Videoconferência Multiponto

José Manuel Costa, Pedro Miguel Teixeira, Joaquim Sousa Pinto, Joaquim Arnaldo Martins

Resumo - **O trabalho descrito foi realizado no âmbito do projecto de fim de curso na licenciatura em Engenharia electrónica e telecomunicações da Universidade de Aveiro. É aqui descrito o projecto e a implementação de um sistema de videoconferência em broadcasting para o sistema operativo Windows95 usando para o efeito um pacote de desenvolvimento de software da VCON, o ARMADA CRUISER 100. Para permitir a gestão da videoconferência, foi desenvolvido um protocolo de comunicações sobre TCP/IP, usando ligações por datagrama (UDP), o que permite criar um ambiente distribuído, em que não existem ligações dedicadas entre as várias máquinas intervenientes, nem um servidor central. O utilitário desenvolvido exige a presença de um utilizador com o estatuto de supervisor que possui privilégios especiais que lhe permitem a gestão da sessão. Em cada momento existe apenas um orador que monopoliza a emissão de imagem e som, funcionando o sistema como uma conferência virtual com orador e audiência. O processo de selecção deste pode ser automático ou depender da intervenção do supervisor.**

Abstract – **Here is described the project and the implementation of a broadcast videoconference system for Windows95 using the VCON software development kit ARMADA CRUSER 100. A communications protocol was developed to allow an easier control of the videoconference. This protocol was implemented over the TCP/IP protocol suite using datagram connections allowing the creation of a distributed environment where neither are dedicated links between machines nor a central server. The application requires the presence of a supervisor user, which has some special privileges, such as the session control. There is only one speaker at a time that broadcasts to all the other machines. The system works like a virtual conference with one speaker and the audience. The process of selecting a new speaker can be done automatically or trough the supervisor.**

I. INTRODUÇÃO

Pretendeu-se desenvolver uma aplicação Win32 para o sistema operativo Windows95, que implementasse um sistema de videoconferência em multicast utilizando um protocolo de comunicações baseado nos protocolos da família TCP/IP.

O sistema deveria ser descentralizado dispensando assim o uso de um servidor central para gestão das comunicações entre as máquinas. Isto permitiria que o protocolo se tornasse menos pesado ao nível do número de ligações necessárias. Foram por isso usadas comunicações por datagrama (UDP).

A aplicação desenvolvida permite a vários utilizadores participar de uma conferência virtual onde existe um orador que "fala" para todos os outros, tornando-os em audiência. Os papeis não são estáticos, pois a qualquer momento poder-se-á dar uma mudança de papéis, possibilitando deste modo tornar a sessão interactiva.

Para controlar a videoconferência há um supervisor que possui privilégios superiores aos dos demais utilizadores, como por exemplo decidir quem poderá intervir em cada momento. Este terá também como função gerir e garantir o bom funcionamento da sessão podendo para tal excluir utilizadores da mesma.

Devido a limitações nos meios disponíveis para o desenvolvimento do sistema, e que mais tarde serão discutidas, poderá apenas ser utilizado em redes locais e é controlado por duas aplicações, uma de 16 bits e outra de 32 bits. Estas aplicações comunicam entre si através do sistema de mensagens do Windows.

II. LIMITAÇÕES

As ferramentas de desenvolvimento utilizadas impuseram duas limitações bastante importantes aos objectivos iniciais. O pacote de desenvolvimento utilizado foi projectado para aplicações a 16 bits e além disso apenas permitia ligações entre duas máquinas. Foi pois necessário encontrar soluções que contornassem estes problemas.

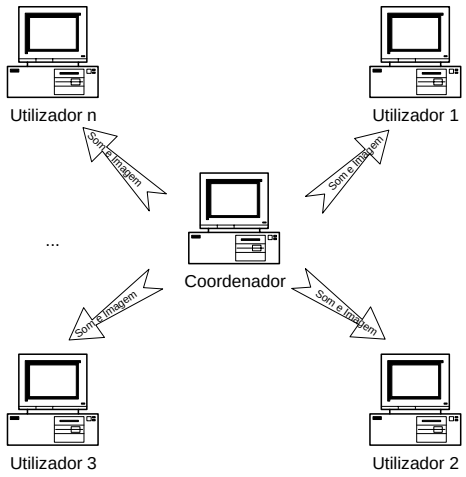

Fig. 1 - Modelo de videoconferência em *broadcast*

A. Broadcasting com a VCON

O hardware e software disponíveis para este projecto não permitem uma implementação directa de um sistema em *broadcasting*. O SDK (Software Development Kit) é bastante limitado nas facilidades proporcionadas ao programador no que diz respeito ao estabelecimento de ligações entre máquinas. A única função disponível no SDK para este efeito, *RequestInitConversation()*, tem como parâmetro de entrada o endereço IP da máquina com a qual se pretende iniciar uma conversação. Esta função não tem qualquer outro parâmetro não havendo assim nenhuma opção para ligações com várias máquinas em simultâneo.

Para atingir os objectivos pretendidos era indispensável colocar uma máquina a emitir para as restantes. Como o sistema operativo Windows95 nem o SDK do placa utilizada não permitiam a utilização de protocolos *multicast*, foi pois necessário recorrer a um processo alternativo representado na Fig. 2

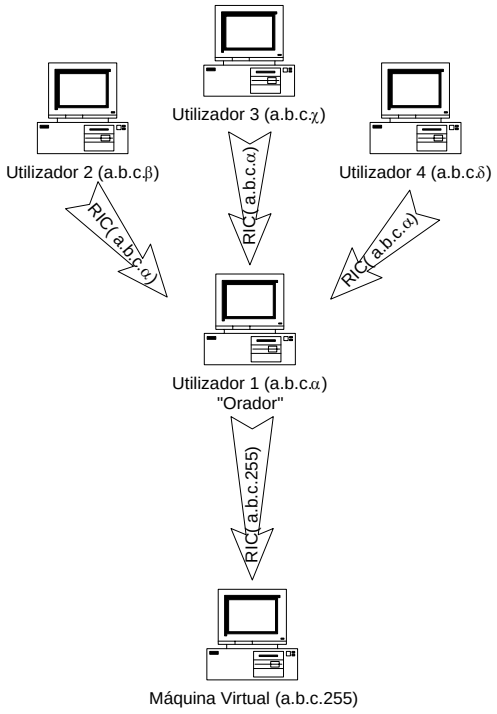

Fig. 2 – *Broadcasting* com a VCON

A máquina do "orador" começa por fazer um RIC (RequestInitConversation) para um endereço, que é igual ao seu até ao nível da rede, mas que tem o campo identificador da máquina igual a 255, endereço *NetBroadcast*. Verificou-se que nesta situação a função RIC não retornava nenhum erro, comportando-se todo o sistema como se de facto tivesse sido estabelecida uma verdadeira ligação. Na realidade a máquina do orador encontra-se a emitir em *broadcast*.

Em seguida, todas as restantes máquinas da conferência fazem um RIC para o endereço do orador. Esta operação não produz qualquer alteração no estado da máquina emissora, contudo, as restantes máquinas têm agora acesso ao som e à imagem que está ser emitida pelo orador. Esta solução limita contudo o uso do sistema a redes locais.

B. Compilador thunk

O Windows95 não permite a mistura de componentes de 16 bits com componentes de 32 bits dentro de uma mesma aplicação. Este facto impedia o uso do Microsoft Visual C++ 4.2, que gera somente aplicações de 32 bits, pois as funções fornecidas com o SDK encontram-se compiladas em dlls de 16 bits. Para superar esta limitação foi necessário usar uma ferramenta, fornecida pela Microsoft, o compilador Thunk.

O compilador ***thunk*** permite, em run-time, realizar o interface entre componentes de 16 e 32 bits. Este tem por função realizar uma conversão entre endereços que num ambiente Windows16 é do tipo 16:16, com um segmento e um *offset*, e num ambiente Windows32 são essencialmente constituídos por um *offset* 0:32. Para tornar este interface portável entre plataformas o código de thunking deverá ser encapsulado em dlls. A implementação do processo de thunking implica pois a criação de duas dlls, uma do tipo Win32 e outra Win16. Estas poderão ser substituídas por outras no caso de mudança de plataforma. O processo encontra-se representado na Fig. 3.

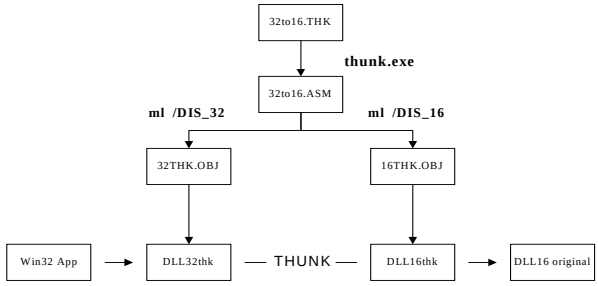

Fig. 3 - Mecanismo de *thunking*

O primeiro passo é a criação de um ficheiro script onde são declaradas todas as funções que se pretendem utilizar. Sendo o compilador thunk bastante limitado é necessário também declarar certos tipos básicos como por exemplo BOOL. É também necessário especificar para os argumentos do tipo ponteiro se estes serão usados como argumentos de entrada, de saídas ou ambos.

Na primeira linha deste ficheiro encontra-se uma declaração que indica ao compilador que o mecanismo a implementar é de 32 para 16 pois o contrário é também possível.

A declaração

```
enablemapdirect3216 = true;
```

indica que será código de 16 bits a ser utilizado por código de 32 em ambiente Windows32.

O exemplo seguinte mostra a declaração de uma função em que um dos argumentos é um ponteiro.

Mensagem	Função
RequestConnection()	Mensagem enviada para a rede com o pedido de ligação a uma sessão de videoconferência ou criação de uma como supervisor.
AcceptUser()	Informar um utilizador em espera de que a sua entrada foi aceite.
RefuseNewUser()	Informar um utilizador de que foi negado o pedido de adesão à sessão.
UserIn()	Informar os intervenientes da videoconferência da entrada de um novo utilizador.
UserOut()	Informar os intervenientes da videoconferência da saída de um utilizador.
EndVConf()	Informar os intervenientes da videoconferência que esta terminou.
FatalError()	Informar os intervenientes da videoconferência da ocorrência de um erro que implica o fim da sessão.
UserDisconnect()	Mensagem usada por um utilizador para sinalizar a sua saída.
KickUser()	Expulsar um utilizador.
CheckConnection()	Verificar o estado das ligações entre o supervisor e os utilizadores.
AckConnection()	Usada por um utilizador para informar o supervisor do estado da ligação.
Check2Master()	Permite ao supervisor verificar se é único.
RequestPermission2Speak()	Usada por um utilizador que deseje tornar-se orador.
AckPermission2Speak()	Mensagem usada pelo coordenador para sinalizar a um utilizador a recepção do seu pedido para se tornar orador.
ChangeWaitOrder()	Mensagem usada pelo supervisor para sinalizar mudança de posição de um utilizador na lista de espera.
RIC()	Mensagem usada pelo coordenador para indicar a uma máquina, que deve estabelecer uma ligação com outra, utilizando a VCON.
Ack_RIC()	Mensagem usada pelo orador para confirmar ao supervisor, que se encontra a emitir.
RTC()	Mensagem usada pelo supervisor da videoconferência, para instruir um utilizador a terminar qualquer ligação em curso, através da VCON.

Tabela 1 - Funções que implementam o protocolo de comunicações entre máquinas.

```
WORD vcll_RIC(CONVTYPE ConversationType,
            LPCONVPARAM_AVD Conv_avd)
{
    Conv_avd = input;
}
```

Concluído o ficheiro script é agora necessário utilizar o aplicativo thunk.exe que o compilará num ficheiro .ASM.

```
thunk.exe 32to16.thk -o 32to16.asm
```

O ficheiro resultante deste processo deverá ser assemblado duas vezes, uma com a opção –DIS_32 gerando assim código de 32 bits e outra com a opção –DIS_16 gerando código de 16 bits. O aplicativo usado para este efeito é o Microsoft MASM.

```
ml  /DIS_32 thk32.obj 32to16.asm
ml  /DIS_16 thk16.obj 32to16.asm
```

Para finalizar, falta apenas criar as duas dl's: uma dll win32 onde será integrado o ficheiro thk32.obj e outra win16 onde será integrado o ficheiro thk16.obj.

Na documentação fornecida pela Microsoft, para uso do compilador thunk, são feitas diversas referências a problemas que podem advir do recurso a este processo. Apesar de terem sido evitadas todas as situações problemáticas que poderão causar erros no uso do compilador thunk, não foi possível conseguir um comportamento aceitável da aplicação.

Todas as funções incluídas neste mecanismo funcionam, no entanto e por razões que não nos foi até agora possível determinar e também porque ocorrem de uma maneira perfeitamente aleatória, a aplicação provoca com demasiada frequência um erro de sistema do tipo "*Page Fault*". Verificou-se no entanto que os erros apenas acontecem quando é invocada uma função que comunica com o hardware.

O processo utilizado para contornar o problema com o SDK a 16 bits consistiu na implementação de duas aplicações distintas, uma a 32 bits e outra a 16 bits. A aplicação Win32 controla o protocolo de comunicações de alto nível entre máquinas e a outra controla todos os processos que envolvam a placa de videoconferência, tais como o estabelecimento de ligações ou a abertura de janelas de vídeo. As duas aplicações comunicam também entre si através de um protocolo implementado sobre o sistemas de mensagens do Windows.

III. PROTOCOLO DE COMUNICAÇÕES ENTRE MÁQUINAS

Visando facilitar o desenvolvimento da aplicação de controlo (Win32), foi implementado um protocolo, que providencia facilidades de comunicação entre as diversas máquinas da sessão. O protocolo tem por base o TCP/IP sendo assim independente do suporte físico utilizado na rede.

O protocolo de rede TCP/IP permite dois tipos de ligações: circuito virtual ou datagrama. O primeiro usando TCP implica o estabelecimento de ligações entre as diversas máquinas, sendo mais indicado para casos em que exista um servidor central. Este permite no entanto

Mensagem	Função
ErrorOnInitialisation()	Notificar a aplicação de controle da videoconferência que ocorreram erros na inicialização do hardware.
InitialisationOk()	Notificar a aplicação de controle da videoconferência que a inicialização do hardware se fez correctamente.
StartConversation()	Indicar à aplicação de 16 bits que deve iniciar uma conversação para um determinado endereço IP.
EndConversation()	Indicar à aplicação de 16 bits que deve terminar qualquer ligação em curso.
ConversationOk()	Indicar à aplicação de 32 bits que a conversação de iniciou sem erros.
ConversationNotOk()	Indicar à aplicação de 32 bits que aconteceram erros no estabelecimento da ligação.
CloseApp()	Indicar à outra aplicação que esta deve terminar.

Tabela 2 – Funções que implementam o protocolo de comunicação entre aplicações.

um aligeirar da complexidade do protocolo pois possui mecanismos para recuperação de erros nas transmissões. Nas ligações tipo datagrama, usando UDP, não existe correcção de erros e por isso o protocolo a implementar deverá ser mais complexo pois a detecção de erros será feita por este. No entanto o uso de UDP aumenta significativamente a versatilidade do protocolo pois evita a existência de um servidor centralizado para a distribuição de mensagens entre as máquinas. Com a opção escolhida (UDP), o sistema de videoconferência evitará assim a existência de uma máquina central permitindo se necessário a comunicação directa entre as máquinas intervenientes sem um sobrecarregar da rede.

A Tabela 1 sumaria as mensagens que deste protocolo. Algumas mensagens contêm parâmetros que fornecem informação adicional. A mensagem *RequestConnection()* é disso um exemplo, pois além de pedir por uma ligação contém ainda parâmetros que permitem a um potencial utilizador especificar opções de ligação.

IV. PROTOCOLO DE COMUNICAÇÕES ENTRE AS APLICAÇÕES

Devido à solução utilizada em que se optou por duas aplicações distintas, foi necessário implementar outro protocolo que possibilitasse a comunicação entre estas.

Esse protocolo foi desenvolvido tendo por base o sistema de mensagens do Windows. O protocolo é extremamente simples pois as aplicações são bastante autónomas e não trocam muita informação entre si. Na tabela 2 encontram-se sumariadas as diversas funções implementadas.

V. ALGORITMOS

Para possibilitar uma melhor compreensão da função de cada mensagem bem como dos processos mais importantes na gestão do sistema implementado serão em seguida descritos alguns dos algoritmos usados.

A. Criação de uma sessão de videoconferência.

O processo de criação de uma sessão de videoconferência ou de adesão de um novo utilizador são de extrema importância pois, a ocorrência de erros, podem conduzir a situações em que existem, por exemplo, dois supervisores para uma só sessão, sendo então necessário terminar a sessão. Este mecanismo de controlo está representado na Figura 4.

A adesão é conseguida enviando para a rede a mensagem *RequestConnection()* com as respectivas opções de ligação. Se a opção for apenas a de criação de uma sessão, como supervisor, e esta já existir, o actual supervisor deverá enviar em resposta a mensagem *RefuseUser()*. Caso a sessão não exista, o potencial supervisor não receberá qualquer resposta e poderá então criar a sua própria videoconferência.

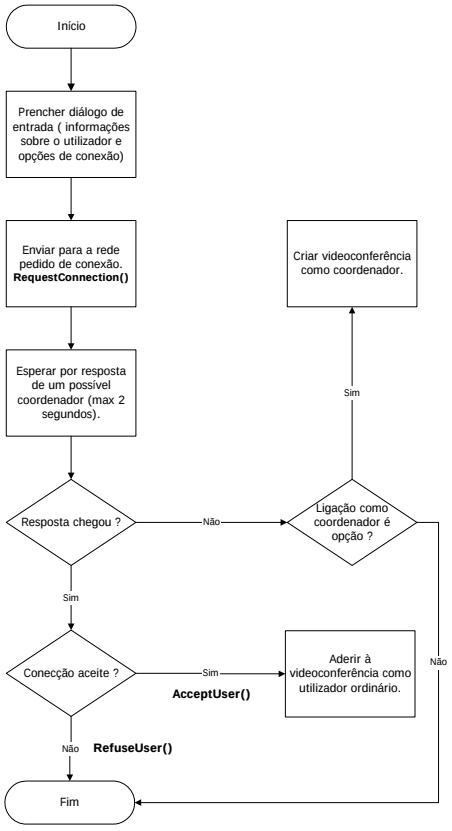

Fig. 4 - Criação de uma sessão de videoconferência

B. Lançamento da aplicação Win16.

Quando é criada uma nova sessão de videoconferência por parte do supervisor, a aplicação Win32 deve lançar a aplicação Win16 que permite controlar o hardware da

placa VCON. Qualquer falha no lançamento da aplicação ou na inicialização do hardware é considerado como um erro fatal o que implica o fim da sessão.

O processo de lançamento da aplicação de controlo da VCON por parte de um utilizador vulgar é em tudo idêntico, excepto no facto de que em caso de erro a única consequência é a sua exclusão da videoconferência. Este mecanismo está representado na Figura 5.

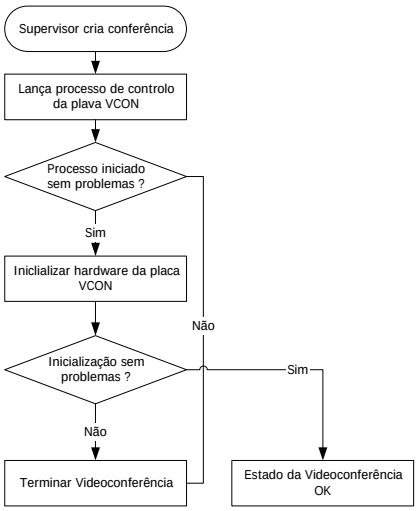

Fig. 5 – Lançamento da aplicação Win16

C. Adesão de novos utilizadores.

A entrada de novos utilizadores numa sessão de videoconferência é independente da vontade do supervisor, não tendo este o poder de negar acesso à sessão. Um utilizador apenas será impedido de conseguir ligação se o endereço IP da sua máquina já pertencer a outro utilizador, o que poderá acontecer no caso de uma saída anormal seguida de uma tentativa de ligação imediata. Em qualquer outra situação o novo utilizador será sempre autorizado a aderir.

Quando o supervisor recebe um pedido de adesão à sessão verifica imediatamente o endereço IP da máquina que o enviou comparado-o depois com os endereços de todos os utilizadores presentes. Caso o resultado da comparação seja negativo, será então feita a verificação das opções de conexão do utilizador, se a ligação como utilizador vulgar não tiver sido escolhida será enviada um recusa de conexão, pois as intenções são apenas de criação de uma nova sessão. No caso desta opção existir será de imediato atribuído um numero de ordem que será enviado juntamente com o perfil do supervisor numa mensagem do tipo *AcceptUser*().

O perfil do novo utilizador contém um campo com o seu acrónimo. Para evitar que dois ou mais utilizadores tenham o mesmo acrónimo, caso já exista um acrónimo igual na sessão é-lhe acrescentado um "a" (josé, joséa, joséaa, …).

O perfil do novo utilizador depois de ser acrescentado à lista de utilizadores é enviado juntamente com a mensagem *UserIn*() a toda a conferência.

Todo este mecanismo está representado na Figura 6.

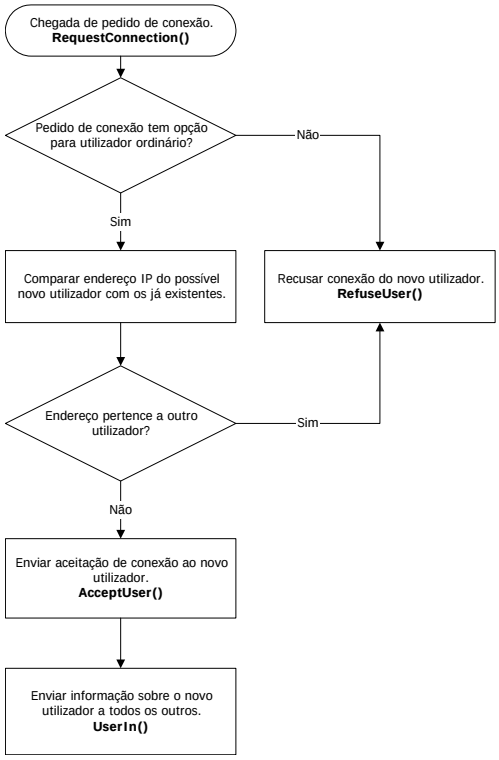

Fig. 6 – Adesão de novos utilizadores.

E. Controlo de ligações.

O processo de controlo das ligações entre as diversas máquinas é de vital importância para a manutenção da integridade da videoconferência. Este controlo deve ser efectuado o mais periodicamente possível não devendo no entanto sobrecarregar em demasia quer a máquina quer a rede. Tanto o supervisor como os utilizadores possuem mecanismos que permitem detectar abandonos não comunicados. Estes abandonos podem ocorrer devido a falhas de energia, falhas na rede, erros no Windows, etc..

O supervisor da conferência tem necessidade de ter presente uma imagem actualizada dos utilizadores na sessão, por isso deve possuir mecanismos que verifiquem periodicamente se realmente eles lá estão.

Para realizar este controlo, o supervisor envia de 15 em 15 segundos a uma mensagem *CheckConnection*() para todos os utilizadores que se encontram na sua lista. Estes têm dois segundos para responderem com uma mensagem *AckCheckConnection()*. Findo este tempo o supervisor irá verificar se algum utilizador falhou o controlo por duas vezes consecutivas. Caso exista algum utilizador nessa situação, o supervisor assume que o utilizador já não se encontra presente e remove-o da conferência, notificando de seguida os demais.

Tal como o supervisor deve garantir a presença dos utilizadores na sessão, também estes devem verificar periodicamente se o supervisor ainda se encontra presente. Esta situação é crítica pois a falta do supervisor significa uma videoconferência sem controlo. Os utilizadores usam

a mensagem *CheckConnection()* para confirmarem a presença do supervisor se num período de tempo de 35 segundos não receberem qualquer contacto por parte deste. Nessa altura o utilizador deve presumir que a videoconferência terminou.

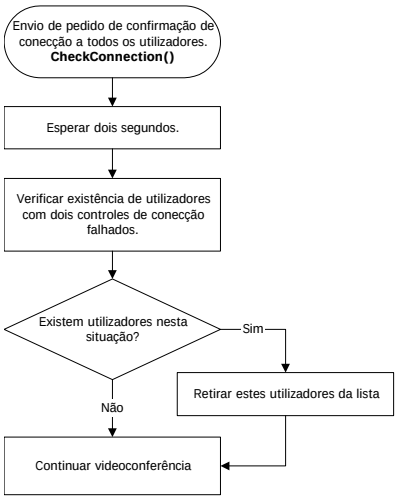

Fig. 7 – Controlo de ligações.

Note-se que o período de 35 segundos equivale a um pouco mais de duas vezes a periodicidade com que o supervisor realiza o controlo, o que significa que é permitida a perda de um *CheckConnection()*, quer por parte do supervisor quer por parte dos utilizadores normais. O mecanismo está representado na Figura 7.

F. Escolha e mudança de orador.

O processo de escolha de orador pode ser iniciado por atribuição do supervisor ou desencadeado de forma automática. O supervisor pode a qualquer momento tornar-se "orador", bastando para isso carregar no botão respectivo. É também prerrogativa do supervisor escolher qual o próximo "orador" de entre os pedidos para intervenção que possui numa fila de espera. Existem no entanto situações em que todo o processo se desenrola automaticamente como, por exemplo, quando o "orador" cede a palavra ou quando o orador previamente escolhido não consegue por qualquer razão iniciar a emissão de som e imagem. Quando se dá uma mudança de "orador" o supervisor envia para todas as máquinas uma mensagem *RTC()*, terminando com possíveis ligações efectuadas pela VCON. De seguida o supervisor envia um *RIC()* ao "orador" instruindo-o para iniciar a sua emissão de som e imagem em *NetBroadcast*. Se o novo orador responder afirmativamente (*Ack_RIC()*), ou seja, se conseguiu iniciar a emissão sem erros, todos os outros utilizadores serão instruídos para também efectuarem uma ligação à máquina do "orador". Caso o novo "orador" não envie confirmação, porque por qualquer razão não lhe foi possível, será automaticamente escolhido um outro e o processo anterior reiniciar-se-á. Este complexo processo está representado na Figura 8.

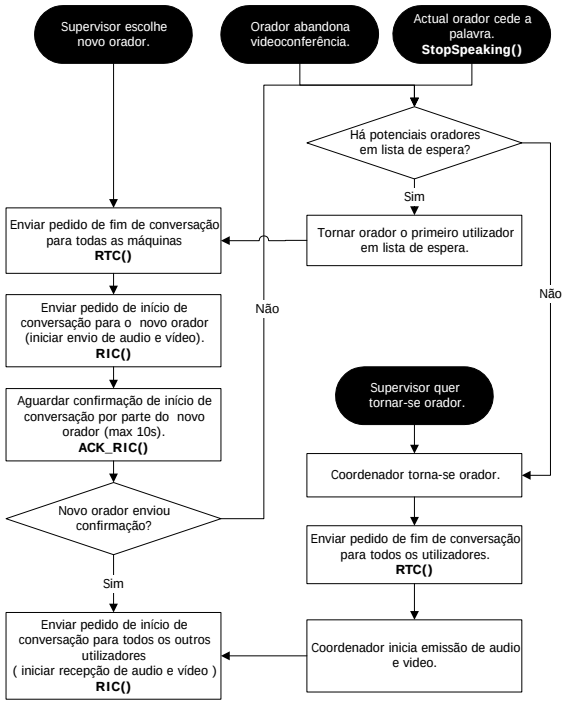

Fig. 8 – Escolha e mudança de orador

VI. SITUAÇÕES DE ERRO.

No decorrer de uma sessão de videoconferência são várias as situações em que o seu estado pode ser comprometido. Torna-se pois necessário implementar mecanismos de detecção e correcção ou superação de erros. O uso de ligações por datagrama é um factor gerador de erros de comunicação. Estes não são detectados ou corrigidos pelo protocolo TCP/IP. A ocorrência de situações estranhas como, por exemplo, falhas de energia, falhas na rede, ou o fecho da aplicação de maneira anormal devem também ser tidas em conta.

Algumas das situações de erro mais comuns são:

- Quebra da ligação entre o supervisor e o utilizador.
  Se por alguma razão (falha da rede, por exemplo) a ligação entre o supervisor e o utilizador for quebrada, o mecanismo de verificação de ligação detectará esta anomalia e retirará o utilizador da sessão.
- Saída anormal do supervisor.
  Se por qualquer razão a aplicação do supervisor terminar sem avisar os demais utilizadores (falha de energia, por exemplo) a sessão terminará. Os demais utilizadores detectarão a falta do supervisor e desligar-se-ão.
- Existência de duas sessões em simultâneo.
  Esta situação traduz-se pela existência de dois supervisores para uma mesma sessão (pode ocorrer quando é restabelecida a comunicação entre dois troços da rede após uma falha prolongada). Se um novo utilizador não detecta a existência de uma sessão e inicia outra (por impossibilidade de detectar a colisão), criará uma situação de conflito que ao ser detectada ocasionará o fecho de ambas as sessões.

O mecanismo de verificação de ligações implementado na aplicação do supervisor assim como na aplicação dos utilizadores é capaz de assegurar que em cada instante o estado da sessão é normal.

VII. DESCRIÇÃO DAS APLICAÇÕES

Como já foi referido anteriormente a implementação do sistema é conseguida através de duas aplicações distintas, a primeira, a 32 bits, que controla todo o protocolo de comunicações entre máquinas e permite a gestão da sessão de videoconferência, a segunda, a 16 bits, que controla todo o processo de gestão do hardware necessário ao estabelecimento de ligações, visualização de imagem, som, etc.

A. Aplicação de controlo da videoconferência

A aplicação de controlo da videoconferência ao ser executada apresenta ao utilizador uma janela de entrada que permite recolher todos os dados considerados necessários para a adesão a uma sessão, tal como o representado na Figura 9.

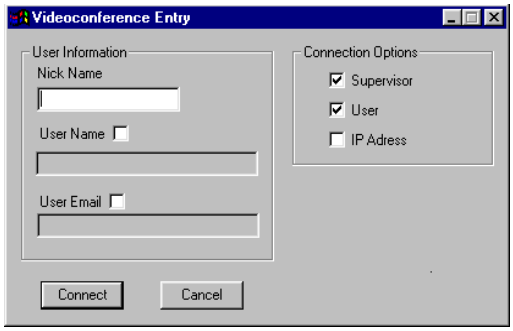

Fig. 9 – Diálogo de entrada na videoconferência

Este diálogo permite ao utilizador escolher o nível de confidencialidade pretendido, necessitando apenas de indicar um acrónimo pelo qual será identificado e as opções de ligação. O nome do utilizador, o seu correio electrónico e a divulgação do endereço da sua máquina são facultativos. Nas opções de ligação utilizador deverá ainda indicar qual o papel que deseja desempenhar (supervisor, utilizador normal ou ambos). No caso de o utilizador ter escolhido o papel de supervisor, a decisão basear-se-á na existência ou não de uma sessão já estabelecida.

A Figura 10 apresenta a janela do supervisor. Esta está dividida em três partes: visualização do estado da sessão, controlo da sessão e mensagens.

Na área de visualização do estado (Figura 11) são visíveis todos os intervenientes da sessão bem como informações a seu respeito. Diferentes ícones identificam o tipo de informação. A área de mensagens é usada para facultar ao supervisor informação sobre o que está a acontecer indicando nomeadamente a saída e entrada de utilizadores, pedidos de palavra, erros , etc.

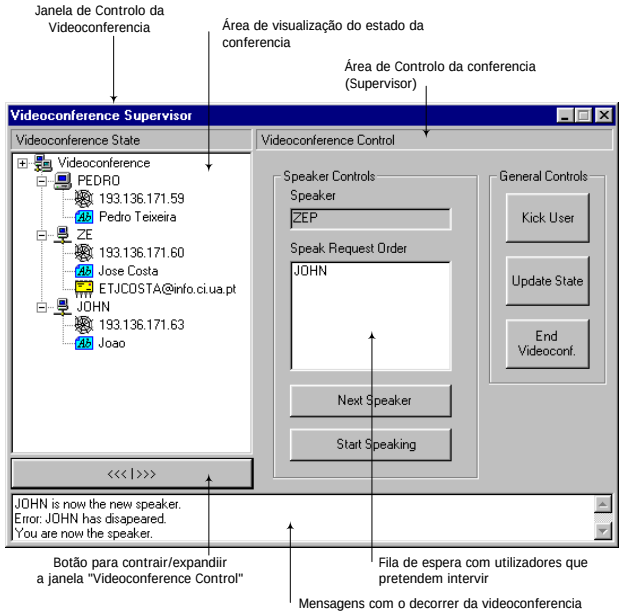

Fig. 10 – Aplicação de controlo da videoconferência do supervisor

Na área de controlo encontram-se todos os comandos que o supervisor necessita para controlar a sessão, bem como a lista dos pedidos para intervir.

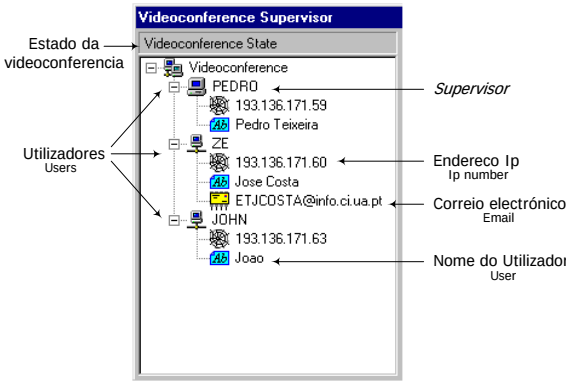

Fig. 11– Área de visualização do estado da sessão

No caso de ligação como utilizador normal, os comandos para controlo da sessão são substituídos pelos comandos relativos a um utilizador com este perfil. Esse painel está representado na Figura 12.

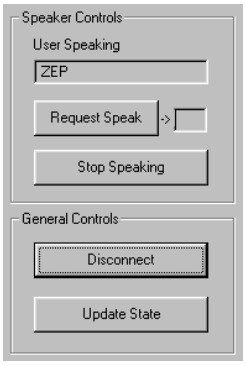

Fig. 12 – Área de controlo com comandos para um utilizador normal

B. Aplicação de controlo da VCON

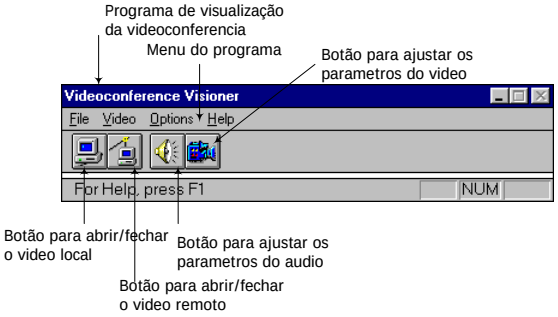

Fig. 13 – Programa de visualização

Para efectuar as ligações entre as máquinas, gerir a imagem e o som e ainda controlar o hardware específico para videoconferência, implementou-se uma aplicação a 16 bits, pelos motivos previamente descritos. Esta é constituída por uma pequena janela, tal como o representado na Figura 13, que nos permite abrir e fechar janelas de vídeo, controlar parâmetros da ligação e modificar características do som e da imagem. Esta aplicação é igual em todos os utilizadores e independente do perfil dos mesmos.

VIII. CONCLUSÕES

Este projecto apresenta, como resultados visíveis, um aplicativo para o sistema operativo Windows95 capaz de gerir uma sessão de videoconferência multiutilizador, para possibilitar a comunicação entre máquinas. Tal como o descrito foi desenvolvido um protocolo de comunicações baseado em TCP/IP capaz de controlar a sessão de trabalho e independente do tipo de hardware utilizado para videoconferência.

Apesar do inicialmente previsto, ou seja, a criação de uma aplicação de 32 bits, o sistema foi desenvolvido através de duas aplicações autónomas que comunicam entre si. Uma delas foi desenvolvida para sistemas Win32 e é responsável pelo controlo da sessão, gerindo a comunicação entre as máquinas e implementando o interface gráfico com o utilizador. Devido a problemas encontrados com o hardware disponível, chegou-se à conclusão que o pacote de desenvolvimento fornecido pela VCON apresentava diversas limitações, como seja o facto de todas as funções disponibilizadas se encontrarem compiladas para aplicações de 16 bits. Este factor tornava problemática a sua utilização em ambientes de programação Win32, pois estes não permitem a mistura de código de 16 bits com código de 32 bits.

Na tentativa resolver este problema recorreu-se a uma ferramenta, fornecida pela Microsoft, chamada compilador *thunk* que permite a criação de um interface entre os dois tipos de código, realizando de uma forma automática a conversão entre endereços de um espaço de endereçamento segmentado (Win16) para um não segmentado ou *flat* (Win32). Este método apresenta no entanto diversas limitações e pelos testes efectuados apenas funciona correctamente em situações muito específicas e que não envolvam controlo de hardware. Foi testada uma versão ainda limitada do aplicativo recorrendo a este processo, no entanto o seu comportamento não era satisfatório provocando erros no sistema operativo.

Não sendo possível tornar a aplicação Win32 fiável optou-se por criar um aplicativo de 16 bits que seria responsável por todos os processos que envolvessem interacção com o SDK. Estas duas aplicações comunicam entre si através de um protocolo implementado sobre o sistema de mensagens do Windows.

As ferramentas de desenvolvimento, fornecidas pela VCON, introduziram ainda uma outra limitação ao projecto, pois o SDK não continha qualquer função que permitisse o estabelecimento de múltiplas ligações (*multicast*), comprometendo assim o objectivo de conseguir implementar uma conferência. A solução encontrada permitiu simular emissão de som e imagem em *broadcast* limitando no entanto o uso do programa a redes locais.

Foi implementado um protocolo de comunicações baseado em TCP/IP usando ligações do tipo datagrama. Esta solução apesar de exigir uma maior complexidade do protocolo, dispensa o uso de um servidor central e é ainda menos exigente do ponto de vista das ligações entre estações de trabalho.

A aplicação final simula uma conferência onde existe um supervisor responsável pelo seu controlo e um “orador” que é escolhido de entre todos os intervenientes. A escolha do “orador” pode ser realizada de forma automática ou depender da vontade do supervisor.

A aplicação foi testada exaustivamente, no entanto e devido à limitação existente em termos de recursos, como sejam computadores e kits VCON não foi possível realizar todos os teste necessários, nomeadamente com um número de intervenientes elevado (superior a 4).

IX. BIBLIOGRAFIA

[1] Joaquim M. H. de Sousa Pinto, "Arquitectura para um sistema Colaborativo baseado em ferramentas hipermédia", Universidade de Aveiro, 1997

[2] Jeff Prosise , "Programming Windows 95 with MFC", Microsoft Press

[3] Steve Oualline, "Practical C++ Programming", O’Reilly & Associates, Inc.

[4] Bjarne Stroutrup, The C++ Programming Language, Addison Wesley

[5] Douglas E.Charles, "Internetworking with TCP/IP : principles, protocols and architecture", Prentice - Hall , 1988

[6] Floyd Wilder, A guide to the TCP/IP protocol suite, 1993