobre o protocolo IP, aumentou-se o leque de aplicações de suporte à comunicação entre utilizadores através da Internet.

Um dos aspectos considerados pelo MSN Messenger é o suporte à partilha de aplicações (ex. *Whiteboard*, *PowerPoint*) entre utilizadores. Embora não haja muita informação disponível acerca do protocolo utilizado na partilha de aplicações, sabe-se que este é baseado no sistema VNC, pois o princípio de funcionamento é a exportação da área visível ocupada pela aplicação que está a ser partilhada.

Encontra-se representada a partilha de uma apresentação de *slides* através do MSN Messenger na Figura 2.

---

[3] *Instant Messaging* é uma forma de comunicação electrónica que envolve correspondência imediata entre dois ou mais utilizadores que se encontram ligados simultaneamente.

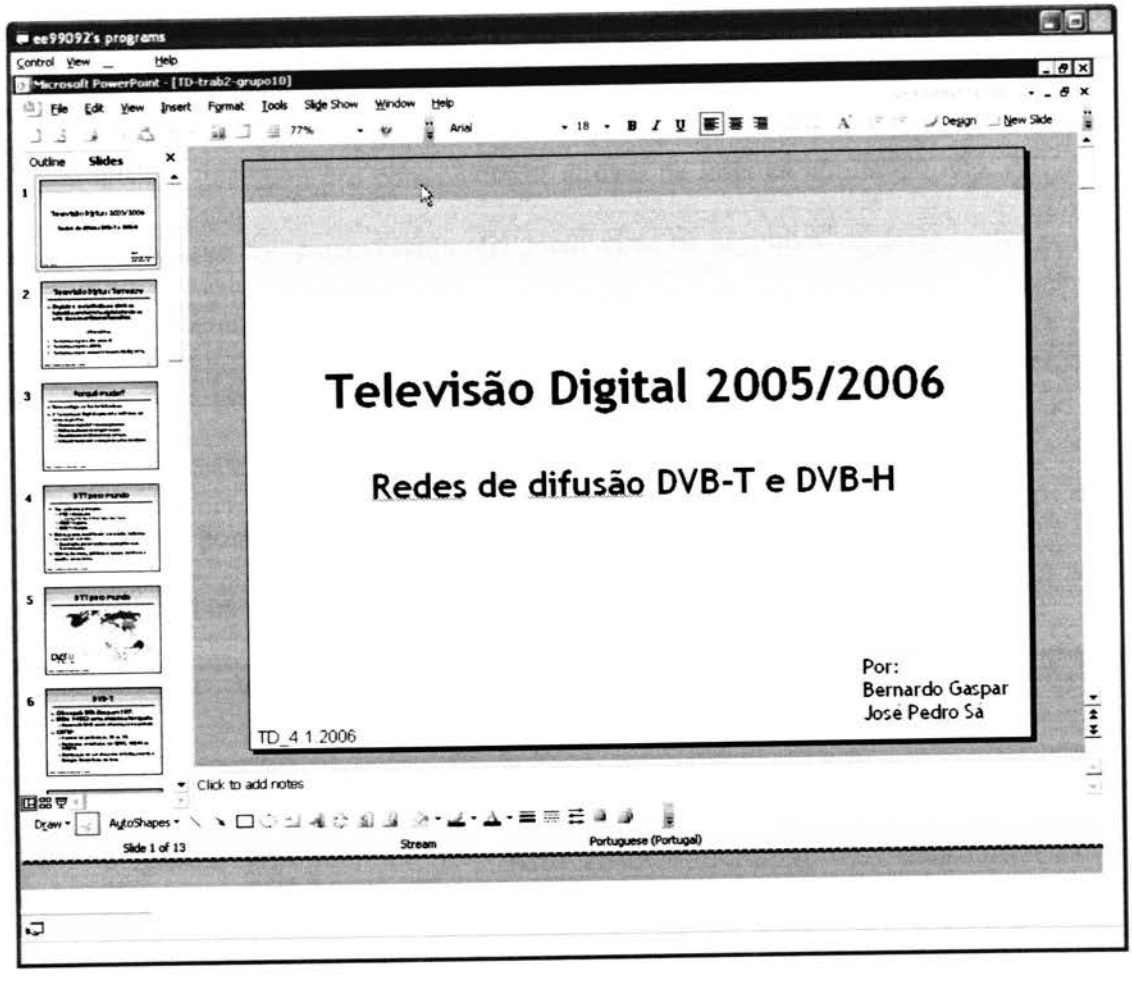

Figura 2 – Partilha de uma apresentação de *PowerPoint*

com o MSN Messenger Application Sharing

Virtualmente qualquer programa pode ser partilhado por duas pessoas através desta aplicação. Através desta funcionalidade do MSN Messenger torna-se possível realizar uma apresentação de slides *PowerPoint*, por exemplo, sendo que a principal desvantagem é o facto de o número de intervenientes estar limitado a dois.

2.3 – Centra Live

O Centra Live permite que empregados, clientes ou parceiros globalmente dispersos aprendam e interajam uns com os outros através da Internet em tempo real. A interface do Centra Live foi desenvolvida tendo em vista os seguintes cenários: aulas virtuais, eMeetings e seminários Web.

O Centra Live permite a difusão de áudio, vídeo e dados através da Internet. Integra ainda a capacidade de partilha de aplicações, nas quais se inclui o *PowerPoint*.

Na Figura 3 é possível observar o ambiente gráfico deste programa. A aplicação partilhada seria visível na caixa central da janela.

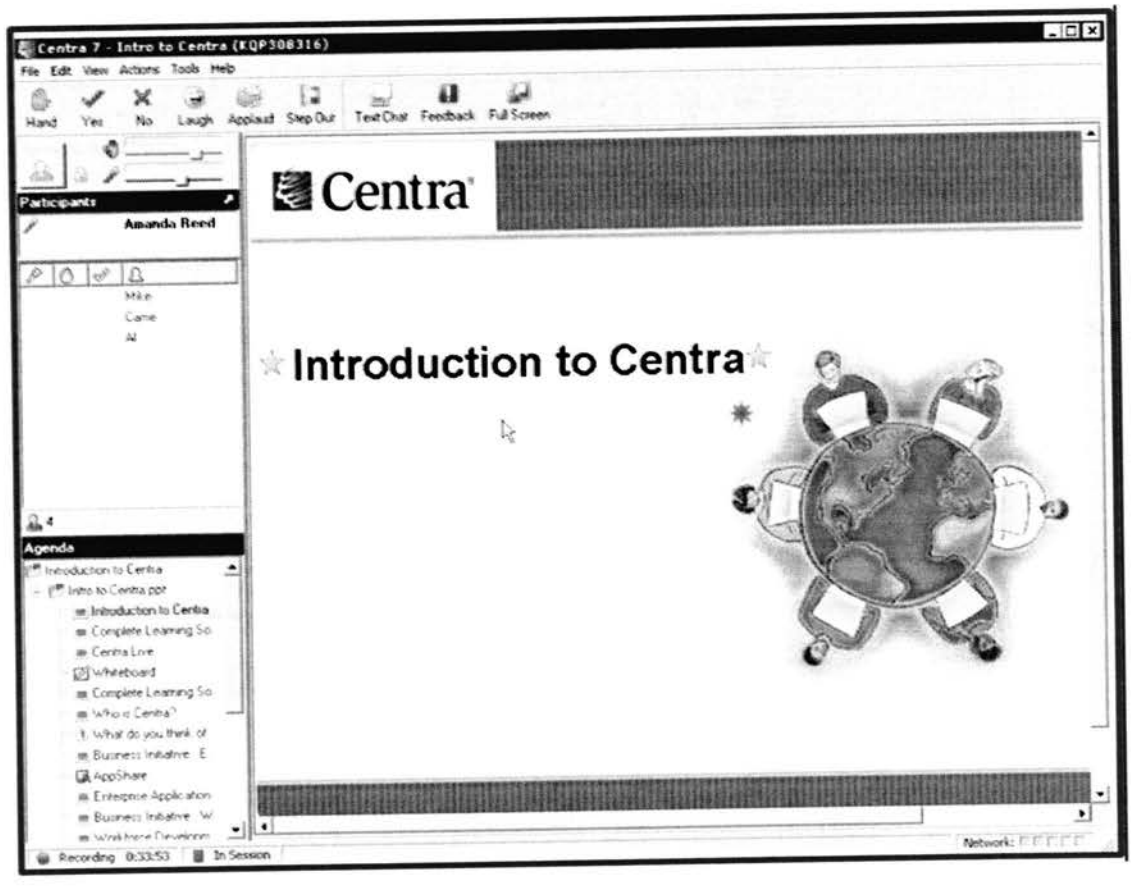

Figura 3 – *Display* gráfico do Centra

Através desta aplicação seria possível realizar reuniões de trabalho interactivas com apresentações baseadas em *PowerPoint* sem recorrer a um *datashow*. No entanto, o modelo adoptado por esta aplicação não é adequado para uma reunião de trabalho onde tipicamente os intervenientes se encontram no mesmo espaço físico. Como se trata de uma aplicação comercial não existe informação disponível relativa aos protocolos usados. No entanto, crê-se que é utilizado um protocolo baseado no mesmo princípio do sistema VNC, isto é, o envio de informação que é visível localmente através da rede de comunicação. Isto deve-se ao facto de o Centra Live suportar a partilha de uma grande variedade de aplicações, sendo o ecrã gráfico a única coisa que essas aplicações possuem em comum.

3 – Requisitos e especificação

Tal como o seu título indica, este capítulo é composto por duas secções. Na primeira secção são enumerados os requisitos da aplicação. É também feita uma descrição pormenorizada do cenário típico das reuniões de trabalho que se pretendem complementar através da aplicação desenvolvida. Na segunda secção encontram-se expostas as decisões tomadas para cumprir esses requisitos e a sua justificação.

3.1 – Requisitos

Com este trabalho pretende-se desenvolver uma aplicação para ser usada em reuniões de trabalho, pressupondo a partilha dos ficheiros das várias apresentações e que permita a visualização dos mesmos nos computadores (portáteis) dos diversos intervenientes na reunião, sendo o fluxo dessa conduzido por uma pessoa de cada vez a partir do seu próprio computador.

No cenário considerado as reuniões são locais e não remotas, ou seja, todos os utilizadores se encontram no mesmo local. Por esta razão, a aplicação não é responsável pela transmissão de voz e/ou vídeo. Consideramos a possibilidade de todos os participantes terem recebido os ficheiros com a apresentação *a priori*. Por esta razão a aplicação deve permitir a abertura de ficheiros locais para além da recepção de ficheiros remotos. Pressupõe-se ainda que os computadores portáteis dos participantes se encontram interligados atavés de uma rede local com ou sem fios.

Cada reunião é vista como uma sessão de pelo menos uma apresentação de *slides*. A aplicação possuirá os meios para anunciar sessões e os utilizadores deverão ter capacidade de as detectar e diferenciar. Assume-se que os utilizadores apenas participarão numa reunião num dado momento. A participação em reuniões simultâneas não é viável à luz do cenário contemplado, uma vez que as reuniões implicam a presença física dos participantes num local específico (i.e., sala ou anfiteatro).

Dada a especificidade da aplicação pretende-se usar uma abordagem diferente da das soluções existentes referidas no Capítulo 2. Como essas aplicações se baseiam na transmissão de imagens, a transição entre slides nas máquinas remotas pode-se tornar um pouco lenta. Para evitar essa desvantagem, pretende-se recorrer à transferência do ficheiro da apresentação na fase inicial. Esta técnica poupa largura de banda durante a fase de apresentação propriamente dita, uma vez que são apenas enviados comandos simples através da rede (ex., para cada máquina mudar o slide visível localmente) durante uma apresentação específica. Com isto pretende-se tornar a transição de slides mais fluida e com menor atraso em todos os computadores, melhorando assim a interactividade da reunião.

3.2 – Especificação

Nesta secção explicam-se com mais detalhe as decisões tomadas para cumprir os requisitos descritos na Secção 3.1 bem como as tecnologias utilizadas na solução.

Como foi referido nos requisitos (Secção 3.1), a aplicação desenvolvida deve permitir o anúncio e descoberta de sessões. No caso de grandes empresas ou conferências pode-se dar o caso de existirem várias sessões paralelamente. Cada sessão segue o modelo típico de uma reunião de trabalho, em que existe um orador e vários ouvintes. Não é invulgar existirem várias apresentações de pessoas distintas ou mesmo apresentações em conjunto numa reunião, pelo que a aplicação também deve suportar a mudança de orador. Por vezes as reuniões têm também um responsável pelo fluxo das mesmas. De forma a englobar todas estas possibilidades, foram considerados três tipos de intervenientes, cada um com um papel específico:

- **Orador** – faz uma apresentação na reunião. Partilha um ficheiro contendo os slides da apresentação e gere o fluxo da mesma.
- **Ouvinte** – visualiza a apresentação.
- **Coordenador** – inicia uma sessão e gere-a (i.e., transfere a permissão de orador de um utilizador para outro). Enquanto coordenador assume ainda um dos outros papéis, ou seja, pode também fazer uma apresentação ou simplesmente assistir à apresentação que esteja a decorrer.

Embora nem todas as sessões tenham ou precisem de um coordenador, a sua existência simplifica substancialmente o problema de mudança de orador. Esta mudança tanto pode estar definida *a priori* como ser discutida verbalmente durante a reunião e definida no momento. Havendo um coordenador, ou seja, um único utilizador com permissão para passar o testemunho (i.e., a autorização concedida a um utilizador para ser

orador), é possível evitar conflitos de mudanças de orador sucessivas realizadas por utilizadores diferentes.

Tendo em conta que o objectivo deste projecto era desenvolver um protótipo não se tornava essencial desenvolver uma aplicação que suportasse vários formatos de ficheiro e que fosse independente do sistema operativo. No entanto, a escolha do sistema operativo, do formato dos ficheiros das apresentações e a linguagem de programação a serem utilizados foi objecto de discussão e pesquisa. Tanto o sistema operativo como a linguagem de programação podem ser fortemente condicionados pelas aplicações e/ou formatos de ficheiros a suportar, daí ser necessário definir este ponto em primeiro lugar. As apresentações de slides podem ser realizadas em vários formatos. Foram considerados e analisados três formatos em particular: Microsoft *PowerPoint* (PPT), Adobe Acrobat (PDF) e HTML. O Microsoft *PowerPoint* acabou por ser o formato escolhido dado que é o principal formato usado no suporte a reuniões de trabalho.

Outro factor que influenciou a escolha deste tipo de ficheiros foi o facto de a Microsoft disponibilizar APIs que possibilitam a interacção com os seus programas, nomeadamente o Microsoft *PowerPoint*. Essas APIs estão preparadas para serem usadas por MFC[4] (Microsoft Foundation Classes) através de *OLE Automation*[5].

---

[4] MFC é uma biblioteca da Microsoft que engloba porções da API de Windows em classes C++. Estas classes permitem a manipulação de grande parte dos objectos do Windows e definem também várias janelas e controles comuns.

[5] *Object Linking and Embedding* (OLE) é um sistema de objectos distribuídos desenvolvido pela Microsoft, normalmente usado com MFC ou Visual Basic.

O OLE é geralmente utilizado para gerir documentos compostos, ou seja, para embeber vários tipos de média num ficheiro contendo outro tipo de média (i.e., ficheiros de texto contendo outros elementos multimédia como folhas de cálculo, gráficos, vídeos, etc.). Pode também ser usado para transferir dados entre diferentes aplicações. O OLE possui uma extensão que permite manipulação automática de documentos, tipicamente do Microsoft Office, denominada *OLE Automation*. Com *OLE Automation* é possível desenvolver programas que abram ficheiros do Microsoft Office e os edite automaticamente. As funcionalidades do OLE incluem, portanto, a possibilidade de abrir ficheiros de *PowerPoint* simplesmente com o intuito de os visualizar.

Pelos motivos apresentados, a linguagem escolhida para desenvolver a aplicação foi MFC devido às facilidades e funcionalidades que apresenta na interacção com os ficheiros de *PowerPoint*. Tendo em conta o formato dos ficheiros e a linguagem escolhida, a aplicação deveria ser desenvolvida em ambiente Microsoft Windows. Era possível usar os mesmos recursos em Linux mas existe um maior suporte para MFC e *PowerPoint* em Windows e, devido ao facto de ser mais *user friendly*, este sistema operativo é geralmente mais usado pelo utilizador comum.

De modo a minimizar o tráfego na rede, optou-se por fazer a difusão de dados através de IP Multicast.

O IP Multicast é um método que permite enviar uma única mensagem simultaneamente para vários computadores. As mensagens Multicast são enviadas para um endereço contido numa gama reservada para grupos particulares (neste caso ente 224.0.0.0 – 239.255.255.255). De modo a poder receber as mensagens dum dado grupo

Multicast a aplicação apenas necessita de se associar ao endereço desse grupo. Todas as mensagens enviadas para um grupo são posteriormente recebidas nas cartas de rede dos respectivos computadores que estejam associados ao grupo. O uso de Multicast faz entrega de informação de uma forma mais eficiente a um grupo de destinatários pois apenas existe um pacote a circular em cada *link* da rede, em vez de um pacote por cada destinatário.

No nosso caso específico pretendemos utilizar IP Multicast em redes *Wireless*, e, num cenário em que estão 5 pessoas numa reunião todas ligadas ao mesmo AP, por cada mensagem enviada (i.e., mudança de slide) circula um pacote na rede em vez de 4.

Tendo em conta a possibilidade de existência de várias reuniões em simultâneo é necessário providenciar um mecanismo que permita ao utilizador descobrir as sessões que estão activas, assim como um mecanismo de associação à reunião que lhe interessa. Foi então decidido que um endereço Multicast específico ficaria reservado para anúncio de sessões e que cada reunião/sessão ocorreria num outro endereço Multicast.

Em qualquer aplicação de rede é necessário definir um protocolo de comunicação e o formato das mensagens a usar. A necessidade de anúncio de reuniões/sessões não é excepção. Da pesquisa que foi realizada sobre soluções já existentes capazes de tratar este assunto, identificou-se o *Session Description Protocol* (SDP), como um forte candidato para tal. O SDP, ou Protocolo de Descrição de Sessão, foi concebido para facilitar o início de sessões multimédia. Para as restantes mensagens de rede decidiu-se desenvolver um protocolo em que as mensagens são formatadas de acordo com as regras de *eXtensible Markup Language* (XML).

Ao iniciar sessões multimédia é necessário definir detalhes relativos aos elementos média, nomeadamente endereços de transporte e ainda metadados da descrição da sessão. O SDP fornece um formato standard para essa informação independentemente do modo como ela é transportada. Contudo, o SDP não define protocolos de transporte pois as sessões podem usar protocolos diferentes de acordo com os cenários de utilização. O SDP começou como um componente de *Session Announcement Protocol* (SAP), ou Protocolo de Anúncio de Sessão, mas rapidamente se estendeu o seu uso ao *Real Time Streaming Protocol* (RSTP), ou Protocolo de *Streaming* em Tempo Real, e ao *Session Initiation Protocol* (SIP), ou Protocolo de Início de Sessão. Pode também ser usado como formato único (*standalone format*) para descrever sessões de media em Multicast. É neste último contexto que SDP apresenta maior utilidade na aplicação que se pretende desenvolver. Este protocolo permite que as mensagens de anúncio de sessão sejam criadas obedecendo a regras *standard*. Algumas das informações obrigatórias neste protocolo incluem o nome do criador da sessão, o endereço IP e porto onde esta se vai realizar, um número de identificação e o tipo de dados enviados durante a sessão. Todas estas informações são úteis na identificação de sessões e poderão ser usadas posteriormente.

O XML é uma linguagem de alto nível (i.e., "*human readable*"), destinada à criação de outras linguagens de *Markup* capazes de descrever diferentes tipos de dados. O objectivo desta linguagem é providenciar a partilha de dados entre diferentes sistemas. Permite ainda que os programas modifiquem e validem documentos nestas linguagens sem conhecimento *a priori* da sua forma. Devido à sua popularidade também é fácil encontrar bibliotecas de *parsers* para incluir na aplicação. Com esta linguagem, a

tarefa de criar e descodificar as mensagens enviadas pela rede fica substancialmente simplificada.

Estando já decidido que o tráfego de rede de cada sessão seria realizado em Multicast restava saber como seria efectuada a partilha dos documentos contendo as apresentações. Um dos protocolos de transferência de ficheiros em redes de computadores mais usados é o *File Transfer Protocol* (FTP), ou Protocolo de Transferência de Ficheiros. Uma possibilidade de tirar partido deste protocolo na solução a desenvolver, seria incorporar um cliente e um servidor FTP em cada aplicação, até porque é fácil encontrar implementações de servidores FTP facilmente adaptáveis à nossa solução. No entanto, com esta solução as comunicações seriam feitas ponto-a-ponto e não ponto-a-multiponto. Após alguma pesquisa descobriu-se que tinham sido realizados diversos estudos tendo em vista a transferência de ficheiros através de IP Multicast. Uma das conclusões a que se chegou foi que as aplicações de comunicações em grupo têm um leque de requisitos mais alargado do que as aplicações ponto-a-ponto. Uma delas é o não ser possível obter uma solução do tipo TCP para transferir dados por IP Multicast pois um único protocolo de transferência de dados em Multicast não poderia responder aos requisitos de todos os tipos de aplicações Multicast. Assim, surgiram vários protocolos de Multicast com características diferentes, tendo os mais recentes sido desenvolvidos com uma aplicação específica em vista.

Procedeu-se posteriormente à escolha de um protocolo adequado, dentre os existentes. O mais apelativo foi o *StarBurst* Multicast *File Transfer Protocol* (MFTP), ou Protocolo de Transferência de Ficheiros em Multicast, devido às suas características. No

entanto, este protocolo é uma solução comercial (actualmente desenvolvido pela StarBurst em associação com a Cisco), não havendo sido encontradas nenhumas implementações gratuitas.

Descobriu-se entretanto o programa *UDP based File Transfer Protocol* (UFTP), ou Protocolo de Transferência de Ficheiros baseado em UDP, que utiliza um protocolo baseado no *StarBurst* MFTP mas cuja distribuição é gratuita. Esta aplicação foi concebida com o intuito de transferir ficheiros para múltiplos receptores simultaneamente de uma forma eficiente e fiável. Consegue isto através dum sistema de mensagens que implementa a retransmissão dos pacotes perdidos. O funcionamento deste protocolo encontra-se descrito na Secção 4.2.

4 – Implementação

A aplicação foi construída com base em 3 módulos principais: o módulo que compreende a interface gráfica, o módulo que lida com as comunicações por rede e o módulo que gere o processamento de dados, implementa grande parte das funções e faz a ligação entre os outros dois módulos.

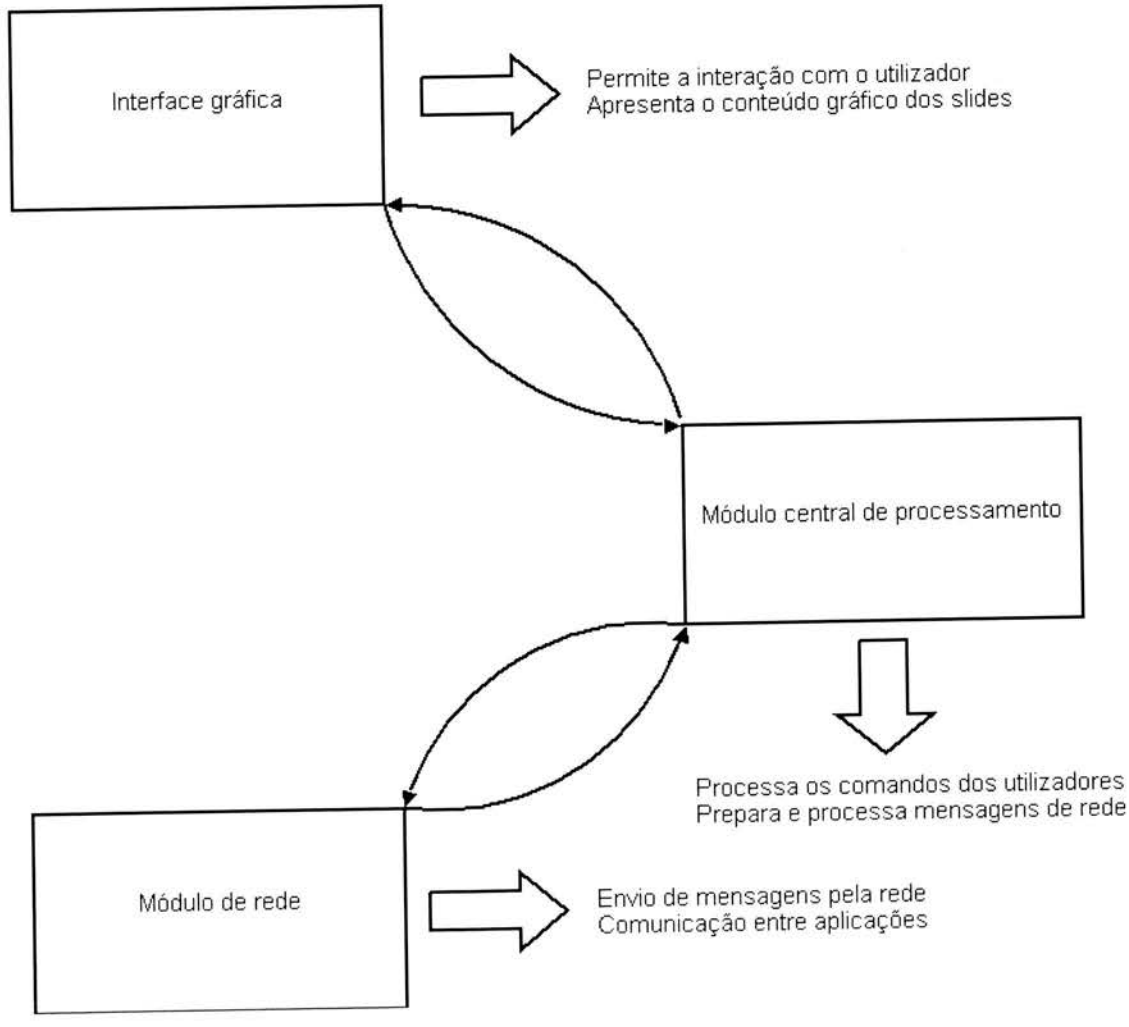

Figura 4 – Funções dos módulos e sua hierarquia

4.1 – Interface Gráfica

A classe CDstSldShowDlg apresenta uma interface gráfica ao utilizador. Possui também instâncias de subclasses que implementam caixas de diálogo auxiliares. Através delas é possível criar uma nova sessão, seleccionar uma sessão já existente, escolher ficheiros ou directórios, entre outras interacções úteis para o utilizador e típicas de ambientes Windows.

Ao iniciar a aplicação surge um diálogo simples onde o utilizador escolhe se deseja iniciar uma sessão ou juntar-se a uma já existente.

Se o utilizador pretende iniciar uma sessão surge um diálogo (Figura 5) para preencher os campos com os parâmetros necessários para a iniciação e identificação da sessão. Estes campos incluem o nome da sessão, o nome de utilizador e o directório onde os ficheiros recebidos serão guardados. Neste diálogo é visível o endereço IP e o porto onde a sessão irá decorrer, estando estes parâmetros disponíveis para edição por parte do utilizador.

Caso a intenção seja a de se juntar a uma sessão existente surge um segundo diálogo (Figura 6) onde é apresentado ao utilizador uma lista das sessões existentes e que a aplicação automaticamente identifica por escutar o endereço Multicast pré-definido para anúncio de sessões. Neste diálogo o utilizador apenas precisa de indicar qual a pasta onde pretende guardar os ficheiros recebidos e o seu nome de utilizador pelo qual será identificado pelos outros utilizadores na sessão.

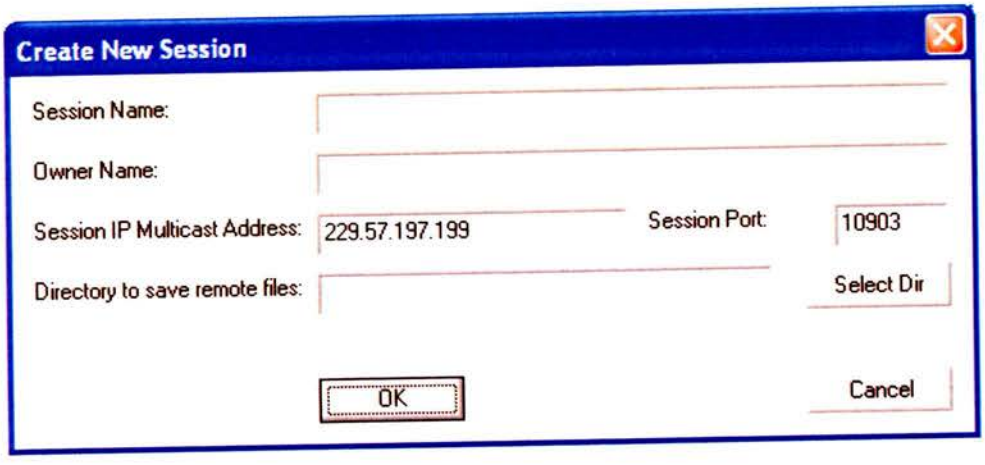

Figura 5 – Diálogo de criação de sessão

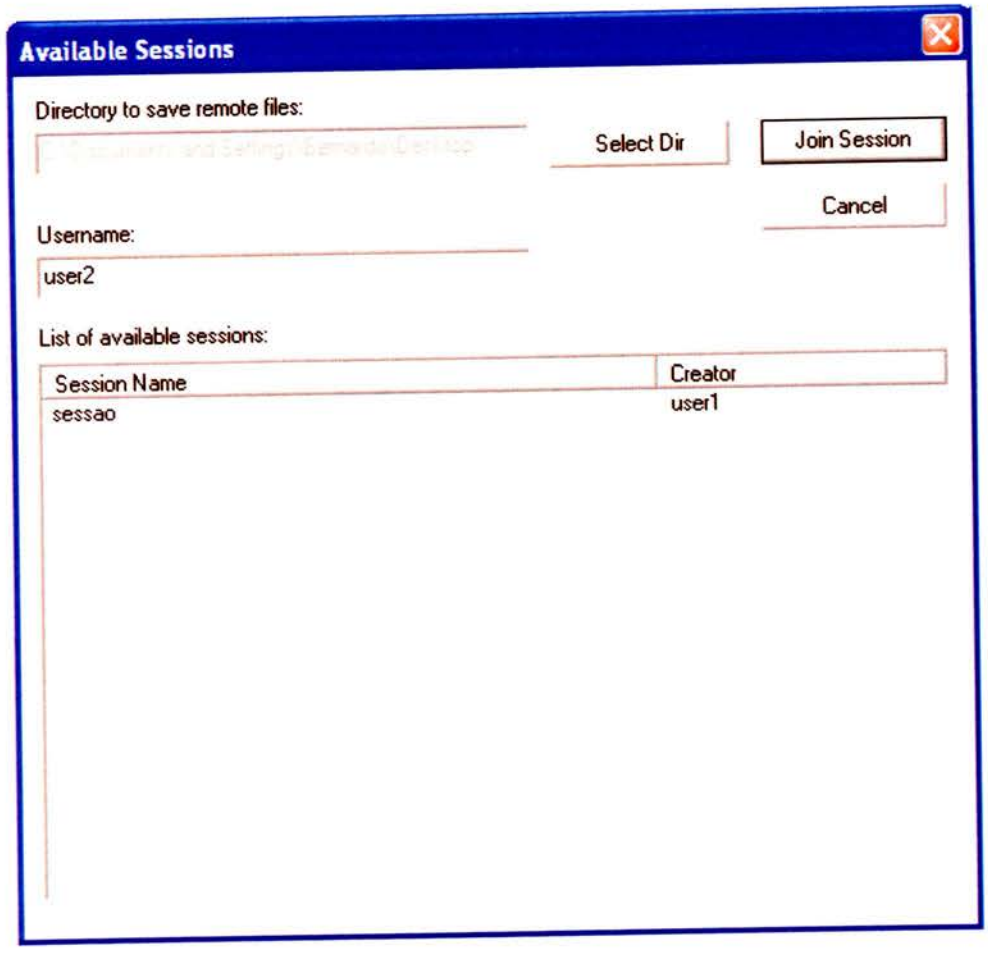

Figura 6- Diálogo de selecção de sessão

O diálogo principal da aplicação permite a transferência e abertura de ficheiros *PowerPoint*, o visionamento de uma apresentação de slides, o controlo do fluxo dessa apresentação e também operações de gestão da sessão (transferência do papel de orador ou exclusão de um dos participantes). Ou seja, a sessão em si é completamente gerida e visível através deste diálogo.

A interacção é feita através de controles para cada uma das operações. Cada controlo está associado a um método da classe CCore, que leva a cabo a tarefa pretendida.

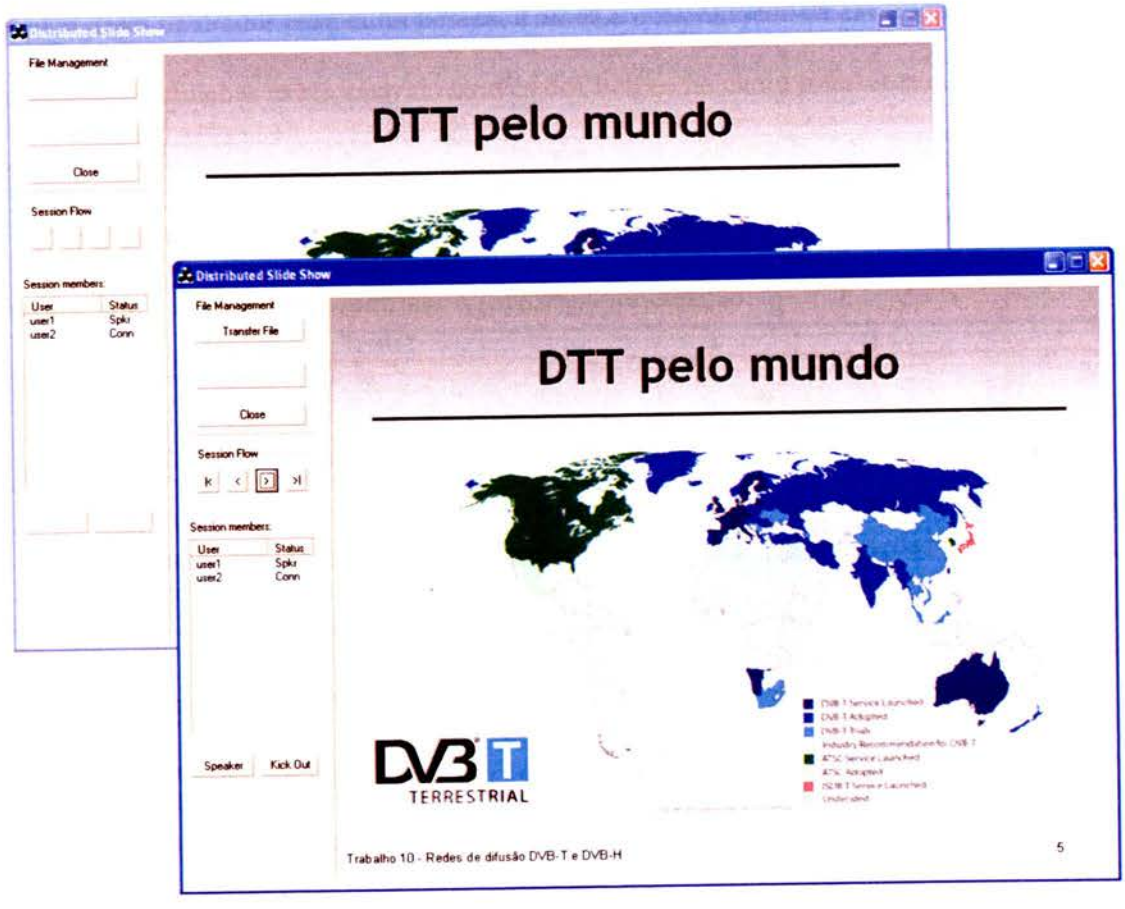

Figura 7 – Interface do diálogo principal

Na Figura 7 está representada a interface gráfica disponível aos utilizadores, com uma apresentação aberta. Esta interface foi desenvolvida pensando numa resolução de 1024x768 pixels. O conteúdo da apresentação é visualizado na área predominante da janela, estando do lado esquerdo uma barra com botões de controlo e informação relativa aos participantes na reunião. Todos os utilizadores têm acesso aos botões de abrir e fechar apresentação.

A aplicação que se encontra em primeiro plano está a ser utilizada por um coordenador no papel de orador. O que diferencia o coordenador dos restantes participantes da reunião é o facto de poder fazer a "passagem de testemunho", isto é, transferir o papel de orador para outra pessoa, e ainda a possibilidade de expulsar pessoas da reunião. Essas funções são acessíveis através dos botões no canto inferior esquerdo da janela.

Enquanto orador, tem também a possibilidade de transferir ficheiros contendo os slides e também a faculdade de controlar o fluxo da apresentação.

4.2 – Módulo de Rede

A ideia inicial era ter uma só classe que gerisse todas as comunicações por rede. No entanto, optou-se por criar várias classes para lidar com os eventos de rede:

- **CRede** – envio de mensagens por rede nos grupos Multicast de anúncio de sessão e de sessão
- **CUFTP** – envio de ficheiros através de UFTP
- **CUFTPReceiver** – recepção de ficheiros através de UFTP

Esta decisão foi tomada de modo a simplificar alguns problemas de ordem técnica. Torna-se mais simples identificar qual a proveniência das mensagens de rede e facilita adaptar o código base do UFTP a uma classe global.

A classe CRede é baseada na classe CAsyncSocket das bibliotecas da Microsoft, a qual disponibiliza uma API para *sockets* assíncronos em Windows. Os *sockets* assíncronos apresentam a vantagem, em relação aos tradicionais *sockets* de Berkeley, de não bloquearem enquanto esperam a chegada de dados pela rede. Permitem que o programa continue a executar instruções e quando chega alguma mensagem de rede, regista a ocorrência do evento, sendo nesse momento processados os dados recebidos. A classe CRede adapta este tipo de *sockets* para uso em grupos Multicast.

Cada instância desta classe cria um par de *sockets* assíncronos, de modo a poder-se usar este tipo de *sockets* em comunicações Multicast. Um desses *sockets* é associado a um grupo Multicast, recebendo assim as mensagens difundidas para esse endereço. Esta associação não é suportada directamente pela API de *sockets* assíncronos tendo sido necessário criar um método próprio que substitui a inicialização da classe CAsyncSocket. O segundo *socket* é utilizado para enviar mensagens para o grupo Multicast, tratando-se de um *socket* assíncrono comum.

Para além dos métodos de inicialização dos *sockets*, esta classe contém ainda métodos para enviar mensagens para o grupo Multicast, receber mensagens desse grupo e também dissociar-se do grupo. Estas funções são implementadas através da API de *sockets* assíncronos.

Esta classe é usada na aplicação para o utilizador se associar aos grupos Multicast de anúncio e de sessão, sendo necessário fazer duas instâncias distintas na aplicação (uma para cada grupo: uma para receber o anúncio e outra para a reunião/sessão).

Adoptou-se o endereço IP de grupo Multicast 229.123.123.1 e o porto UDP 10000. Caso a aplicação esteja a ser utilizada para dar início a uma sessão, uma mensagem contendo a descrição da sessão é enviada periodicamente para este endereço. Para os clientes que se pretendem juntar a uma sessão, a sua aplicação escuta as mensagens enviadas para este grupo Multicast apenas durante a fase de selecção da sessão.

O endereço IP onde a sessão efectivamente decorre é gerado aleatoriamente, podendo ocorrer na gama 229.x.x.x com o porto a variar entre 10000 e 50000. Todas as mensagens que constituem o protocolo da aplicação são enviadas para este endereço. Esta precaução é tomada para minimizar a probabilidade de ocorrerem várias sessões no mesmo endereço. No entanto, mesmo que tal se verificasse, dificilmente existiriam conflitos entre as sessões pois cada uma é identificada por um número aleatório.

As classes CUFTP e CUFTPReceiver foram baseadas em código já existente, como foi referido na Secção 3.2, que implementa o protocolo UFTP (FTP baseado em UDP). Este protocolo é constituído por 3 partes principais: a fase de anúncio/registo, a fase de transferência/NAKs[6] e a fase de finalização/confirmação.

Na primeira fase o servidor começa por enviar anúncios num endereço Multicast público, no qual os clientes devem estar a escutar. A mensagem de anúncio contém vários

[6] Non Acknowledgements – indicação de que não se recebeu uma mensagem esperada

detalhes sobre o ficheiro e o endereço Multicast "privado" no qual será transmitido o ficheiro do orador para os restantes utilizadores. Quando um cliente recebe um anúncio, na fase de associação, envia uma mensagem de registo para o servidor que responde com uma mensagem de confirmação. O servidor envia mensagens de anúncio durante um intervalo de tempo, começando depois a transmitir o ficheiro.

Na fase seguinte, a fase de transferência/NAKs, o servidor envia continuamente os pacotes com os dados do ficheiro. Contudo, uma vez que o protocolo UDP não garante a entrega ordenada de pacotes, cada pacote vai numerado ficando a cargo do cliente guardar a informação pela ordem certa. Quando todo o ficheiro foi enviado, o servidor transmite uma mensagem indicando que a transferência está terminada e aguarda uma mensagem de estado de cada cliente. Essa mensagem de estado contém a lista de pacotes que chegaram correctamente ao destino. Caso o ficheiro não tenha chegado completo a um ou mais clientes, o servidor reenvia os pacotes em falta. Este processo repete-se até que todos os clientes tenham recebido o ficheiro ou até um limite de tempo pré-configurado em que o servidor aguarda pela mensagem de estado.

Quando um cliente acaba de receber um ficheiro envia um sinal especial na mensagem indicando este facto, entrando assim na fase de finalização/confirmação. Quando o servidor recebe esta mensagem envia uma confirmação para esse cliente. Ao receber essa confirmação, o cliente termina o processo que está a receber o ficheiro. Esta fase não é simultânea para todos os clientes pois os clientes enviam o sinal de terminação à medida que vão recebendo o ficheiro, podendo estar mais clientes a receber alguns pacotes perdidos.

Na aplicação desenvolvida, os anúncios são efectuados no mesmo endereço de sessão mas utiliza-se o porto seguinte. O endereço privado onde é efectuada a transferência do ficheiro é gerado aleatoriamente na gama 230.5.5.x. Esta gama de endereços privados já vinha definida no protocolo UFTP original. Optou-se por não a alterar essa gama, embora o UFTP suportasse configuração manual do endereço privado a usar.

4.3 – Módulo Central

A classe CCore é o núcleo da aplicação. Esta classe faz a ligação entre a interface gráfica e a rede: implementa as funções disponíveis ao utilizador através da interface, prepara as mensagens a enviar e processa as mensagens provenientes da rede. Possui também métodos para controlar as apresentações de slides e para gerir a reunião. Um caso típico do funcionamento desta classe como elemento mediador entre a interface gráfica e as mensagens de rede é na mudança de slide. O utilizador carrega no botão para avançar o slide e a interface gráfica invoca o método apropriado da classe CCore. Este método recolhe os dados necessários para formar o comando a enviar por rede e em seguida invoca o método da classe CRede que envia a mensagem para o grupo Multicast.

Na instância da aplicação que corre em cada um dos outros intervenientes na sessão, uma vez que estes não têm permissões de oradores, regista-se o processo contrário: ao receber uma mensagem proveniente da rede, a classe CCore recolhe-a com recurso à classe CRede e processa a mensagem. Ao identificá-la como sendo um

comando para mudança de slide, processa a respectiva mudança de slide que se torna então visível para o utilizador através da interface gráfica.

Para o lidar com as mensagens de anúncio de sessão, que são em formato SDP, foi incluída uma biblioteca para a qual foi construída uma *wrapper class*. Esta *wrapper class* é uma classe que, através dos seus métodos, permite a invocação de várias funcões dessa biblioteca. Optou-se por este procedimento pois o código encontrado para fazer o parsing de SDP não era orientado ao objecto. A referida biblioteca permite preencher os campos necessários para se obter uma mensagem no formato SDP bem como recuperar os campos pretendidos de forma simples. Neste último caso, a informação relevante é o endereço IP e o porto da sessão, o nome da sessão, o seu número de identificação e o nome do utilizador que a iniciou.

À excepção das mensagens de anúncio de sessão, todas as mensagens de rede estão em formato XML. Para proceder à construção e descodificação destas mensagens foi integrado na aplicação um *parser* de XML, o XMLite. Este *parser* apresenta uma interface através de uma classe adicionada ao módulo central da aplicação. Recorrendo a esta classe, os actos de construir mensagens ou lê-las tornam-se bastante mais simples.

As mensagens XML que a aplicação tem que conseguir descodificar são:

- **Alive** – mensagem enviada periodicamente que indica aos outros utilizadores que um dado cliente ainda se encontra activo
- **Flow** – mensagens que controlam o fluxo da apresentação (avançar ou recuar um slide, saltar para o início ou fim da apresentação)

- **Kick** – exclui o utilizador seleccionado da sessão
- **SpkrChange** – indicação proveniente do coordenador de mudança de orador
- **SpkrOK** – confirmação do novo orador de que recebeu a mensagem de mudança de orador
- **Repeated** – indicação de que já existe um utilizador com o mesmo nome
- **Transfer** – sinal de que o orador pretende enviar um ficheiro

Um dos requisitos da aplicação era a possibilidade de abrir ficheiros de *PowerPoint* dentro da aplicação após o estabelecimento de uma sessão. Depois da escolha do ficheiro, a aplicação executa vários métodos da classe CCore que vão abrir uma instância de *PowerPoint* usando os comandos disponíveis através da API de *OLE Automation*. Para que a instância seja aberta dentro desta aplicação e não numa janela à parte é necessário indicar ao *PowerPoint* que a sua janela-pai é uma caixa desenhada dentro do diálogo principal da aplicação. Todas as operações relacionadas com os ficheiros de *PowerPoint*, desde a abertura da apresentação à mudança de slide, são realizadas através da API disponibilizada pela Microsoft.

5 – Avaliação

5.1 – Metodologia

Foram realizados dois tipos de testes para proceder à avaliação da aplicação desenvolvida. O primeiro teste visa comparar a usabilidade e a performance da aplicação em termos visuais. O segundo teste consistiu na medição do tráfego de rede gerado pela aplicação desenvolvida. Ambos os testes foram também realizados para o MSN Messenger de modo a estabelecer um termo de comparação. Apenas se usou o MSN Messenger pois é a aplicação mais conhecida de uso gratuito de partilha de aplicações. Não foi possível realizar testes com a aplicação *Centra Live* uma vez que é uma aplicação comercial não disponível gratuitamente.

Para a realização dos testes foi criado um ficheiro de *PowerPoint* contendo 14 *slides*. Os slides estão dispostos da seguinte forma: 2 *slides* de imagens alternados com 2 *slides* de texto. Esta disposição serve para confirmar se o facto de se tratarem de transições entre slides de imagens ou entre slides de texto influencia a duração da mudança de *slide*. Prevê-se que no caso do MSN Messenger as transições contendo imagens demorem mais tempo pois, como a informação enviada por rede é uma imagem codificada, o tamanho da mensagem também será superior ao caso em que se use texto simples.

5.2 – Teste de Usabilidade e Performance

Neste teste filmaram-se duas apresentações de *slides* com a aplicação desenvolvida envolvendo dois utilizadores. Na primeira, o utilizador A foi o orador

havendo troca de papéis na segunda apresentação. Neste teste não foi considerada a transferência de ficheiro pois o que se pretende é medir os tempos de transição dos *slides*.

A mudança de slide foi efectuada com alguns segundos de intervalo de modo a que a transição fosse bem visível. Os tempos entre *slides* não foram completamente rigorosos pois apenas os instantes da transição importam para este teste.

Em seguida foi repetido o mesmo procedimento para o MSN Messenger.

Para medir os tempos utilizou-se o Windows Movie Maker para se poder contar o número de *frames* desde a instrução de mudança de *slide* até a transição estar completamente efectuada no ecrã do ouvinte. Foram medidos quatro instantes diferentes em número de *frames*, em relação à *frame* na qual se comanda a transição:

- t1 – instante em que se começa a notar a transição no computador do orador
- t2 – instante em que a transição no computador do orador está completa
- t3 – instante no qual se detecta o início da transição no computador do ouvinte
- t4 – instante em que a transição está completa

De realçar que o tempo de refrescamento do ecrã não é instantâneo, daí se medir a duração da transição em cada computador desde o momento em que esta começa a ser visível até estar completa.

Os números de *frames* contabilizados foram inseridos numa folha de cálculo e para cada caso da experiência foi calculada o número médio de *frames*. Sabendo que a *frame*rate é de 12,5 *frames* per second (fps) foi possível calcular o tempo médio

aproximado para cada instante relevante da transição. As folhas de cálculo encontram-se disponíveis no Anexo B.

A partir dos valores medidos fizeram-se 2 gráficos para cada apresentação.

O primeiro gráfico representa a diferença entre t4 e t2 para cada *frame*. Como esta diferença nos dá o número de *frames* decorridas entre o final da transição no orador e o final da transição no ouvinte, podemos dizer que este valor corresponde ao atraso introduzido pela aplicação na apresentação.

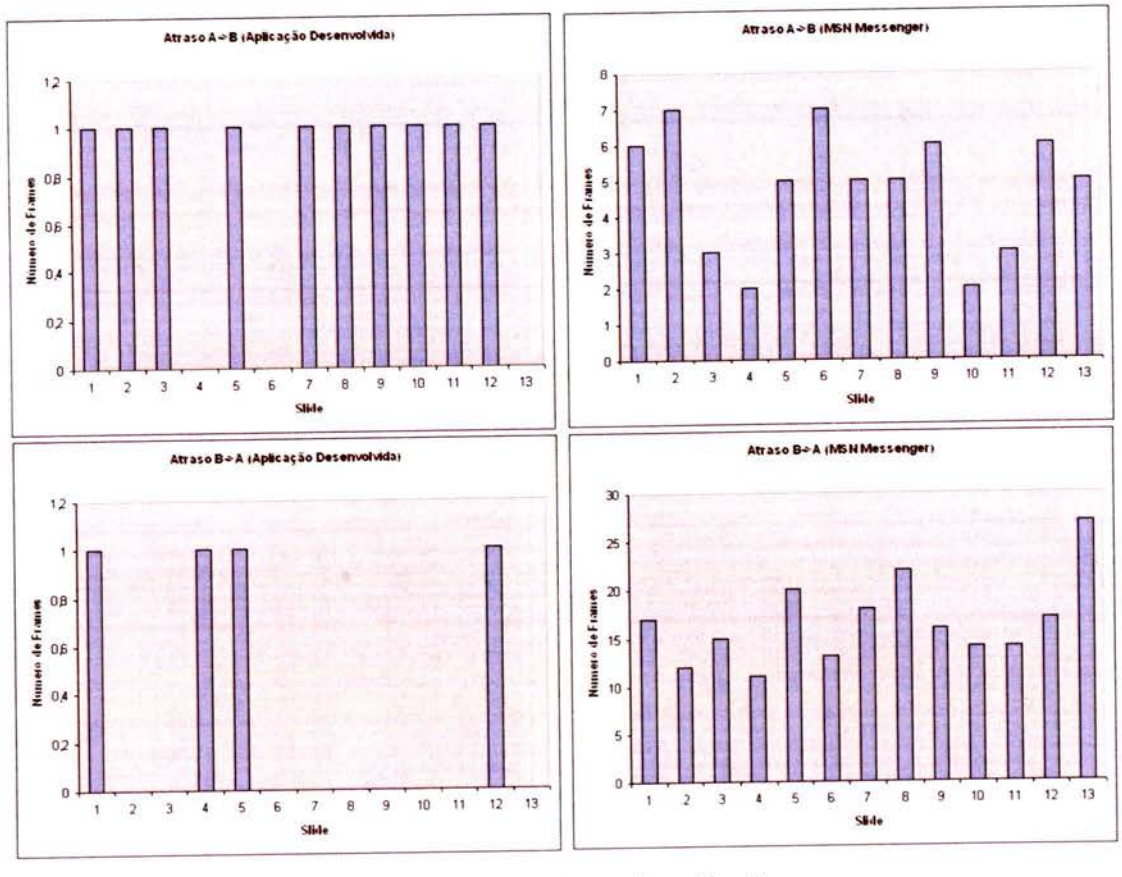

Figura 8 – Atrasos introduzidos pelas aplicações

Como se pode obervar na Figura 8, os atrasos introduzidos pelo MSN Messenger são substancialmente mais elevados. O pior caso chega a atingir 27 *frames* numa transição, enqanto que, com a aplicação desenvolvida durante este projecto, o atraso máximo introduzido é de apenas 1 *frame*, chegando mesmo a haver transições que aparentam ser simultâneas.

No segundo gráfico representou-se o número de *frames* que durou a transição no ouvinte. Este valor é obtido subtraindo t3 a t4 e representa a qualidade da transição.

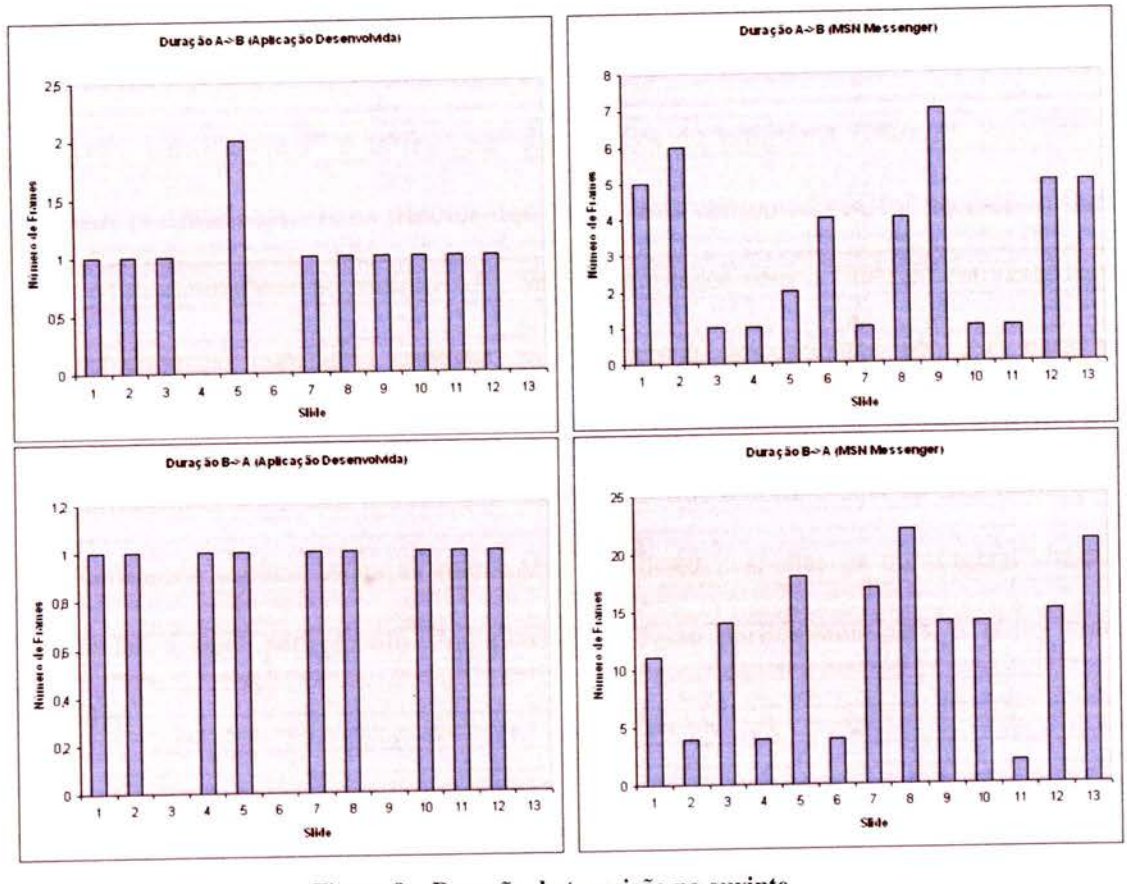

Figura 9 – Duração da transição no ouvinte

Em relação à duração da transição no ouvinte, registaram-se também valores mais altos para o MSN Messenger, atingindo um máximo de 22 *frames*, enquanto, que a aplicação desenvolvida, atinge as 2 *frames* apenas uma vez.

Em termos temporais, a aplicação demora em média menos de 1 décima de segundo a efectuar a transição. O MSN Messenger demora, em média, pouco mais de 1 segundo no teste de B para A e 0,38 segundos de A para B.

5.3 – Teste de Tráfego

Para realizar este teste recorreu-se ao programa *Ethereal* para medir o tráfego na rede. Usaram-se três computadores para o efeito: um como orador, outro como ouvinte e o terceiro para "escutar" o tráfego na rede através do programa *Ethereal*. Para que o *Ethereal* pudesse capturar os pacotes dos outros dois computadores foi necessário ligar todos os computadores ao mesmo AP[7]. Se o computador com o Ethereal estivesse num AP separado só conseguia capturar as mensagens descendentes (do AP para as máquinas). Ou seja, apenas as mensagens de Multicast eram capturadas. Caso apenas um dos participantes na sessão estivesse no mesmo AP que o computador de monitorização, este conseguia escutar todas as mensagens Multicast e apenas as mensagens Unicast destinadas a esse participante. As mensagens desse participante para o outro eram perdidas.

---

[7] *Access Point* – Pontos de acesso a uma rede sem fios. Cada computador liga-se a um AP que tem um funcionamento semelhante ao de um *switch* de rede.

Embora o protocolo desenvolvido para a aplicação assente unicamente em Multicast, o protocolo UFTP nela incorporado utiliza algumas mensagens de Unicast, nomeadamente nas confirmações e envio da lista de NAKs. De qualquer modo o MSN Messenger utiliza apenas Unicast, pelo que era sempre necessário conseguir-se escutar este tipo de mensagens para se poder proceder a uma comparação do tráfego utilizado.

Para se proceder à captura dos pacotes relevantes na rede, o *Ethereal* foi configurado para modo promíscuo e utilizou-se o filtro de pacotes adequado de modo a considerar apenas os pacotes referentes à aplicação em análise.

Este teste pretendia simular de uma forma mais realista o decorrer de uma apresentação, em relação ao primeiro teste efectuado. Por essa razão a transição de *slides* foi efectuada de 5 em 5 segundos e no final da apresentação recua-se rapidamente até ao *slide* do meio. Optou-se por este procedimento já que é usual no final das apresentações regressar atrás para explicar melhor algum ponto que não tenha sido bem compreendido pelos ouvintes. Nesta altura há uma sobrecarga de mensagens de mudança de slide e considerou-se de interesse comparar o tráfego nesta situação entre a aplicação desenvolvida e o MSN Messenger.

O teste foi efectuado três vezes para cada uma das aplicações. Da informação capturada pelo Ethereal extraiu-se o instante de ocorrência do pacote e o seu tamanho. Estes dados foram inseridos numa folha de cálculo e foram construídos dois tipos de gráficos para cada teste efectuado:

- um gráfico com o tráfego instantâneo

- um gráfico de barras em que o tráfego foi contabilizado em cada 5 segundos, estando visível o tráfego ocorrido nesse intervalo e o tráfego acumulado até esse intervalo

Os resultados encontram-se apresentados de seguida. As folhas de cálculo utilizadas para processar os dados obtidos não estão incluídas nos Anexos devido à sua extensão.

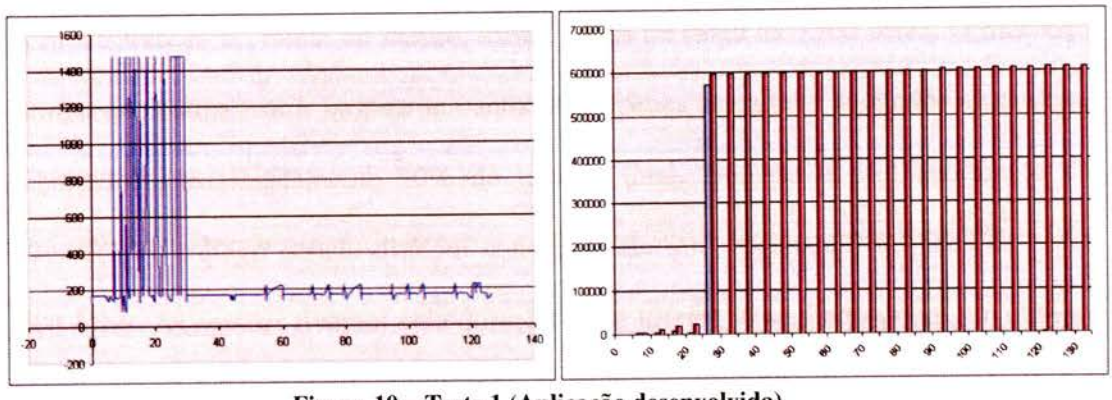

Figura 10 – Teste 1 (Aplicação desenvolvida)

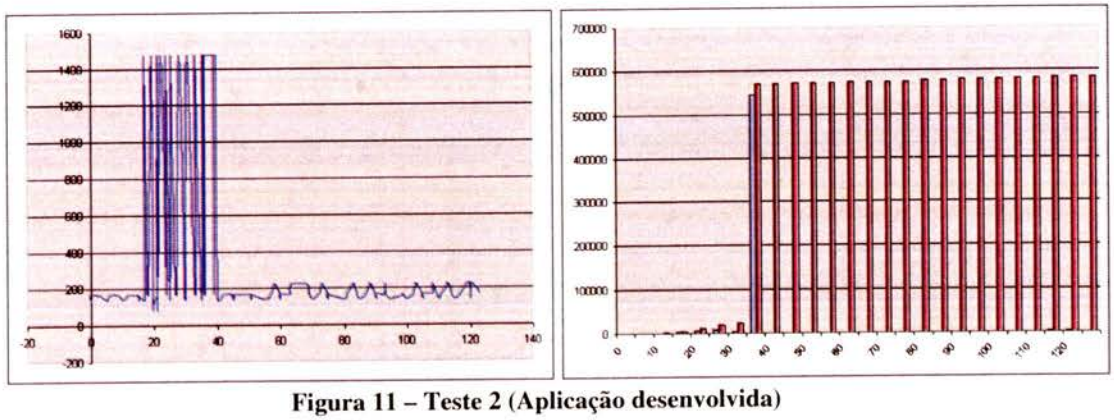

Figura 11 – Teste 2 (Aplicação desenvolvida)

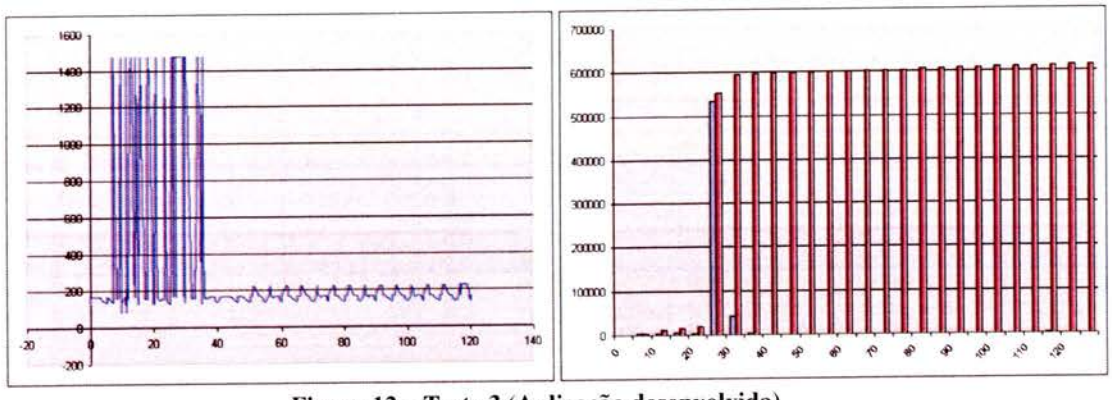

Figura 12 – Teste 3 (Aplicação desenvolvida)

Pode-se observar, a partir das figuras anteriores, que a aplicação desenvolvida gera muito tráfego no início da **sessão, atinge** picos de cerca de 1500 *bytes*. O intervalo de tempo que ocupa maior largura de banda dura apenas cerca de 5 segundos, estando aí concentrados aproximadamente 90% do tráfego total. Durante o prosseguimento da apresentação o tráfego gerado diminui consideravalmente, registando-se picos de cerca de 200 *bytes*. As sessões tiveram uma duração entre 120 e 130 segundos, com um tráfego total na ordem dos 600 KBs.

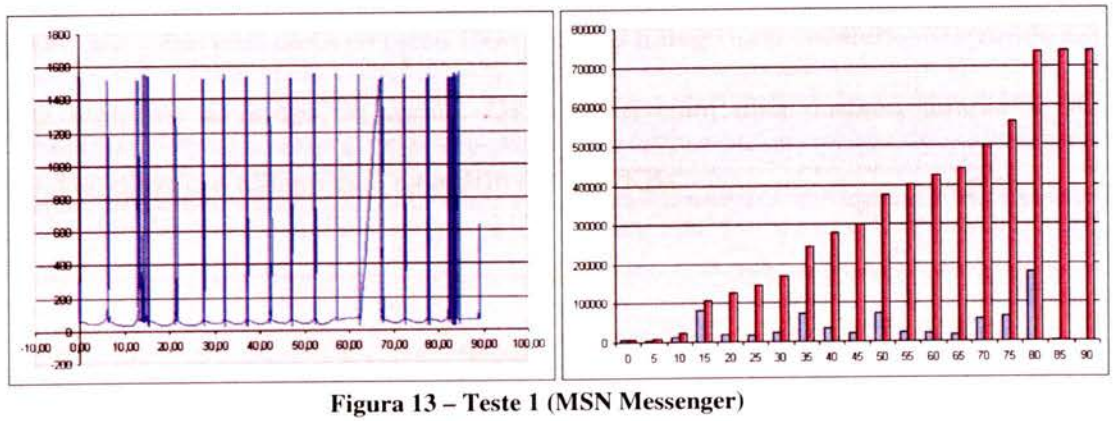

Figura 13 – Teste 1 (MSN Messenger)

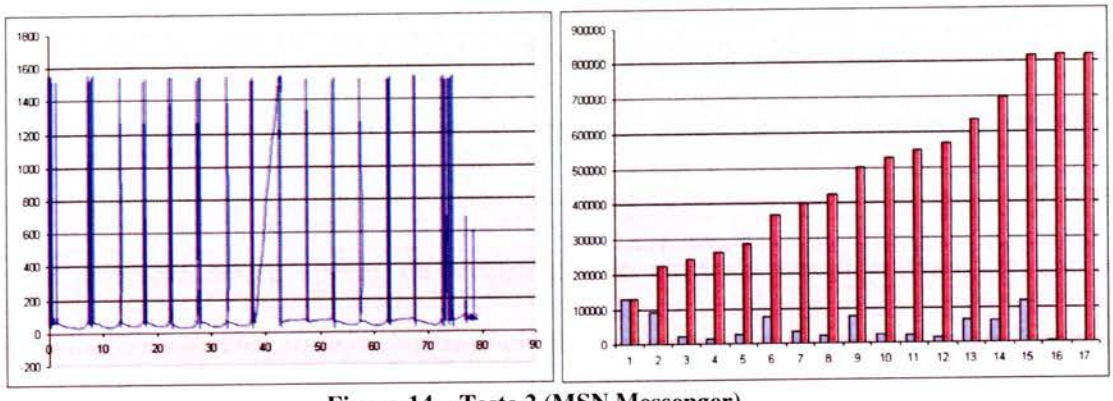

Figura 14 – Teste 2 (MSN Messenger)

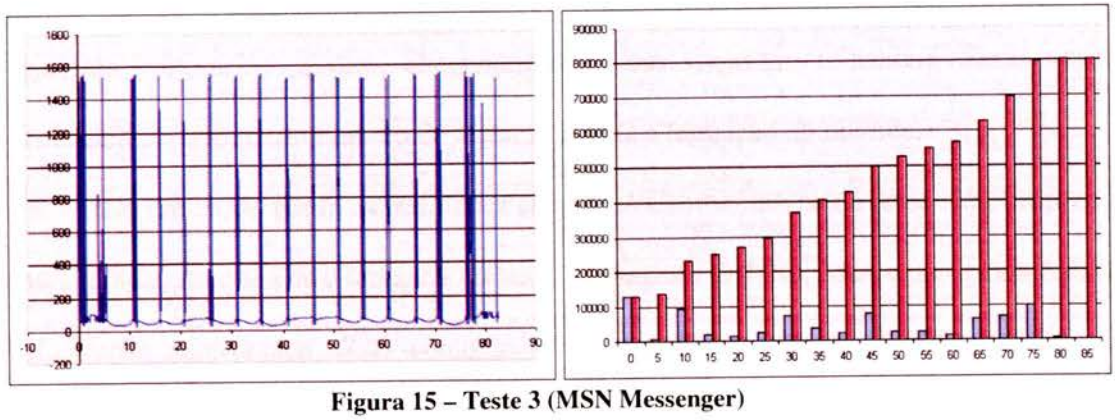

Figura 15 – Teste 3 (MSN Messenger)

Nas imagens anteriores estão apresentados os resultados obtidos para o MSN Messenger. Ao longo da sessão estabelecida através desta aplicação registam-se curtas rajadas, mas todas com picos de cerca 1500 *bytes*. O tráfego total encontra-se repartido de forma uniforme ao longo da sessão. Os testes tiveram uma duração inferior a 90 segundos, estando o tráfego total na ordem dos 800 KBs.

5.4 – Discussão dos Resultados

Analisando os resultados do primeiro teste efectuado é possível chegar à conclusão que as transições de *slides* são efectuadas muito mais rapidamente na aplicação desenvolvida do que no MSN Messenger. Isto deve-se ao facto de as mensagens enviadas pela aplicação serem apenas de sinalização de mudança de *slide*, uma vez que a informação acerca do *slide* já está disponível no próprio computador do ouvinte. No caso do MSN Messenger as quantidades de informação a enviar em cada transição são bastante superiores pois é necessário transmitir o próprio *slide*. A informação tem que ser repartida por vários pacotes, chegando assim em vagas ao utilizador. Essas vagas provocam o aumento do número de *frames* durante a transição no ouvinte.

De um modo geral, as transições entre *slides* consecutivos de texto simples foram mais rápidas do que entre imagens ou texto e imagem. O MSN Messenger envia toda a informação relativa aos *slides* como uma imagem. Provavelmente o MSN Messenger utiliza codificação diferencial, isto é, envia apenas as diferenças entre a imagem anterior e a actual. A imagem é descodificada no destinatário através da soma da imagem das diferenças e a construída anteriormente. A partilha de aplicações do Messenger deveria tirar partido desta propriedade pois é esperado que uma grande parte da imagem a enviar se mantenha estática na maioria dos programas partilhados. Nestas condições as diferenças relativas a essa parte da imagem são nulas, sendo enviada muito menos informação. De qualquer forma, os *slides* de texto são essencialmente brancos com algumas zonas a preto. Existem várias técnicas de compressão que também tiram partido deste tipo de imagens (imagens com grandes extensões no mesmo tom). Através de qualquer uma destas técnicas é possível reduzir bastante o tamanho das imagens que

representam *slides* de texto. Consequentemente, o tempo de transição entre este tipo de *slides* é também inferior.

A aplicação desenvolvida apresenta um grande fluxo de informação durante o início da sessão, como foi observado no segundo teste. Este fluxo corresponde à transferência do ficheiro. A maior parte do tráfego gerado encontra-se concentrado nesta fase pois as mensagens seguintes são curtas mensagens de sinalização, na sua maioria inferiores a 100 *bytes*.

O tráfego gerado pelo MSN Messenger possui características diferentes. Para além das suas mensagens de sinalização indicando que a sessão ainda se encontra activa, existem rajadas de informação quando se verifica a mudança de *slide*, acabando por se ocupar maior largura de banda ao longo de toda a sessão.

Embora o MSN Messenger envie cada *slide* à medida que este vai sendo visualizado o tráfego total gerado é superior ao da aplicação desenvolvida. Teoricamente, cada *slide* enviado pelo MSN Messenger teria um tamanho inferior ao do seu equivalente no ficheiro de *PowerPoint* pois a imagem enviada pela rede é codificada, ou seja, é enviada menos informação. Assim seria de esperar que o tráfego gerado fosse inferior. No entanto, o acto de recuar rapidamente até ao meio da apresentação no final do teste provoca um reenvio de grande parte, senão mesmo toda, a informação relativa aos *slides* que são visíveis apenas por breves instantes, enquanto que na aplicação desenvolvida apenas são enviadas pequenas mensagens de sinalização. De realçar que a codificação

utilizada pelo MSN Messenger deteriora significativamente a qualidade da imagem, sendo o resultado final pouco atractivo para o utilizador.

As sessões do MSN Messenger foram de duração significativamente mais curta do que as da aplicação. Esta discrepância deve-se a dois factores: o MSN Messenger não faz a transferência do ficheiro antes da sessão ser iniciada e não foi contabilizada a criação da sessão para esta aplicação. Optou-se por medir o tráfego apenas durante a fase da apresentação propriamente dita pois o estabelecimento da sessão por vezes pode ser bastante demorado e o tráfego gerado não iria influenciar grandemente os resultados finais.

De um modo geral, pode-se afirmar que a aplicação desenvolvida apresenta uma melhor performance dentro do contexto das reuniões de trabalho com suporte de apresentações de *slides*. A duração das transições de slide e o tráfego gerado pela aplicação são menores que os do MSN Messenger. A desvantagem que esta aplicação apresenta é que, caso seja necessário fazer a distribuição do ficheiro pelos utilizadores, existe um tempo morto no início de cada apresentação enquanto que com o MSN Messenger se pode começar a sessão de imediato.

Como já foi referido, o MSN Messenger apenas permite a partilha de aplicações entre dois utilizadores. Por esta razão não foram efectuadas medições de tráfego com mais de dois intervenientes por não haver possibilidade de comparar os resultados. No entanto, dado que a aplicação desenvolvida se baseia em IP Multicast, é seguro afirmar que o tráfego gerado não seria muito superior ao existente, pois ao utilizar este protocolo

a informação é enviada uma única vez, atingindo todos os destinatários. Tendo em conta que no MSN Messenger as ligações são feitas ponto-a-ponto, se este suportasse vários utilizadores, o tráfego gerado aumentaria proporcionalmente com o seu número, pois a mesma informação seria enviada distintamente para cada ouvinte.

6 – Conclusões

6.1 – Considerações Finais

Neste trabalho foram cumpridos todos os objectivos propostos. Foi desenvolvida uma aplicação que cumpriu os requisitos impostos de providenciar a realização de sessões de apresentação de uma forma distribuída, de modo a obter uma melhor performance em relação às soluções previamente existentes.

O desenvolvimento desta aplicação foi uma experiência enriquecedora pois permitiu-me aprofundar os conhecimentos de redes e de sistemas distribuídos, nomeadamente de IP Multicast. Entrei também em contacto com protocolos que visam implementar uma transferência de ficheiros fiável através de IP Multicast. Tive ainda a oportunidade de aprender uma nova linguagem de programação e de consolidar os conhecimentos de engenharia de *software* e programação orientada a objectos.

6.2 – Trabalho Futuro

Embora a aplicação desenvolvida cumpra os requisitos especificados existe ainda espaço para algumas melhorias.

Era possível melhorar a aplicação a nível gráfico, permitindo a visualização da apresentação em ecrã completo (*fullscreen*), ou pelo menos, possibilitando a maximização da janela do diálogo principal. As dimensões desta janela podiam ainda ser definidas dinamicamente, de acordo com a resolução do monitor do utilizador.

Ao nível de usabilidade e interactividade poder-se-ia incluir o suporte para *slides* com animações e transições. Seria também interessante fazer a exportação do ponteiro do rato sobre a área de visualização das apresentações. Através desta funcionalidade poder-se-ia apontar para zonas do *slide* de modo a realçar algum aspecto importante, à semelhança do que se faz actualmente nas projecções ordinárias.

Neste momento a aplicação encontra-se limitada a ficheiros de *PowerPoint* e ao sistema operativo Microsoft Windows. Poder-se-ia também proceder à adaptação a outras plataformas e outros tipos de ficheiros.

Referências e Bibliografia

1. SDP Internet Draft, http://www.ietf.org/internet-drafts/draft-ietf-mmusic-sdp-new-26.txt, Janeiro de 2006
2. MSDN Library, http://msdn.microsoft.com/library/, 2006
3. The Code Project, http://www.codeproject.com, 2006
4. CodeGuru, http://www.codeguru.com, 2006
5. How to use MFC to create and show a *PowerPoint* presentation, http://support.microsoft.com/kb/q180616/, 21 de Março de 2005
6. Chapman, Davis; *"Sams Teach Yourself Visual C++ in 21 Days"*, Macmillan Computer Publishing, 1998
7. SinisterSDP, http://sourceforge.net/projects/sinistersdp/
8. XMLite: simple XML Parser, http://www.codeproject.com/cpp/xmlite.asp, 2004
9. VNC – Virtual Network Computing form AT&T Laboratories Cambridge, http://www.csd.uwo.ca/staff/magi/doc/vnc/index.html, 19 de Março de 2001
10. VC++ Tutorial, http://www.softlookup.com/tutorial/vc++/vcu39fi.asp#I17
11. Saba, http://www.saba.com/centra-saba/

Anexo A: Lista Extensiva das Mensagens de Rede

Neste anexo são apresentados exemplos das mensagens de rede com formato XML usadas na aplicação.

Alive:

```
<DstShowCmd sessionid = "111111" type = "alive">
      <Username> Nome </Username>
      <Status> Conn </Status>
</DstShowCmd>
```

Flow:

```
<DstShowCmd sessionid = "111111" type = "flow">
      <Direction> Forward </Direction>
      <Slide Number> 2 </Slide Number>
      <AnimationNumber> %d </AnimationNumber>
</DstShowCmd>
```

Kick:

```
<DstShowCmd sessionid = "111111" type = "kick">
      <Username> Nome </Username>
</DstShowCmd>
```

SpkrChange:

```
<DstShowCmd sessionid = "111111" type = "SpkrChange">
      <Username> Nome </Username>
</DstShowCmd>
```

SpkrOK:

```
<DstShowCmd sessionid = "111111" type = "SpkrOK">
      <Username> Nome </Username>
</DstShowCmd>
```

Repeated:

```
<DstShowCmd sessionid = "111111" type = "repeated">
      <Username> nome </Username>
      <ip> 127.0.0.1 </ip>
      <port> 1000 </port>
</DstShowCmd>
```

Transfer:

```
<DstShowCmd sessionid = "111111" type = "transfer">
      <File> Nome do ficheiro<File>
</DstShowCmd>
```

Anexo B: Folhas de cálculo com os resultados do teste de usabilidade e performance

Tabela 1 – Resultados do utilizador A para B com a aplicação desenvolvida

A->B	t1	t2	t3	t4	Atraso	Duração da mudança	Transição
1	3	4	4	5	1	1	im - im
2	3	4	4	5	1	1	im - tx
3	4	4	4	5	1	1	tx - tx
4	3	4	4	4	0	0	tx - im
5	4	5	4	6	1	2	im - im
6	3	4	4	4	0	0	im - tx
7	3	4	4	5	1	1	tx - tx
8	4	4	4	5	1	1	tx - im
9	4	5	5	6	1	1	im - im
10	3	4	4	5	1	1	im - tx
11	4	4	4	5	1	1	tx - tx
12	3	4	4	5	1	1	tx - im
13	4	5	5	5	0	0	im - im
fps	3,461538	4,230769	4,153846	5	0,769231	0,846153846	
12,5	0,276923	0,338462	0,332308	0,4	0,061538	0,067692308	

Tabela 2 - Resultados do utilizador B para A com a aplicação desenvolvida

B->A	t1	t2	t3	t4	Atraso	Duração da mudança	Transição
1	1	2	2	3	1	1	im - im
2	1	2	1	2	0	1	im - tx
3	1	2	2	2	0	0	tx - tx
4	2	2	2	3	1	1	tx - im
5	1	2	2	3	1	1	im - im
6	1	2	2	2	0	0	im - tx
7	1	2	1	2	0	1	tx - tx
8	1	3	2	3	0	1	tx - im
9	1	1	1	1	0	0	im - im
10	1	2	1	2	0	1	im - tx
11	1	2	1	2	0	1	tx - tx
12	1	2	2	3	1	1	tx - im
13	1	2	2	2	0	0	im - im
fps	1,076923	2	1,615385	2,307692	0,307692	0,692307692	
12,5	0,086154	0,16	0,129231	0,184615	0,024615	0,055384615	

Tabela 3 - Resultados do utilizador A para B com o MSN Messenger

A->B	t1	t2	t3	t4	Atraso	Duração da mudança	Transição
1	1	2	3	8	6	5	im - im
2	1	2	3	9	7	6	im - tx
3	1	1	3	4	3	1	tx - tx
4	1	2	3	4	2	1	tx - im
5	1	3	6	8	5	2	im - im
6	1	1	4	8	7	4	im - tx
7	1	1	5	6	5	1	tx - tx
8	1	2	3	7	5	4	tx - im
9	1	3	2	9	6	7	im - im
10	1	2	3	4	2	1	im - tx
11	1	2	4	5	3	1	tx - tx
	1	3	4	9	6	5	tx - im
12	1	3	3	8	5	5	im - im
fps	1	2,076923	3,538462	6,846154	4,769231	3,307692308	
12,5	0,08	0,166154	0,283077	0,547692	0,381538	0,264615385	

Tabela 4 - Resultados do utilizador B para A com o MSN Messenger

B->A	t1	t2	t3	t4	Atraso	Duração da mudança	Transição
1	1	1	7	18	17	11	im - im
2	1	1	9	13	12	4	im - tx
3	1	2	3	17	15	14	tx - tx
4	1	2	9	13	11	4	tx - im
5	1	2	4	22	20	18	im - im
6	1	1	10	14	13	4	im - tx
7	1	1	2	19	18	17	tx - tx
8	1	2	2	24	22	22	tx - im
9	1	2	4	18	16	14	im - im
10	1	2	2	16	14	14	im - tx
11	1	1	13	15	14	2	tx - tx
12	1	2	4	19	17	15	tx - im
13	1	2	8	29	27	21	im - im
fps	1	1,615385	5,923077	18,23077	16,61538	12,30769231	
12,5	0,08	0,129231	0,473846	1,458462	1,329231	0,984615385	