

Ferramenta de Vídeo em Tempo Real e Aplicações a Ensino a Distância

Daniel Sadoc Menasché¹, Edmundo de Souza e Silva¹, Rosa M. M. Leão¹

¹Universidade Federal do Rio de Janeiro - COPPE/Prog. de Engenharia de Sistemas e Computação
Caixa Postal 68511, CEP: 21941-972–Rio de Janeiro, RJ
{sadoc, edmundo, rosa}@land.ufrj.br
Área: Aplicações em Redes Avançadas, Ensino a Distância

Com a disseminação da Internet, o ensino a distância surge como uma aplicação com significativo potencial e impacto social. Para dar suporte ao ensino a distância, várias ferramentas podem ser utilizadas incluindo ferramentas de transmissão de vídeo em tempo real. A implementação desta aplicação traz uma série de desafios de projeto na área de redes de comunicação, uma vez que a Internet não fornece a qualidade de serviço que se faz necessária. O objetivo deste trabalho é apresentar os problemas inerentes à concepção e à implementação desta aplicação, e algumas das estratégias utilizadas no aplicativo que foi desenvolvido.

No Laboratório de Análise e Modelagem de Sistemas de Computação e Comunicação (LAND/COPPE/UFRJ, www.land.ufrj.br) vêm sendo desenvolvidas pesquisas que resultaram em ferramentas com aplicações tanto no ensino a distância quanto no trabalho cooperativo. Dentre estas aplicações, destacam-se um Servidor Multimídia [9] e uma ferramenta de transmissão de voz (Viva-Voz) que implementa um algoritmo eficiente de recuperação de pacotes [6]. Este trabalho, por sua vez, descreve uma ferramenta de transmissão de vídeo em tempo real para videoconferência.

Existem várias ferramentas de transmissão de vídeo hoje disponíveis. No entanto, a maioria destas não está em domínio público ou tem código-fonte fechado. Duas delas atendem os padrões H.323: o NetMeeting e o GnomeMeeting. Outro programa de videoconferência popular no ambiente Linux é o VIC [8]. O GnomeMeeting e a respectiva biblioteca H.323 têm o código aberto. Entretanto, a complexidade dos programas que fazem parte deste conjunto tornam a alteração do código para efeitos de implementação de novos algoritmos de QoS, uma tarefa bastante árdua. Além disso, pouca flexibilidade se tem quanto à coleta de estatísticas.

Como o nosso objetivo é o de realizar pesquisa em redes e por conseguinte experimentar diferentes soluções para aumentar a QoS fornecida ao usuário e ainda coletar estatísticas durante uma transmissão, optou-se por seguir uma abordagem mais simples: implementar um programa com um mínimo de requisitos e amigável no que diz respeito a modificações e ajustes no código-fonte. Desta forma, pode-se facilmente realizar experimentos que envolvam, por exemplo, coleta de estatísticas, diversas abordagens de controle de fluxo e diferentes algoritmos de redução de “jitter”.

São dois os principais problemas de uma transmissão de vídeo na Internet: requisitos restritos de tempo, e a perda de informação [5, 7] por descarte de pacotes. Aplicações de vídeo são muito **sensíveis a retardo**. Por exemplo, apenas pequenos retardos entre a captura dos dados ao vivo e sua exibição na tela dos clientes são tolerados. O atraso de um pacote por mais de 1 ou 2 segundos pode tornar inviável uma aplicação de teleconferência com alta interatividade. Por outro lado, estas aplicações são em geral **tolerantes a falhas**, isto é, a transmissão não é comprometida pela perda de alguns pacotes.

O modo de operação do sistema de transmissão de vídeo pode ser orientado ao servidor (“server driven”) ou orientado ao cliente (“client driven”). Num sistema orientado ao cliente, estes clientes solicitam cada bloco em separado ao servidor, que atende os pedidos na medida em que são recebidos. Já no sistema orientado ao servidor, este envia continuamente dados aos clientes, que não fazem solicitações individuais de blocos.

A principal motivação para se utilizar a política orientada ao cliente é o controle refinado de

fluxo que se pode estabelecer entre o servidor e o cliente. As taxas de transmissão do emissor e consumo de blocos no cliente não são exatamente sincronizadas, o que pode causar, por exemplo, um aumento pequeno mas gradual do consumo de “buffer” com conseqüente aumento do tempo entre a captura e visualização da imagem. Assim sendo, haverá significativa perda de qualidade em um ambiente interativo, além de perdas de informação pelo estouro (“overflow”) de “buffer”. No caso da política orientada ao cliente, o fluxo é equalizado pelos pedidos dos clientes a medida que os dados são consumidos. Esta política é adequada, por exemplo, num sistema de vídeo sob demanda, onde o vídeo a ser enviado encontra-se previamente armazenado no servidor.

No caso de um sistema em tempo-real, o servidor enviará dados continuamente a medida que são gerados. A não ser que seja possível variar a taxa de captura de vídeo continuamente, a técnica “client driven” não é adequada, pela impossibilidade de ajuste da taxa de geração de dados às de pedidos dos clientes. Além disso a implementação de mecanismos de acesso de vários clientes fica por demais complexa. O mecanismo é também menos sensível a interrupções na rede, já que não há praticamente nenhuma informação de controle trafegando entre cliente e servidor.

Em ambas as técnicas é possível estabelecer táticas para o envio ou pedido de dados, seja no caso de vídeo pré-armazenado como em transmissões com alta interatividade [5, 12]. A estratégia pode levar em consideração a banda disponível e a variação desta com o tempo. O servidor também pode estabelecer prioridade de atendimento aos pedidos.

No aplicativo aqui descrito foi implementada inicialmente a técnica orientada ao cliente, de forma a verificar a sua viabilidade em um ambiente de alta interatividade. Embora esta técnica seja apropriada para um sistema de vídeo sob demanda (sendo inclusive a técnica usada no servidor de vídeo em operação no nosso laboratório), o mecanismo mostrou-se bastante inapropriado para transmissão com alta interatividade e onde a rede possui uma alta taxa de perda de pacotes. A técnica usada atualmente é então orientada ao servidor.

A Figura 1 ilustra a arquitetura da ferramenta de vídeo. Uma característica importante desta

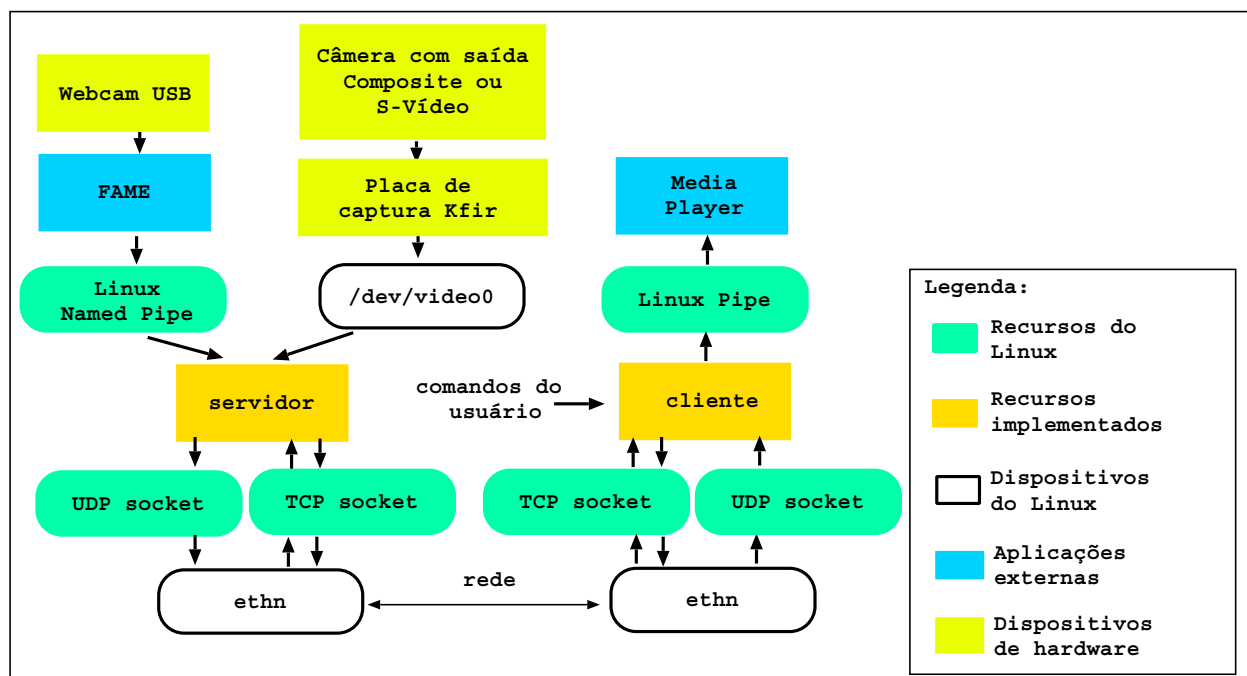


Figura 1: Arquitetura da ferramenta de vídeo.

ferramenta é a sua flexibilidade. Os dados a serem enviados podem ser obtidos a partir de uma webcam convencional, conectada à entrada USB do micro, ou a partir de uma câmera com saída Composite ou S-Vídeo, conectada a uma placa digitalizadora (no caso placa kfir [1]). Já no cliente, os dados podem ser exibidos por qualquer tocador de mídia (“media player”) compatível com os dados sendo transmitidos.

O método utilizado para se conseguir esta grande flexibilidade foi a utilização massiva de “pipes”, tanto por parte do cliente quanto no servidor. Os “pipes” são mecanismos de comunicação interprocessos (“interprocess communication”, ou IPC) bastante familiares entre os usuários de Linux. No entanto, existe um tipo especial de “pipes”, os “named pipes” [10], que não são tão populares mas que mostraram-se extremamente úteis na implementação do sistema em questão. Cabe destacar ainda a utilização de “buffers” no cliente e no servidor para redução de “jitter” e adequação dos fluxos de transmissão e consumo. (A Figura 2 ilustra o efeito de “buffers” na redução do “jitter”.)

Um dos principais objetivos da ferramenta é a coleta de estatísticas para avaliar a QoS obtida, e o controle sobre os dados, desde a captura e transmissão até o recebimento e a exibição. Assim, o nosso aplicativo permite determinar o tempo exato de captura de um quadro de vídeo, o instante em que este é enviado pela rede, o instante em que é recebido e o momento em que é exibido. Para a exibição dos vídeos MPEG-2 gerados a partir da placa de captura que usamos foi utilizado o programa MPlayer [3]. Para tocar vídeos no formato MPEG-1, foi utilizado o MTV Player [2].

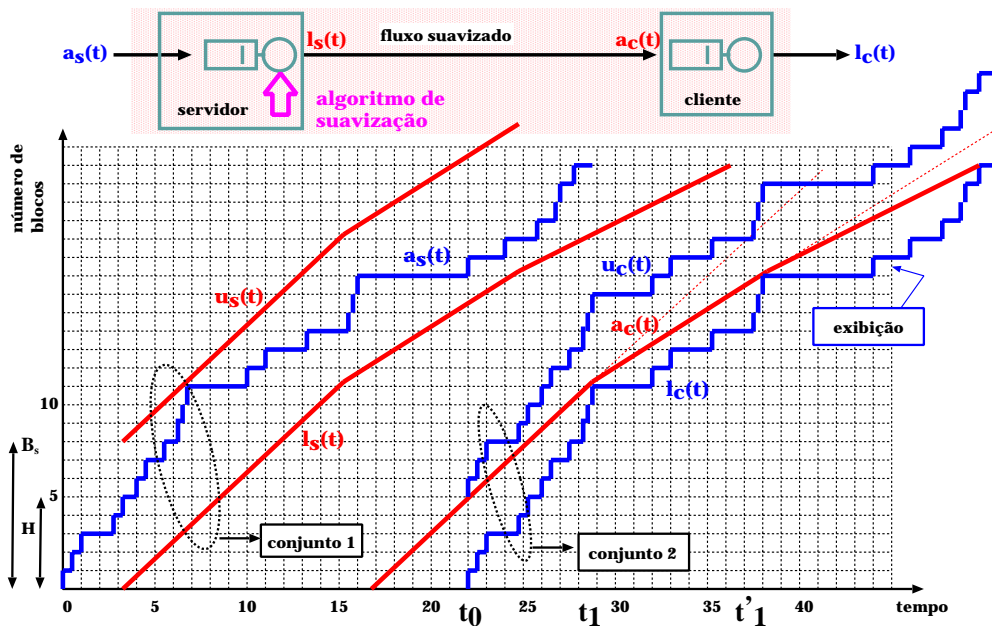


Figura 2: Transmissão (conjunto 1) e recepção (conjunto 2) de dados.

Como exemplo de coleta de estatísticas, suponha que se queira reproduzir um gráfico como o da Figura 2. É necessário, para tanto, saber o instante em que ocorreu a captura de um determinado quadro a partir da câmera de vídeo, o instante em que este foi enviado pela rede, além do momento em que foi recebido e tocado pelo cliente. Todas estas informações que são coletadas pelo aplicativo.

São gerados três arquivos de log pelo programa de vídeo, a saber:

- camera.server.sending-times.log, armazena o instante em que os pacotes foram enviados pela rede.
- camera.client.generation-playout.log, informa os instantes em que os pacotes foram capturados a partir do dispositivo de vídeo, e tocados no cliente.
- camera.client.req-rcv.log, apresenta os instantes em que os pacotes foram solicitados (aplicável somente à abordagem orientada ao cliente) e recebidos pelo do cliente. No caso da ferramenta orientada ao cliente, este arquivo pode ser utilizado, por exemplo, para um cálculo de estimativa do RTT.

Os tempos nos arquivos estão no formato segundos:microsegundos passados desde 0 horas de 1º de Janeiro de 1970 (formato retornado pela função gettimeofday, segundo padrão adotado no Unix). A Figura 3 ilustra os dados obtidos.

Nesta figura, é importante notar que o tempo de geração e de exibição são coletados por máquinas diferentes. Portanto, devido a problemas de “skew” e “offset” dos relógios [11, 13] das

0	00001039449619:586595
1	00001039449619:586754

Seq. num.	Generated at	Played at
0	00001039449619:586585	00001039449525:770182
1	00001039449619:586734	00001039449525:770248

Seq. num.	Requested at	Received at
0	n/a	00001039449525:159659
1	n/a	00001039449525:160989
2	n/a	00001039449525:162346

Figura 3: À esquerda, formato do arquivo de log `camera.server.sending.times.log`. À direita, formato dos arquivos de log gerados no cliente: `camera.client.generation-playout.log` e `camera.client.req-rcv.log`. Note que como foi utilizada a abordagem orientada ao servidor, a informação “Requested at” não se aplica (n/a, “not available”).

diferentes máquinas, é preciso tratar esses dados antes de serem utilizados. A ferramenta de geração de tráfego do TANGRAM-II [4] implementa algoritmos de remoção de “skew” e “offset”, e esses algoritmos devem ser utilizados no processo. Uma vez removidos os “skew” e “offset”, é fácil se obter um gráfico semelhante ao da 2. Atualmente “scripts” para a geração e visualização dos dados coletados estão sendo implementados.

A ferramenta de transmissão de vídeo ilustrada neste trabalho juntamente com o aplicativo VivaVoz têm sido utilizadas em cursos entre a COPPE/UFRJ e o Departamento de Computação da University of Massachusetts at Amherst. Em particular um curso de duração de um trimestre foi ministrado ao vivo pelo professor Don Towsley em 2002 para alunos do LAND/COPPE. Atualmente seminários regulares, sob a coordenação dos professores Don Towsley, Edmundo de Souza e Silva e James Kurose, estão sendo realizados, contando com a participação de alunos e professores dos grupos de ambas instituições. Durante as aulas, estatísticas estão sendo coletadas para posterior análise, de forma a que seja possível a melhoria das ferramentas de transmissão de acordo com os resultados coletados. O programa de vídeo assim como as demais ferramentas desenvolvidas pelo grupo do LAND estão disponíveis em www.land.ufrj.br.

Referências

- [1] URL <http://www.linuxtv.org/>. Site do LinuxTV.
- [2] URL <http://www.mpegTV.com/>. Site do MTV Player.
- [3] URL <http://www.mplayerhq.hu/>. Site do MPlayer.
- [4] A.A. de Aragão Rocha, E. de Souza e Silva, and R.M.M. Leão. Uma ferramenta para estimar características fim-a-fim na internet. In *submetido ao WRNP2/SBRC*, 2003.
- [5] E. de Souza e Silva, R.M.M. Leão, B.R. Neto, and S.V. Campos. *Performance Evaluation of Complex Systems: Techniques and Tools*, chapter Performance Issues of Multimedia Applications, pages 374–404. Springer-Verlag, 2002.
- [6] D.R. Figueiredo and E. de Souza e Silva. Efficient mechanisms for recovering voice packets in the internet. In *Proceedings of IEEE/Globecom*, pages 1830–1837, 1999.
- [7] J.F. Kurose and K.W. Ross. *Computer Networking: a top-down approach featuring the Internet (ISBN 0-201-47711-4)*. Addison-Wesley, 2001.
- [8] S. McCanne and V. Jacobson. vic: a flexible framework for packet video. In *Proceedings of ACM Multimedia*, pages 511–522, 1995.
- [9] A. Quevedo. Mecanismos para Garantir Qualidade de Serviço de Aplicações de Vídeo sob Demanda. Master’s thesis, COPPE/UFRJ, 2002.
- [10] R. Stevens. *Advanced Programming in the UNIX Environment*. Addison-Wesley, 1992.
- [11] M. Tsuru, T. Takine, and Y. Oie. Estimation of clock offset from one-way delay measurement on asymmetric paths. In *SAINT International Symposium on Applications and the Internet*, 2002.
- [12] D. Wu, Y. Hou, W. Zhu, Y. Zhang, and J. Peha. Streaming video over the internet: Approaches and directions. In *IEEE Trans. on Circuits and Systems for Video Technology*, pages 1–20, 2001.
- [13] L. Zhang, Z. Liu, and C. H. Xia. Clock synchronization algorithms for network measurements. In *IEEE/Infocom*, 2002.