Universidade do Minho - Dep.to Informática
Mestrado em Engenharia de Telecomunicações & Informática

Serviços de Rede & Aplicações Multimédia

Edição 2023-2024

Trabalho Prático N°2
Protocolo Transporte Multimédia

v1

Os protocolos de transporte de informação vídeo e/ou áudio digital na Internet devem ter um cariz especial porque devem permitir o consumo da informação no destino a um ritmo determinado pela fonte e com um atraso que deve ser o menor possível, dependendo do tipo de aplicação. Ou seja, o ritmo e o atraso da transferência de informação entre os componentes participantes num sistema multimédia devem poder ser controlados, o mais possível, pelos próprios componentes. Esta é uma das principais razões para que esses protocolos aplicacionais costumem ser encapsulados em UDP e não em TCP, como é o caso do RTP.

O modelo mais comum destes sistemas multimédia inclui um ou vários servidores de informação e vários clientes, normalmente em número muito superior. Também é comum os clientes incluírem um sistema de *bufferização* da informação (onde cabem até B *frames* de informação), estático ou dinâmico. A quantidade máxima de informação útil que poderá ser armazenada temporariamente no *buffer* depende do atraso máximo permitido pelo tipo de serviço e pela diferença máxima entre o ritmo a que a informação é transmitida na rede e o ritmo a que deve ser consumida no cliente. Isto é, o *buffer* serve como almofada que regula o ritmo da informação que chega da rede e o ritmo a que tem de ser consumida no cliente.

Objetivos

O principal objetivo do trabalho é a construção dum sistema simples de *streaming* entre um servidor e um cliente. O servidor gera a informação multimédia a um determinado ritmo (medido em *frames*/segundo) e transmite-a continuamente para o cliente que deve reproduzir a informação ao ritmo em que é gerada pelo servidor e com um atraso máximo também definido pelo servidor.

Para além destes dois componentes deve ser integrado um componente especial intermédio, um reencaminhador, que deve receber a informação do servidor e reenvia transparentemente (sem a alterar) para o cliente a um ritmo igual ou inferior ao ritmo de receção como forma de simular o atraso da informação introduzido pela rede. Assim, este componente simula o atraso da rede (adicionalmente também poderá simular a perda de PDU).

A informação multimédia gerada pelo servidor é um relógio digital em que cada *frame* de informação é um passo do relógio (um segundo, por exemplo). A cada passo do relógio o servidor constrói uma *frame* com a informação dos dígitos necessários para mostrar a hora do dia com a precisão que o passo determinar. O servidor deve ter uma biblioteca de imagens, uma para cada dígito, que deve usar para construir a informação de cada *frame*. Por exemplo, com um passo a cada segundo, a informação necessária para mostrar

a hora 21:33:55 é a sequência de seis imagens que representam os seis dígitos: 2,1,3,3,5,5. Esta sequência de seis imagens tem de estar contida numa única *frame* a ser enviada pelo servidor. Um segundo depois, o servidor enviará uma outra *frame* com a sequência de seis imagens dos dígitos 2,1,3,3,5,6, e assim por diante. A informação duma *frame* deve ser incluída num único PDU do protocolo a desenvolver. Quando o cliente receber um PDU, retira a informação da *frame* e coloca no seu *buffer* de entrada. Tendo em conta a informação dessa frame (e informação de controlo adicional que também esteja presente no PDU, como o ritmo de reprodução e o atraso máximo permitido), o cliente gere o buffer e determina o tempo exato em que deverá reproduzir a informação para o utilizador combinando as imagens dos dígitos da *frame*.

Seque-se uma lista dos objetivos para cada um dos três componentes do sistema multimédia:

- O servidor deve implementar um relógio com uma precisão dependente do parâmetro F (0,1,2,3,4, etc.), em que 10^{-F} indica a duração em segundos de cada passo do relógio (i.e., se F=0 então o relógio terá uma precisão de 1 segundo, se F=2 o relógio terá uma precisão igual a um décimo de segundo e assim por diante). A cada passo de relógio o servidor deve enviar a hora para o reencaminhador sob a forma duma *frame* de informação. Esta *frame* de informação é uma sequência dos dígitos que compõem a hora, com a precisão adequada ao valor de F (se F=0 a informação inclui HH:MM:SS, se F=1 a informação inclui HH:MM:SS:D, se F=2 a informação inclui HH:MM:SS:DC, e assim por diante). Cada dígito pode ser passado como uma imagem (num qualquer formato à escolha) ou como o próprio carater, i.e., '0', '1', etc. No primeiro caso, a informação a incluir na *frame* é a sequência de imagens (uma para cada dígito do relógio) e, no segundo caso, para além da sequência respetiva de carateres, deve ser incluída uma quantidade de informação irrelevante, mas que sirva de carga no PDU. As *frames* de informação devem ser transmitidas ao ritmo de 10^{F} *frames* por segundo, ou seja, a cada passo do relógio uma *frame* deve ser transmitida do servidor para o reencaminhador. No servidor também deve ser definido um parâmetro A (em segundos) que representa o atraso máximo que é aconselhado ao cliente na reprodução. O valor deste parâmetro deve ser incluído no PDU, tal como o de F. O valor de A define indiretamente a quantidade máxima de informação/*frames* que pode ser mantida em *buffer* no cliente antes da sua reprodução. Por exemplo, se F=1 e A=2, então o cliente pode manter no *buffer* a informação de até 20 *frames*.

- O reencaminhador deve receber as *frames* do servidor e deve reencaminha-las para o cliente de imediato. No entanto, a cada N *frames*, o reencaminhador deve fazer uma pausa de P segundos antes de voltar a reenviar e a cada M segundos deve ignorar um PDU e não reencaminhar. Enquanto está em pausa o reencaminhador deve guardar as *frames* recebidas num buffer (com um tamanho suficiente para conter todas as frames recebidas durante a pausa) para reenvia-las. O mais rapidamente possível, assim que a pausa acabar. Os parâmetros M, N e P são de configuração e uso exclusivo do reencaminhador.

- O cliente deve receber as *frames* de informação do reencaminhador (para o cliente o reencaminhador é que é o servidor) e reproduzi-las como um relógio digital com um passo/ritmo e precisão definidos pelo

parâmetro F incluído no PDU. Para ajudar nesta sincronização o cliente deve implementar um *buffer* com um tamanho dependente dos parâmetros F e A incluídos no PDU. O cliente deve implementar um algoritmo de ressincronizarão quando já não for possível manter a sincronia (quando os PDU chegam demasiado rápido ou demasiado lentamente durante muito tempo). Se os PDU que o cliente recebe contiverem a sequência de imagens com os dígitos é com estas imagens que deve ser apresentado o relógio ao utilizador. Se o cliente receber PDU com a sequência de carateres dos dígitos deve indicar o relógio imprimindo simplesmente a sequência de dígitos ou usando imagens ou outro tipo de animação construídas/armazenadas localmente, ignorando o resto da carga do PDU.

- Nenhum dos três componentes necessita de implementar qualquer interface interativo com o utilizador. Os parâmetros de configuração do servidor e do reencaminhador e os endereços de transporte (endereços IP + portas UDP) dos componentes podem ser passados por um ficheiro de configuração ou argumentos de arranque.

- Os componentes devem fornecer estatísticas de utilização e informação útil que permita a verificação do que está a acontecer em tempo real (ritmos de informação, atraso, *frames* perdidas, etc.).

Unidade Protocolar de Dados

Este protocolo especial de streaming só precisa funcionar num sentido (do servidor para o reencaminhador ou do reencaminhador para o cliente). Deve ser definido um PDU que contenha toda a informação de controlo e de dados necessária para implementar os requisitos da interação dos componentes definidos anteriormente. Além disso:

- O protocolo deve ser atómico, i.e., a informação duma *frame* deve caber num único PDU, não podendo ser dividida em mais do que um PDU.

- O protocolo é assíncrono, inseguro e não fiável, ou seja, não existem mecanismos de retransmissão, o ritmo de transmissão é definido pelo emissor e não existem mecanismos de encriptação da informação.

- Não é necessário que os campos do PDU sejam codificados duma forma otimizada, mas é preferível que a metodologia de codificação seja otimizada para binário ou usando um formato normalizado como o *Basic Encoding Rules* (BER).

- O PDU do protocolo deve ser definido e explicado em detalhe no relatório do trabalho.

- Cada PDU deve ser encapsulado num datagrama UDP.

Relatório e outras recomendações

O trabalho deve ser desenvolvido em grupos de, no máximo, dois alunos.

O código pode ser desenvolvido em C/C++, Rust, Java ou Python. Não use API, funções ou excertos de código de terceiros que não tenham sido fornecidos ou aprovados pelo docente.

O código deve ser claro e usar convenções de nomeação de variáveis, tipos, funções e constantes. O código deve ser estruturado duma forma o mais modular possível, sem complexidades desnecessárias, e permitir reutilização sempre que possível.

A qualidade e correção do código não se mede pelo seu tamanho.

Os alunos devem documentar/explicar o código criado através de comentários nas secções mais relevantes e no relatório. Todos os ficheiros do código devem ter um cabeçalho com a informação relevante que identifique os seus autores e explique as principais funções criadas, tipos de dados e variáveis usadas, etc.

O relatório deve incluir:

- Na primeira página, o título do trabalho e a identificação dos autores (incluindo fotografia), universidade, curso, unidade curricular e data de entrega;
- Um índice do conteúdo;
- Uma secção com a discussão das estratégias escolhidas, as opções tomadas e os mecanismos adotados, incluindo eventuais otimizações;
- Uma secção com a explicação e análise crítica (apenas) das principais funções implementadas, os seus principais méritos e a suas limitações mais importantes;
- Uma secção com a análise dos resultados dos testes efetuados, tentando detalhar as razões lógicas para os resultados encontrados;
- Uma secção de conclusões que inclua uma eventual discussão sobre o que gostava de ter feito melhor ou de ter acrescentado e não conseguiu;
- Uma lista de eventuais referências bibliográficas, artigos científicos ou recursos informais na web e que tenham sido úteis.

O relatório é para ser avaliado pelo docente por isso não se deve incluir informação genérica e irrelevante que o docente já conhece. Deve aspirar-se concisão e clareza. Sempre que se incluir uma afirmação importante no contexto do relatório e que seja de autoria de terceiros, ou que seja baseada diretamente em afirmações de terceiros ou concluída de informação retirada de recursos alheios, deve referenciar-se corretamente essas autorias ou proveniências e acrescenta-las na lista das referências.

O relatório pode incluir algumas partes relevantes do código quando estas ajudam às análises e justificações apresentadas. Não se deve integrar código desnecessário no texto do relatório. Sempre que possível, comente-se antes o próprio código.

Todo o material entregue deve juntar-se num único ficheiro zip. Este ficheiro zip deve conter um ficheiro PDF com o relatório e todos os ficheiros do código do projeto. O nome do ficheiro zip deve ser igual a SRAM-2023-2024-II-*Número_Aluno_A*-*Número_Aluno_B*.zip, como, por exemplo, SRAM-2023-2024-II-83974-87766.zip.

Por fim, é recomendável que, durante a defesa do trabalho, se responda honesta e concisamente apenas às questões colocadas por forma a que as sessões de apresentação não se arrastem para além dos 20-30 minutos.