

UNIVERSIDADE FEDERAL DE JUIZ DE FORA
INSTITUTO DE CIÊNCIAS EXATAS
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Streaming Adaptativo para Ambiente IPTV Móvel

Josimar Wilton Volpini

JUIZ DE FORA
DEZEMBRO, 2014

Streaming Adaptativo para Ambiente IPTV Móvel

JOSIMAR WILTON VOLPINI

Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Bacharelado em Ciência da Computação

Orientador: Marcelo Ferreira Moreno

JUIZ DE FORA
DEZEMBRO, 2014

STREAMING ADAPTATIVO PARA AMBIENTE IPTV MÓVEL

Josimar Wilton Volpini

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DO INSTITUTO DE CIÊNCIAS EXATAS DA UNIVERSIDADE FEDERAL DE JUIZ DE FORA, COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE BACHAREL EM CIÊNCIA DA COMPUTAÇÃO.

Aprovada por:

Marcelo Ferreira Moreno
Doutor em Informática

Eduardo Barrére
Doutor em Engenharia de Sistemas e Computação

Romualdo Monteiro de Resende Costa
Doutor em Informática

JUIZ DE FORA
09 DE DEZEMBRO, 2014

À minha mãe Aparecida Volpini.

Resumo

A presente monografia, intitulada “Streaming Adaptativo para Ambiente IPTV Móvel”, trata da transferência de fluxos contínuos de áudio/vídeo (streams) de um modo adaptativo, usando o serviço de IPTV em dispositivos móveis. Como tais dispositivos estão propensos a certas limitações e problemas, como conexões instáveis, é proposta uma solução que proporcione adaptabilidade, ou seja, o cliente se torna capaz de informar problemas, como perda de pacotes e/ou congestionamento na rede, causados devido à instabilidade na conexão. Estas informações são agregadas em um Componente de Adaptação, implementado em prova de conceito como um servidor à parte, independente do servidor e do cliente de streaming. Com base nas informações reportadas principalmente pelos clientes, o Componente de Adaptação é capaz de nortear os clientes de IPTV móvel a tomar a decisão correta e se adaptar de modo a continuar recebendo o serviço com a melhor qualidade possível no momento.

Há alguns modos de se conseguir tal adaptação, usando uma diversidade de protocolos e algoritmos. Especificamente no ambiente de IPTV móvel, por decisão deste projeto, são utilizados dois protocolos. O RTP (Real Time Transport Protocol) que envia pacotes de dados de conteúdo aos clientes e o RTCP (Real Time Control Protocol), que especifica os dados que podem ser intercambiados para reportar a qualidade das transmissões dos streams. Em especial, o RTCP tem papel de destaque na proposta, subsidiando as ações de adaptação tomadas pelo Componente de Adaptação.

Palavras-chave: Streaming Adaptativo, IPTV, Protocolo RTP, Protocolo RTCP.

Abstract

This work, entitled “Adaptive Streaming for Mobile IPTV Environment”, talks about the transfer of continuous fluxes of audio/video (streams) in an adaptive way, using IPTV services on mobile devices. As such devices are bound to certain limitations and problems, as unstable connections, proposes a solution that provides adaptability, in other words, the client become able to report problems, like the loss of packages and/or net jam, caused due connection instability. These information are put together into an Adaptation Component, implemented in proof of concept as a separated server, independent from the client’s server and streaming. Based on the reported information, specially by the clients, the Adaptation Component is able to direct the mobile IPTV’s clients to take the right choice and to adapt themselves in a way to keep receiving the service with the best quality available at the time.

There are some ways to achieve such adaptation, using a variety of protocols and algorithms. Specifically in the mobile IPTV’s environment, by this project’s decision, two protocols are used: RTP (Real Time Transport Protocol) which sends content’s data packages to the clients and RTCP (Real Time Control Protocol), which specifies the data that can be exchanged to report the quality of stream’s transmissions. In a special way, RTCP has a featured role in the proposal, subsidizing the adaptation’s actions taken by the Adaptation Component.

Keywords: Adaptative Streaming, IPTV, RTP Protocol, RTCP Protocol.

Agradecimentos

Agradeço, primeiramente, a Deus por ter me proporcionado o dom da vida e por sempre guiar meus passos em bons caminhos.

À minha querida mãe, que me acompanhou durante toda essa trajetória, me dando bons conselhos e acreditando sempre em mim, mesmo quando nem eu mesmo acreditava.

Ao meu Pai (in memoriam) que mesmo ausente esteve sempre presente em todos os momentos.

Aos meus irmãos e amigos pelo incentivo e apoio que sempre me proporcionaram.

Aos professores, pela paciência, dedicação e ensinamentos.

Enfim, agradeço a todos que, de algum modo, contribuíram para que eu chegasse até aqui.

Meu muito obrigado.

Sumário

Lista de Figuras	7
Lista de Tabelas	8
Lista de Abreviações	9
1 Introdução	10
1.1 Justificativa	11
1.2 Objetivos do trabalho	13
2 Streaming	15
2.1 Streaming Adaptativo	15
2.2 HTTP Live Streaming	17
2.3 Smooth Streaming	19
2.4 O Padrão MPEG-DASH	21
2.5 O Protocolo RTP	23
3 IPTV Móvel	28
3.1 Arquitetura do serviço de IPTV Móvel	28
3.2 Requisitos Funcionais da Arquitetura de IPTV Móvel	30
3.2.1 Requisitos funcionais de serviço	30
3.2.2 Requisitos funcionais de rede	31
3.2.3 Requisitos funcionais de QoS e QoE	31
3.2.4 Requisitos funcionais de dispositivos terminais	32
3.2.5 Requisitos funcionais de codificação de mídia	33
3.2.6 Requisitos funcionais de mobilidade	33
3.2.7 Requisitos funcionais de segurança	33
4 Reportando o desempenho de sessões RTP	34
4.1 Tipos de pacote RTCP	34
4.1.1 Pacote de relatório do remetente (SR)	35
4.1.2 Pacote de relatório do receptor (RR)	37
4.1.3 Pacote de descrição da fonte (SDES)	37
4.1.4 Pacote BYE	38
4.1.5 Pacote definido pela aplicação (APP)	38
4.2 Intervalo de transmissão RTCP	38
4.3 Atualização das variáveis do protocolo	40
4.4 Análise dos relatórios	41
5 Proposta de streaming adaptativo para IPTV Móvel	43
5.1 Implementação	46
5.1.1 Implementação do Servidor	46
5.1.2 Implementação do Cliente	49
6 Resultados Obtidos	52

7	Considerações Finais	59
	Referências Bibliográficas	61

Lista de Figuras

2.1	Arquitetura do HLS	18
2.2	MPD-DASH	22
2.3	Escopo do padrão MPEG-DASH	23
2.4	Formato dos pacotes RTP	26
3.1	Arquitetura de IPTV Móvel	29
5.1	Cenário proposto na implementação	43
6.1	Gráfico para Probabilidade P1	53
6.2	Gráfico para Probabilidade P2	54
6.3	Gráfico para Probabilidade P3	54
6.4	Gráfico para Probabilidade P4	55
6.5	Gráfico para Probabilidade P5	56
6.6	Gráfico Melhor Probabilidade	57

Lista de Tabelas

2.1	Comparação de Implementações de Streaming Adaptativo	21
-----	--	----

Lista de Abreviações

HTTP	Hypertext Transfer Protocol
IPTV	Internet Protocol Television
QoE	Quality of Experience
QoS	Quality of Service
RTP	Real Time Transport Protocol
RTCP	Real Time Control Protocol

1 Introdução

A cada ano, graças aos estudos e avanços tecnológicos, a Internet vem deixando de ser uma ferramenta só para pesquisas, troca de emails e outras funcionalidades comuns no seu início e vem ganhando novas utilidades. Uma dessas novas utilidades é a reprodução de áudio, vídeo, e outros conteúdos multimídia. Tecnologias mais recentes, como as que aplicam a técnica de streaming, possibilitaram aos usuários reproduzir conteúdos multimídia sem que para isso tivessem que fazer o download dos arquivos para só então visualizá-los. Atualmente, à medida que se faz o download de um conteúdo, o mesmo pode ser visualizado, sendo descartado logo em seguida (ou não, dependendo da situação). Isso proporcionou um grande avanço na distribuição e visualização de conteúdos multimídia pela internet.

Certos tipos de conteúdos multimídia possuem a característica de apresentação com restrições de tempo real, ou seja, o conteúdo deve ser transmitido de acordo com algumas propriedades de tempo, não permitindo assim atrasos ou apresentação desordenada. Para lidar com esse tipo de conteúdo, notadamente os que incluem mídias contínuas, foram desenvolvidos protocolos de tempo real como RTP e RTCP, que juntos proporcionam uma entrega de fluxos contínuos (streams) de forma mais adequada aos tipos de dados de mídias contínuas.

Contando com tais protocolos de tempo real, torna-se possível o oferecimento de serviços de IPTV, que fazem uso da técnica de streaming para distribuir conteúdo multimídia, tais como programas de TV ao vivo, vídeo sob demanda, aplicações interativas para seus clientes, entre outros. A União Internacional de Telecomunicações define IPTV como sendo serviços multimídia incluindo áudio, vídeo, imagens, textos, entregue por meio do protocolo IP em redes gerenciadas, para garantir os níveis necessários de qualidade de serviço (QoS), qualidade de experiência (QoE), segurança e interatividade (ITU-T Y.1910, 2008).

Avaliar QoS e QoE é uma tarefa complexa. A QoS é uma forma de avaliação objetiva baseada em parâmetros mensuráveis como atraso, jitter, perda de pacotes e

variação da largura de banda. Tais parâmetros representam as principais dificuldades enfrentadas na distribuição de conteúdo de áudio e vídeo em uma rede. A QoE é uma medida subjetiva do desempenho e um indicador de quão bem o sistema está atingindo seus objetivos, considerando-se a perspectiva do usuário. Para serviços de distribuição de vídeo, o mais importante é a medida perceptual da qualidade do vídeo. Muitos fatores afetam a QoE, como: fatores do ambiente (ruído, iluminação, etc), fatores da fonte (nível do som, bitrate, taxa de quadros, etc), fatores de distribuição (perda de pacotes, jitter, atrasos, largura de banda) e fatores do receptor (buffering, encobrimento de perdas e melhoras no controle do congestionamento) (GALVIS, 2013). Esse é um assunto complexo e, quando usado nessa monografia, subentende-se que se objetiva melhorar os parâmetros de QoS visando proporcionar aos usuários uma melhor percepção do conteúdo visualizado, deixando-o o melhor possível. Mais esclarecimentos sobre o assunto, pode ser encontrado em (GALVIS, 2013).

Nesse contexto, o grande avanço em hardware tem permitido também que vários dispositivos terminais possam ser usados para se acessar conteúdos multimídia, não ficando mais restrito a computadores de mesa convencionais. Computadores portáteis, celulares e outros dispositivos móveis têm sido muito usados para o consumo de conteúdo multimídia de tempo real. Entretanto, essa mobilidade introduz uma série de problemas, tais como capacidade limitada de processamento e armazenamento, telas de tamanho reduzido, restrições de energia, conexões instáveis, entre outros.

Assim, para viabilizar o oferecimento de serviços de IPTV em redes móveis, faz-se necessário o uso de mecanismos de adaptação que levem em conta as restrições impostas pela mobilidade, fornecendo qualidade de serviço (QoS) e qualidade de experiência (QoE) adequadas às necessidades e situações de momento de cada usuário.

1.1 Justificativa

A principal motivação para o presente trabalho está no fato de os ambientes móveis se apresentarem suscetíveis a variações relevantes na conectividade, levando à degradação da QoS necessária em serviços IPTV. Tais variações podem ser causadas pelo distanciamento do dispositivo terminal em relação às Estações Rádio Base (ERBs),

deslocamento entre diferentes ERBs, entre outros fatores que podem interferir no sinal em redes móveis.

Quando o dispositivo móvel se desloca de uma ERB para outra ocorre o que se denomina handover ou handoff (ROCHA, 2008). Existem alguns tipos de handover como: Soft Handover, Hard Handover, Seamless Handover, Horizontal Handover, Vertical Handover e SeamLess Vertical Handover. Cada um desses possui suas características próprias, podendo interferir no sinal das redes móveis em maior ou menor grau. O assunto não será detalhado, pois para o momento só é necessário saber que ele pode afetar a qualidade das redes móveis. Mais detalhes em (ROCHA, 2008).

Um dos problemas do serviço de IPTV em redes móveis é essa variação na qualidade do canal de transmissão, provocada pela instabilidade da conexão, fazendo com que haja momentos em que a vazão total de transmissão seja reduzida, afetando também outros parâmetros como retardo e variação estatística do retardo (jitter). Visando, portanto, aumentar a QoE do usuário e a eficiência do IPTV, é que se busca soluções para contornar esses problemas.

A falta de exibidores (players) de mídia que implementem os protocolos de IPTV de forma eficiente, aberta, e que ofereçam suporte adequado à adaptabilidade necessária podem ser citadas como principais barreiras ao uso do IPTV móvel com maior qualidade e popularidade.

As implementações existentes, como o serviço de IPTV da GVT, da Oi e da Vivo/Telefônica, trabalham com ambientes IPTV fixos. Isso faz com que não se leve em conta aspectos como variação de conectividade, por exemplo, já que os ambientes fixos, geralmente, apresentam certa estabilidade e controle de QoS facilitado.

Geralmente, softwares clientes de streaming de áudio/vídeo (players) implementam apenas dois dos protocolos (RTP e RTSP), não utilizando o RTCP. Mesmo quando o RTCP é utilizado, não tem toda a sua capacidade explorada. É necessário desenvolver aplicações que implementem formas de fazer com que as transferências de streams se adaptem a fatores externos que podem variar, prejudicando a qualidade dos serviços.

Esses fatores podem justificar a importância de se desenvolver uma aplicação que implemente os protocolos RTP e RTCP de maneira integrada, para que se possa explorar

a adaptabilidade, fazendo com que as transferências de mídia se adaptem às variações que sempre ocorrem nos ambientes móveis, de modo a conseguir maior qualidade e eficiência no uso do IPTV nesses ambientes.

Em (FROJDH et al., 2006) os autores apresentam o 3GPP Packet-Switched Streaming Service (PSS) que é um padrão para streaming de áudio e vídeo para terminais portáteis que se resume ao uso dos dois protocolos anteriormente citados, juntamente com um terceiro, o RTSP (Real-Time Streaming Protocol). É dada uma visão geral do padrão PSS e em particular ao suporte para streaming adaptativo. É dada também uma revisão detalhada das características das transmissões adaptativas. O esquema proposto deve ser capaz de melhorar substancialmente a qualidade das transmissões de streams.

No entanto, no artigo acima citado, os autores enfatizam que embora o PSS padronize os recursos que são necessários para implantar uma solução usando streaming adaptativo, não é imposto nenhum comportamento específico ao servidor e, assim, propõem para trabalhos futuros aumentar a lógica do controle de streaming no servidor.

1.2 Objetivos do trabalho

O objetivo geral da monografia é o estudo, o projeto e a implementação de um Componente de Adaptação, a ser adicionado a sistemas de IPTV móvel. O Componente de Adaptação deve monitorar a transmissão de conteúdo de áudio e vídeo de um servidor de mídia a um software cliente móvel, fazendo uso de relatórios periódicos enviados por esse software cliente. Através desses relatórios, o Componente de Adaptação pode analisar as transmissões e informar ao cliente quando se conectar a um fluxo de mídia com maior ou menor qualidade, conforme julgue necessário.

Como objetivos específicos, o presente trabalho inclui:

- Realizar um estudo sobre streaming, em especial sobre streaming adaptativo.
- Estudar e descrever o ambiente IPTV, com ênfase nos ambientes móveis, destacando seu funcionamento e principais características.
- Estudar o protocolo RTP e especialmente o RTCP, que é o protocolo fundamental para se obter subsídios para a adaptação dos streams.

- Implementar a lógica de adaptação de streams, permitindo assim adaptar a recepção às condições de cada usuário, possibilitando assim conseguir a QoE esperada nas aplicações.

Esta monografia está organizada da seguinte forma: o Capítulo 2 descreve as técnicas existentes para streaming adaptativo. O Capítulo 3 expõe as características de serviços de IPTV móvel. O Capítulo 4 descreve o protocolo RTCP, usado para o desenvolvimento da proposta deste trabalho, exposta no Capítulo 5. O Capítulo 6 mostra os testes realizados e os resultados obtidos e o capítulo 7 é reservado às considerações finais.

2 Streaming

Streaming é uma técnica que permite a entrega e a visualização de conteúdos de mídia contínua em tempo real. Esse conteúdo é dividido em pequenas partes e transferido aos clientes através de um fluxo contínuo usando um protocolo de tempo real, como o RTP (veja mais detalhes em 2.5).

Essas pequenas partes são armazenadas em um buffer que, a partir de certo momento, começa a ser consumido pelo software cliente para apresentação do conteúdo enquanto, simultaneamente, o mesmo continua sendo entregue. Caso haja um decréscimo nesse buffer, causado, por exemplo, por uma falha na conexão, a informação presente no buffer ainda é utilizada, proporcionando que o conteúdo ainda seja exibido por um curto período de tempo. Porém, caso esse buffer se esvazie a execução pára até a restauração da conexão. O conteúdo pode ser visualizado através de um player dedicado ou um plug-in presente em alguns navegadores. Esses, decodificam as informações e as exibe, descartando-as em seguida. Idealmente, essas informações são transmitidas mais rápido que reproduzidas, fazendo com que, nessas condições, sempre exista dados no buffer, possibilitando que a exibição do conteúdo continue mesmo se houver rápidas quedas ou decréscimos na conexão. (WILSON, 2014).

A transmissão dos streams pode ser ao vivo, quando são transmitidos ao mesmo tempo em que são gerados ou, ainda, sob demanda, quando são gravados com antecedência para posterior transmissão. Também podem ser transmitidos ponto a ponto, onde cada cliente tem seu próprio fluxo, ou em transmissões ponto-multiponto, nas quais o fluxo é multiplicado nos roteadores e enviado a várias estações de trabalho.

2.1 Streaming Adaptativo

Streaming adaptativo se baseia na técnica de streaming e permite a melhor experiência de visualização possível de conteúdos multimídia para cada usuário, uma vez que a transmissão do conteúdo se adapta às variações na rede. (OZER, 2011).

Na tecnologia de streaming um conteúdo ao vivo ou armazenado no servidor encontra-se codificado em uma determinada qualidade (resolução, taxa de bits, codec, etc). Já no streaming adaptativo, um mesmo conteúdo é disponibilizado em diferentes versões, cada uma com uma taxa de bits e níveis de qualidade diferentes, podendo ser entregue aos clientes, de acordo com parâmetros de desempenho como a taxa real de transferência, congestionamento da rede, perda de pacotes, ciclos de CPU disponíveis e outros aspectos. Cabe ao player ou ao servidor de streaming realizar a adaptação da qualidade da mídia a ser entregue, para manter níveis adequados de QoE.

Sem transmissão adaptável, as opções seriam a distribuição de uma única versão do conteúdo, de qualidade abaixo da média na configuração de visualização ideal ou seriam criadas várias versões e os clientes obrigados a selecionarem a versão desejada.

O streaming adaptativo fornece transparência ao usuário, já que este não precisa se preocupar com o monitoramento da rede ou com a adequação da transmissão a possíveis mudanças. O usuário pode apenas perceber alterações na qualidade, para melhor ou pior, dependendo da análise do monitoramento.

A maioria das tecnologias que usam streaming adaptativo possuem tanto características semelhantes como monitoramento do estado do buffer de vídeo e falta de quadros, usadas para determinar quando mudar o fluxo, quanto diferenças fundamentais como o envolvimento ou não de um servidor de streaming.

Assim, o streaming adaptativo permite que os provedores de conteúdo entreguem seus fluxos com a melhor qualidade possível para cada cliente em especial de acordo com suas necessidades e possibilidades.

Atualmente, com a necessidade de convergência de serviços rumo a World Wide Web (WWW), a maioria das técnicas de streaming adaptativo é implementada usando-se o protocolo HTTP. De forma interessante, estas técnicas evitam a modificação dos servidores WWW, concentrando as tarefas de adaptação no cliente.

No entanto, não somente o protocolo HTTP pode ser usado para streaming adaptativo. O próprio protocolo RTP, típico em sistemas IPTV, suporta diretamente adaptação de conteúdo.

O Real Time Transport Protocol (RTP) é um protocolo que define o formato de

pacotes de dados multimídia e faz o gerenciamento de cada sessão, o que permite entrega dos dados com baixo nível de overhead. Entretanto, na Internet não gerenciada, de melhor esforço, o RTP não é uma boa opção: não é suportado pelas redes de distribuição de conteúdo (CDN), firewalls costumam bloquear seus pacotes e é exigido que o servidor gerencie uma sessão separada para cada cliente. Essas questões, juntamente com o aumento da largura de banda da internet e o crescimento da WWW possibilitou o uso do streaming adaptativo HTTP.

A seguir, neste Capítulo, encontram-se descritas as variações mais populares de streaming adaptativo HTTP e o protocolo RTP.

2.2 HTTP Live Streaming

HTTP Live Streaming (HLS) (APPLE, 2014) permite enviar conteúdo multimídia, ao vivo ou sob demanda, de um servidor web comum para dispositivos cliente. Foi projetado visando dispositivos de diferentes resoluções de tela, suportando fluxos com taxas de bits diferentes, o que permite que o software cliente possa alternar entre esses fluxos de maneira inteligente e dinâmica, se adaptando a mudanças como a largura de banda da rede, proporcionando assim sempre o fluxo da melhor qualidade possível. Isso representa uma vantagem sobre o download progressivo, única forma até então possível para transferência de vídeo na WWW. HTTP Live Streaming é indicado para transferir streaming de mídia para aplicativos iOS, plataformas com suporte do Quicktime X, ou sites baseados em HTML5.

O principal motivo de se usar HTTP padrão para entregar a mídia, é que o mesmo é suportado por quase todos os servidores, distribuidores de mídia, sistemas de cache, roteadores e firewalls.

O conteúdo multimídia é entregue em uma série de pequenos arquivos, com cerca de 10 segundos de duração, chamados arquivos de segmentos de mídia. É produzido então um arquivo de índice ou lista de reprodução, que dá aos clientes informações sobre os arquivos de segmentos de mídia.

O HLS oferece suporte à segurança através da autenticação de usuários e criptografia de mídia. Quando se criptografa uma mídia, é gerada uma chave de decodificação

podendo ser uma única chave para todos os fluxos, uma chave diferente para cada fluxo, ou uma série de chaves geradas de forma aleatória que mudam periodicamente.

A arquitetura do HLS é composta pelos componentes do servidor, componentes de distribuição e o software do cliente, conforme ilustra a figura a seguir.

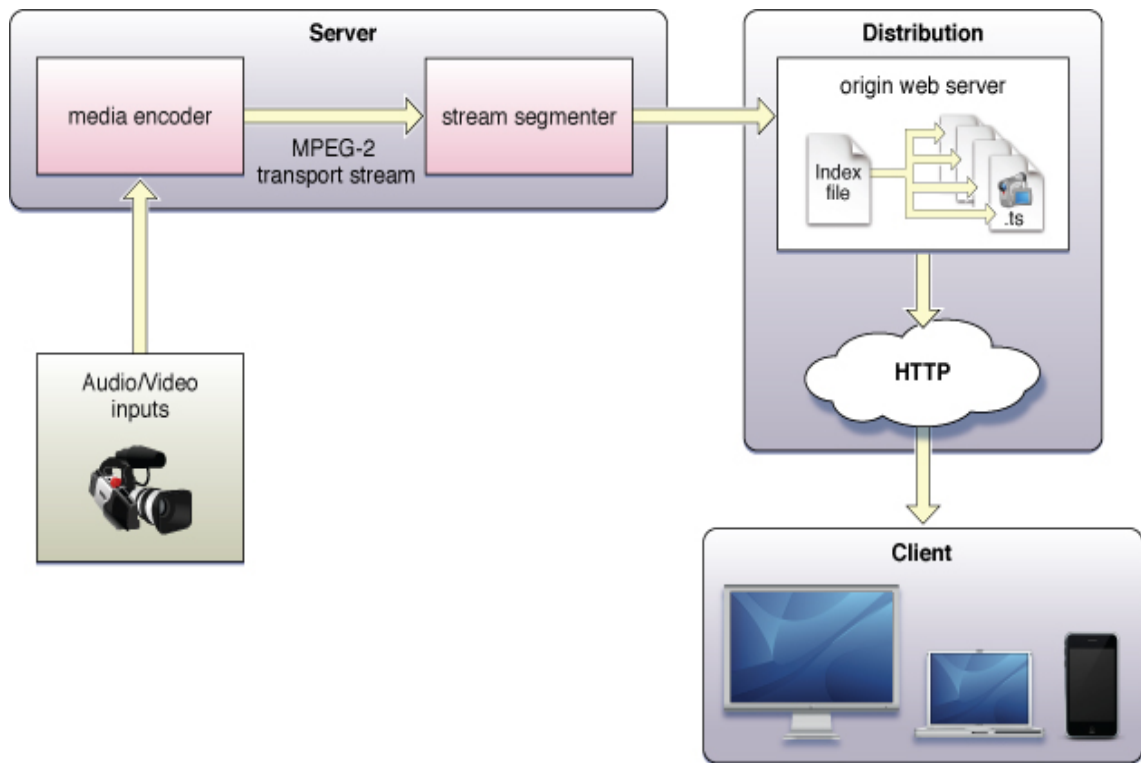


Figura 2.1: Arquitetura do HLS

Os componentes do servidor codificam os fluxos de entrada, os encapsula em um formato adequado e prepara o conteúdo para ser distribuído. Esses componentes são: o codificador de mídia, que codifica as mídias num formato compatível com os dispositivos cliente suportados e as incorpora para o transporte. O segmentador de fluxo, que divide o fluxo em uma série de pequenos arquivos e cria um arquivo de índice contendo referências aos arquivos de mídia individuais. O segmentador de arquivo, que compacta o arquivo de mídia codificado em um fluxo de transporte MPEG-2 e o divide em segmentos de igual comprimento. O arquivo de segmentos de mídia, consistindo de uma série de arquivos, cada um contendo um segmento de mídia. O arquivo de índices, que especifica a localização dos arquivos de mídia disponíveis, chaves de decodificação e qualquer fluxo alternativo disponível, com o mesmo conteúdo em diferentes níveis de qualidade.

O componente de distribuição, servidores web ou sistemas de cache web, fornecem os arquivos de mídia e os arquivos de índices aos clientes através de HTTP. Esse componente recebe solicitações dos clientes e entrega o conteúdo multimídia aos mesmos. Deve-se usar um validador de fluxo para garantir que os mesmos são totalmente compatíveis com HTTP Live Streaming.

O software cliente pode ser uma página HTML ou um aplicativo que determina os meios de comunicação, busca e seleciona no arquivo de índice a URL que identifica o fluxo e faz o download de cada arquivo de mídia sequencialmente. A partir do momento que o cliente recebe uma certa quantidade de dados, a apresentação é iniciada e o download continua sendo feito simultaneamente.

O HTTP Live Streaming suporta transmissões ao vivo ou vídeo sob demanda (VoD).

Quando se transmite vídeos sob demanda, os arquivos de mídia representam a apresentação completa. O arquivo de índices é estático e contém uma lista completa de todo o conteúdo. Há vantagens como suporte à criptografia e troca dinâmica entre fluxos com taxas de dados diferentes.

Já em sessões ao vivo, novos arquivos de mídia são criados e disponibilizados a todo momento, o que torna o arquivo de índices dinâmico. Podem-se descartar arquivos de mídia antigos ou pode-se simplesmente ir adicionando os novos arquivos de mídia no arquivo de índices sem descartar os antigos, o que converte a apresentação ao vivo em vídeo sob demanda quando a apresentação termina.

2.3 Smooth Streaming

Smooth Streaming (SS) (MICROSOFT, 2014) é uma extensão do Internet Information Service (IIS), desenvolvida pela Microsoft, que permite adaptação de streaming de mídia baseado na Web para o Silverlight e outros clientes que usam o padrão HTTP. Proporciona uma experiência de visualização que se adapta às condições das redes de distribuição de conteúdo.

Tal qual outras propostas de streaming adaptativo HTTP, o conteúdo é codificado em vários níveis de qualidade, cada um desses contendo o arquivo completo. Esse conteúdo

é entregue por um servidor habilitado para Smooth Streaming. São criados pequenos fragmentos consecutivos de áudio e vídeo, cada um contendo o nome do conteúdo, a taxa de bits, sua resolução, a duração do fragmento, um timestamp que aponta para o início do fragmento correspondente, entre outras informações.

Ao receber uma solicitação, o servidor entrega os fragmentos ao usuário final, monitorando, dinamicamente, a largura de banda local e o desempenho da exibição do conteúdo, verificando se cada fragmento chegou no tempo certo e foi reproduzido com a qualidade esperada. Em caso negativo, os fragmentos seguintes serão entregues com um nível menor ou maior de qualidade, dependendo do que foi constatado no monitoramento. Isso permite que se experimente uma reprodução da mais alta qualidade disponível, evitando falhas ou interrupções no fluxo.

Aos criadores de conteúdo, é oferecido implementação de código livre, gerenciamento de conteúdo simplificado e codificação, tanto para vídeos sob demanda quanto para transmissões ao vivo. Aos provedores de conteúdo, fornece a opção de poder usar o IIS Media Services para servir os Smooth Streams codificados. Para os consumidores de conteúdo, oferece a experiência de visualização de vídeo, com confiabilidade e qualidade, por meio de um cliente Smooth Streaming compatível.

Abaixo, uma tabela compara alguns aspectos entre o Smooth Streaming e o HTTP Live Streaming (AKNOWLTON, 2010).

Tabela 2.1: Comparação de Implementações de Streaming Adaptativo

	Smooth Streaming	HTTP Live Streaming
Custo por servidor de Streaming	Download grátis + \$469 por OS	Custo de um servidor web + segmentador OS
Streaming ao vivo e sob demanda	Dá suporte	Dá suporte
Protocolo de Streaming	HTTP	HTTP
Entrega para dispositivos móveis	Dá suporte	Dá suporte
Suporte a servidor 64 bits nativo	Dá suporte	Dá suporte
Container de Mídia	MPEG 4 - Part 12 (MP4 fragmentado)	MPEG-2 TS
Taxa máxima de bits	Sem limite	1.6 Mbps
Comprimento padrão do fragmento	2 segundos	10 segundos
Tipo de arquivo no servidor	Contínuo	Fragmentado

2.4 O Padrão MPEG-DASH

O DASH (Dynamic Adaptative Streaming over HTTP) (SODAGAR, 2011) é um padrão baseado em HTTP, que se destina à transmissão de dados multimídia através da Internet. Desenvolvido pelo MPEG (Moving Picture Expert Group), tem o intuito de fornecer interoperabilidade entre dispositivos e servidores de diferentes fornecedores para tornar as transmissões de streaming mais eficientes.

Há muitas implementações comerciais, como as descritas nas Seções anteriores, que usam HTTP streaming, porém cada uma possui formatos de metadados e de arquivos de segmento diferentes. Assim, é necessário fornecer interoperabilidade entre essas implementações.

Como toda solução de streaming adaptativo HTTP, o DASH apresenta vantagens como: suporte eficiente pela Internet, é amigável a firewalls e não é necessário manter o estado da sessão no servidor, permitindo atender vários clientes ao mesmo tempo sem custo adicional nos recursos do servidor, além de poder ser gerenciado por uma Rede de Distribuição de Conteúdo (CDN).

No MPEG-DASH, o conteúdo é capturado, armazenado e entregue pelo servidor e

é composto pela Descrição de Apresentação da Mídia (MPD) e pelos segmentos contendo os bitstreams.

O MPD é um documento XML (eXtensible Markup Language) que descreve os componentes de mídia de um conteúdo. É composto por períodos, que são intervalos de programa ao longo do tempo. Cada período possui hora de início, tempo de duração e é composto por conjuntos de adaptação. Cada um desses conjuntos é composto por múltiplas representações, que são codificações alternativas do mesmo componente de mídia e variam a taxa de bits, resolução e/ou outras características. Cada uma dessas representações é constituída por segmentos. Esses segmentos são pequenos pedaços do fluxos de mídia em sequência temporal e são definidos como o corpo da entidade de resposta do cliente DASH ao HTTP GET.

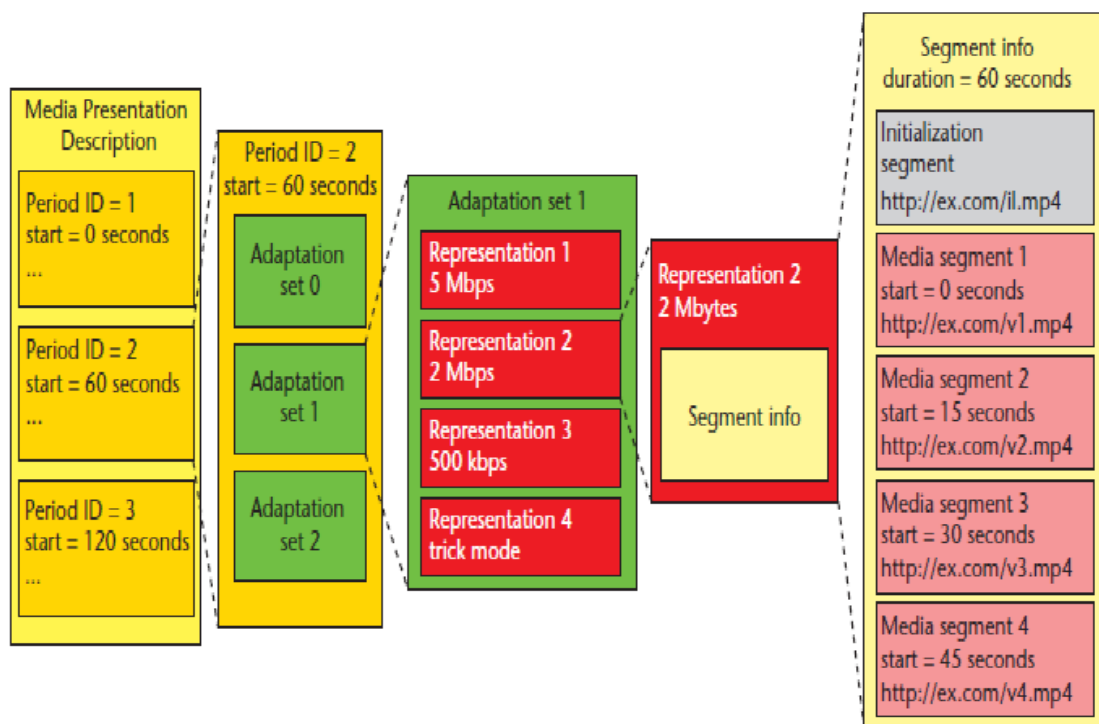


Figura 2.2: MPD-DASH

Após codificada, a componente de mídia é fragmentada nesses segmentos, sendo o primeiro, geralmente, apenas de inicialização. O fluxo de mídia é formado por segmentos consecutivos, que possuem uma URL, um índice, hora de início e tempo de duração.

Para iniciar a reprodução, o cliente obtém e analisa o MPD e, com base nessa análise, seleciona o conjunto de representação a ser usado, baseando-se, além do MPD,

nas capacidades do cliente e em possíveis escolhas do usuário. Feito isso, o cliente começa a receber os segmentos, buscando-os por meio de solicitações HTTP GET. O cliente busca os segmentos enquanto monitora as flutuações na largura de banda da rede e, com base nesse monitoramento, decide como se adaptar à largura de banda buscando segmentos de diferentes alternativas. Um buffer permite variações na taxa de transferência e retardo da rede.

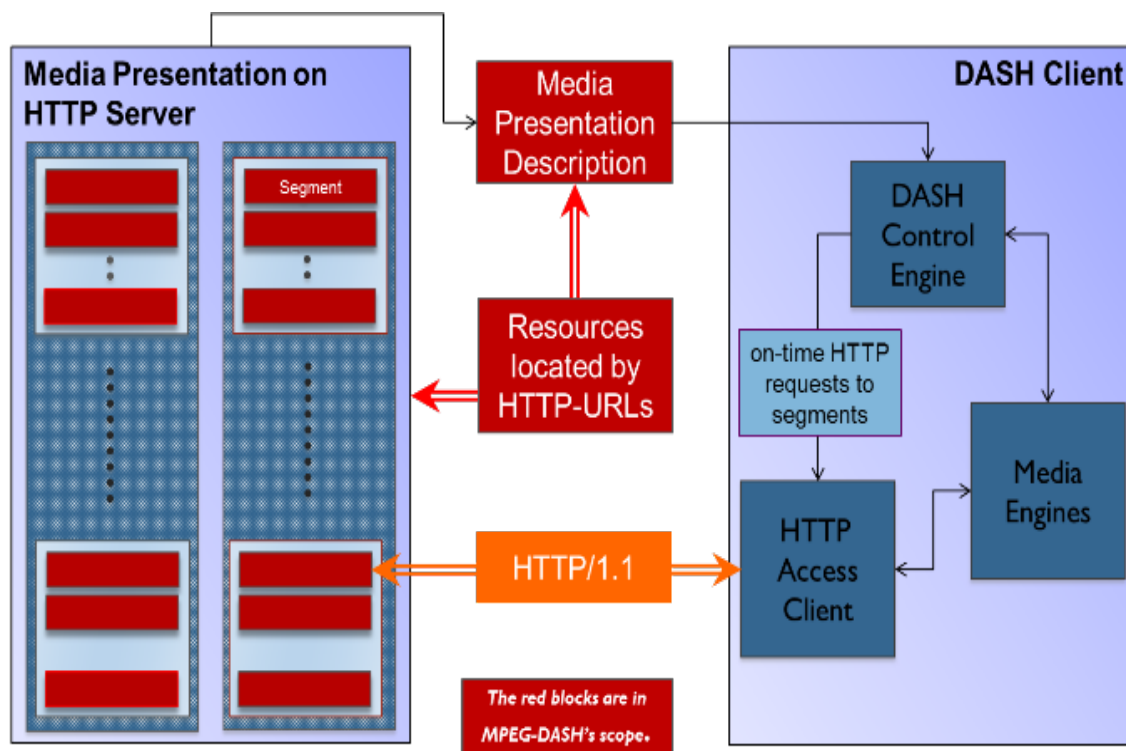


Figura 2.3: Escopo do padrão MPEG-DASH

MPEG-DASH possui vários recursos adicionais tais como: Fluxos alternativos selecionáveis e comutação entre os mesmos; Inserção de anúncios; MPD compacto e fragmentado; Segmentos com duração variável; Múltiplas URLs de servidores; Métricas de qualidade que relatam a experiência do usuário na sessão.

2.5 O Protocolo RTP

O protocolo RTP (SCHULZRINNE et al., 2003) presta serviços de entrega de dados com características de tempo real, como áudio e vídeo, para um único destinatário usando unicast ou para múltiplos destinatários usando distribuição multicast. Normal-

mente é executado sobre UDP, não sendo, entretanto, uma exigência.

O RTP envia pacotes de dados tipados, mas não assume que a rede seja confiável. Como resultado, não há garantia de entrega dos pacotes e nem que os mesmos, quando entregues, estejam ordenados. Portanto não há garantia da qualidade do serviço. Devido ao fato de que os pacotes podem ser entregues fora de ordem, com atrasos ou mesmo serem perdidos, o cabeçalho de cada pacote RTP contém informações de tempo e um número de sequência, permitindo que os receptores reconstruam o fluxo produzido por cada fonte de dados. O número de sequência também pode ser utilizado para estimar o número de pacotes perdidos.

Para cada fluxo, o protocolo usa um endereço IP (unicast) ou um endereço de grupo multicast e um par de portas, uma para dados de mídia, e outra para os pacotes RTCP, que serão vistos adiante, tendo em vista que os dois protocolos se “completam”, embora nem sempre estejam disponíveis juntos em implementações comerciais. Uma porta é a “abstração que protocolos de transporte utilizam para distinguir entre os vários destinos dentro de um determinado computador host” (SCHULZRINNE et al., 2003). E o endereço é a combinação de um endereço de rede e a porta que identifica um endpoint de nível de transporte. Essas informações (endereço e portas) são distribuídas aos participantes da comunicação, juntamente com uma chave de criptografia (para o caso dos pacotes de dados e de controle serem criptografados).

Quando há fluxos simultâneos, cada um é tratado como uma sessão separada, usando a informação de tempo para sincronizá-las, se os relógios das origens forem sincronizados. Assim, os participantes têm a opção de escolher apenas um tipo de conteúdo (áudio ou vídeo). Para associar as sessões, o participante deve usar o mesmo nome canônico (CNAME) distinto nos pacotes de reporte RTCP, em ambas as sessões.

Como RTP suporta aplicações multicast, adaptações na taxa de transmissão segundo a capacidade do receptor ficaria dificultada, uma vez que cada fluxo possui vários receptores. Para isso, o RTP conta com o conceito de codificação em camadas. Fluxos RTP multicast adicionais carregariam cada uma das camadas de melhorias. Assim, os receptores podem se adaptar à heterogeneidade da rede juntando apenas o subconjunto apropriado dos grupos multicast.

Pode ser necessário também o uso de multiplexadores e tradutores associados com o protocolo. Um multiplexador é usado geralmente em pontos onde a largura de banda da rede é menor. Ele recebe pacotes de mídia de várias fontes e ressincroniza esses pacotes gerando um único com as mídias reconstruídas. O multiplexador mantém uma lista com os identificadores das fontes que contribuíram para a construção desse pacote. Assim, as informações chegam aos receptores certos. Os tradutores, por sua vez, são usados contra bloqueio de pacotes (em um firewall, por exemplo). Um tradutor é instalado em cada lado do bloqueio. Um recebe o pacote e o traduz para um formato que é capaz de passar no bloqueio e o outro recebe o pacote do primeiro e o traduz novamente para a forma que o receptor entende.

No protocolo RTP, é possível fazer sincronização de diferentes fluxos no receptor, uma multiplexação identificada pelo endereço de transporte de destino (endereço de rede e número da porta). Para que se consiga uma maior eficiência, deve-se manter apenas um pequeno número de pontos de multiplexação. Fluxos separados devem ser transportados em sessões separadas e sincronizados com base nos campos tipo de carga ou identificação de sincronismo de fonte (SSRC). Intercalar pacotes com diferentes tipos de mídia (para adaptação) utilizando o mesmo SSRC irá introduzir vários problemas, tais como não teria como identificar os fluxos que mudaram suas codificações, um multiplexador RTP não combina fluxos intercalados de mídia incompatíveis em um único fluxo, entre outros problemas.

Enviar todos os fluxos RTP, mas usando um SSRC diferente para cada mídia, resolveria alguns problemas, mas não todos, embora os não resolvidos nem sempre se aplicam. Para que isso seja possível, a sincronização entre relógios de origens diferentes deve ser feita com ajuda externa. O RTCP pode ser usado neste caso.

Os pacotes RTP possuem o formato mostrado na figura a seguir:

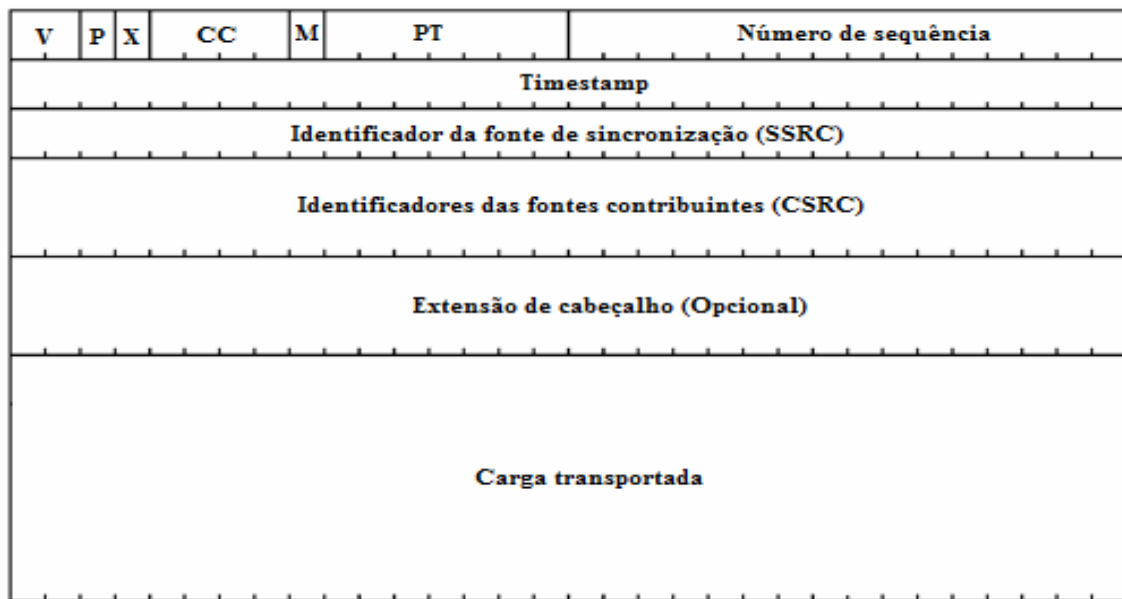


Figura 2.4: Formato dos pacotes RTP

Os pacotes RTP são formados por um cabeçalho de 128 bits divididos entre uma parte obrigatória de 96 bits, presente em todos os pacotes, e uma lista de fontes contribuintes, possivelmente vazia (inserida por um multiplexador), de 32 bits, uma extensão de cabeçalho (opcional), e os dados a serem transportados. O cabeçalho é composto pelos seguintes campos: versão (2 bits), preenchimento (1 bit), extensão (1 bit), contador CSRC (4 bits), marcador (1 bit), tipo de carga útil (7 bits), número de sequência (16 bits), timestamp (32 bits), SSRC (32 bits), lista CSRC (0 a 15 itens com 32 bits cada).

O campo versão (V) identifica a versão atual do protocolo. O campo preenchimento (P) indica se um pacote contém, ou não, octetos adicionais no final, sendo que o último octeto indica quantos devem ser ignorados, incluindo ele próprio. O campo extensão (X) indica se o pacote possui, ou não, uma extensão. O campo contador CSRC (CC) dá o total de identificadores CSRC, que são fontes de um fluxo de pacotes RTP que contribuem para o fluxo combinado produzido por um multiplexador RTP. O campo marcador (M) é utilizado para permitir eventos significativos, tais como limites dos quadros a serem marcados no fluxo de pacotes. O campo tipo de carga útil (PT) identifica o formato das cargas e determina sua interpretação pela aplicação, podendo ser mudado durante uma sessão. O campo número de sequência é um inteiro com valor inicial aleatório,

incrementado a cada pacote e é útil para detectar perdas e/ou restaurar a sequência dos pacotes desordenados. O campo timestamp define uma marca de tempo que indica o momento que o primeiro octeto foi inserido no pacote de dados e deve ser obtido de um relógio que cresce monotonicamente e linearmente no tempo permitindo calcular jitter e obter sincronização de pacotes. O campo SSRC define um inteiro aleatório que identifica a fonte de sincronização, assim não pode haver duas fontes com o mesmo número e caso a fonte mude o endereço de transporte deve ser gerado um novo identificador. O campo lista CSRC identifica cada uma das fontes contribuintes definidas no campo contador CSRC.

Embora o cabeçalho dos pacotes consigam dar suporte à maioria das aplicações, pode ser necessário fazer modificações e/ou adições em especificações particulares de algum perfil. Modificar os octetos definidos para os campos fixos do pacote ou adicionar uma nova funcionalidade a uma classe particular de aplicações são exemplos dessas adições e modificações que podem ser feitas.

3 IPTV Móvel

A União Internacional de Telecomunicações define IPTV como sendo serviços multimídia incluindo áudio, vídeo, imagens, textos, entregues por meio do protocolo IP em redes gerenciadas, para garantir os níveis necessários de qualidade de serviço (QoS), qualidade de experiência (QoE), segurança e interatividade (ITU-T Y.1910, 2008).

O IPTV móvel usa o serviço de IPTV e se baseia na conexão de um dispositivo terminal móvel com um provedor de serviços IPTV. Dessa forma, é possível se comunicar e acessar esses serviços independentemente de mudanças na localização e no ambiente técnico desses dispositivos. Essas mudanças acarretam problemas em aspectos como recursos de rede, características e configurações dos dispositivos e políticas entre a rede local e as redes visitadas, podendo afetar a QoS, cabendo a esses serviços se adaptar a essas mudanças para garantir a QoS (ITU-T, 2013).

Várias questões técnicas merecem atenção, podendo-se destacar: garantir a continuidade do serviço quando o dispositivo se move, visando minimizar o impacto sobre a QoS e QoE; Suporte à QoS no transporte através da rede, que possui recursos compartilhados e limitados; Adaptabilidade às mudanças dinâmicas na rede, uma vez que os recursos e os estados são dinâmicos; Apoio à heterogeneidade e portabilidade dos dispositivos terminais.

3.1 Arquitetura do serviço de IPTV Móvel

Podemos destacar quatro componentes principais na arquitetura de IPTV móvel: provedor de conteúdo, provedor de serviços, provedor de rede e dispositivos terminais móveis (ITU-T, 2013).

Os provedores de conteúdo são responsáveis por armazenar e fornecer conteúdo através dos servidores de conteúdo. Os provedores de serviço são responsáveis por gerenciar e controlar os serviços através dos controladores de serviços. Os provedores de rede são responsáveis por oferecer serviços de IPTV a cada região de serviço por meio

de dispositivos de rede. Os dispositivos terminais móveis são responsáveis por receber e consumir serviços no lado do usuário final.

A rede móvel é constituída por células cobertas por um único ponto de acesso e fornece serviços através de uma conexão sem fio entre esse ponto de acesso e os dispositivos terminais. Essas células são reagrupadas em regiões de serviço. Assim, a rede é composta por diferentes regiões de serviço, sendo que cada uma dessas regiões possui estações base ou pontos de acesso, chamadas estações base dos membros.

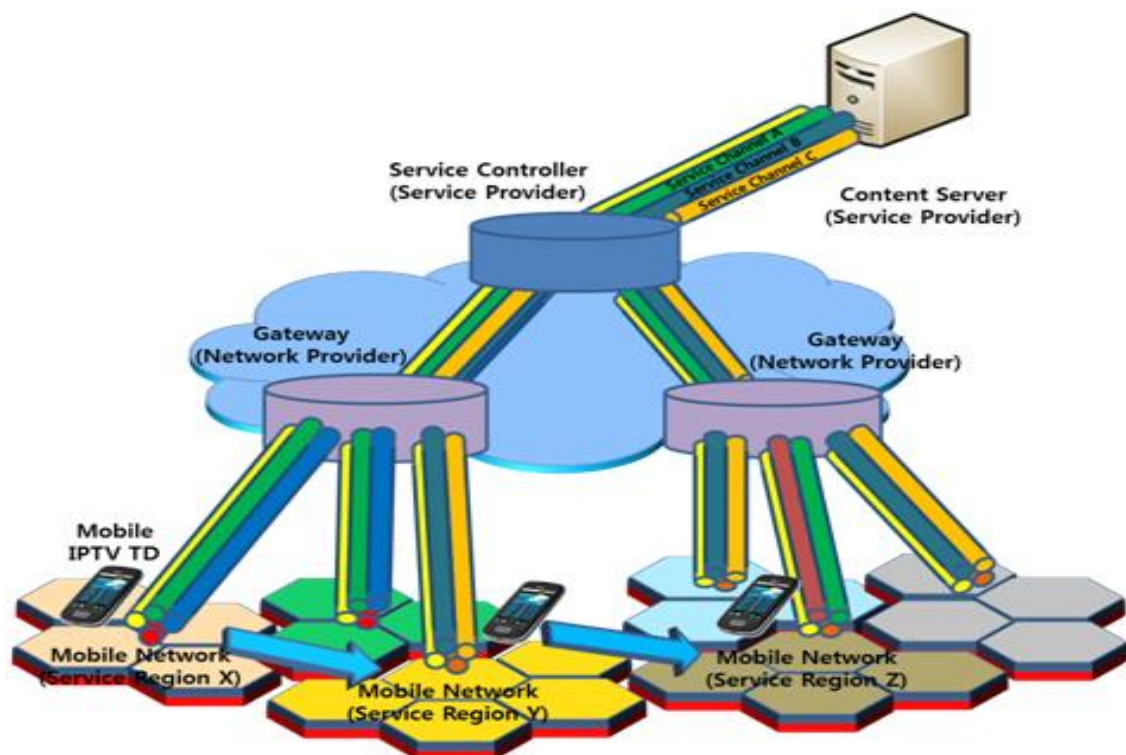


Figura 3.1: Arquitetura de IPTV Móvel

As estações base dos membros são escolhidas pelo controlador de serviços, que pode criar, atualizar e/ou remover uma região de serviço, podendo também alocar recursos de rádio para cada uma dessas regiões, nas quais os serviços podem ser distribuídos utilizando mecanismos como broadcast (região de serviços broadcast) e multicast (região de serviços multicast).

A configuração das regiões de serviço é feita quando se utiliza multicast ou broadcast, mais importantes em ambientes sem fio. Utilizando-se unicast, a configuração não é obrigatória sendo também difícil acomodar o tráfego do serviço, já que um provedor de serviços pode fornecer muitos canais de alta qualidade ao mesmo tempo.

O dispositivo terminal pode mover-se em uma mesma região ou para regiões vizinhas. No primeiro caso, não há descontinuidade do serviço, já no segundo a descontinuidade ocorre ao cruzar a fronteira de regiões diferentes (handover). Para esses dispositivos acessarem serviços prestados através de canais broadcast/multicast, as informações de broadcast/multicast devem ser fornecidas aos mesmos.

3.2 Requisitos Funcionais da Arquitetura de IPTV Móvel

Para garantir o bom funcionamento do serviço de IPTV móvel, garantindo uma QoS e QoE satisfatórias, deve-se atentar a uma série de requisitos que podem ser divididos em algumas categorias principais como: Requisitos funcionais de serviço, requisitos funcionais de rede, requisitos funcionais de QoS e QoE, requisitos funcionais de dispositivos terminais, requisitos funcionais de codificação de mídia, requisitos funcionais de mobilidade e requisitos funcionais de segurança (ITU-T, 2013).

Nos próximos tópicos veremos mais sobre cada um desses requisitos.

3.2.1 Requisitos funcionais de serviço

A arquitetura de IPTV móvel deve fornecer metadados com informações como tipo de serviço, dispositivo terminal utilizado, meio de transmissão dos dados, preferências do usuário, níveis de QoS disponíveis, entre outros. Além disso, deve dar suporte a APIs de middleware aberto para apoiar o desenvolvimento eficiente de aplicações.

É recomendado o uso de codecs para que se mantenha uma qualidade mínima no caso de perda de dados. O nível de proteção contra erros em tempo real deve ser adaptado ao ambiente de rede sem fio móvel. Esquemas de codificação de mídia devem ser negociados visando selecionar um esquema apropriado à cada dispositivo. Usar esquemas de codificação de vídeo escalável na entrega de conteúdo. Fornecer middleware para o dispositivo terminal de IPTV móvel. A interface do usuário deve apresentar uma interface gráfica eficiente com características de acordo com a capacidade de cada dispositivo.

3.2.2 Requisitos funcionais de rede

A arquitetura de IPTV móvel deve permitir o uso eficiente de broadcast, multicast, unicast e dos recursos de rádio na entrega, além de sincronizar os conteúdos de transmissão na mesma região de serviço, dar suporte à transferência sem descontinuidade entre regiões de serviço e ainda poder alternar entre os modos multicast/broadcast e unicast.

Deve-se permitir selecionar pontos de acesso que formam as regiões de serviço e transmitir informações a cada um desses pontos, além de gerenciar e configurar essas regiões, permitindo acrescentar ou eliminar pontos de acesso, bem como distribuir informações associadas com cada região.

Nessas regiões o estado de cada dispositivo associado deve ser controlado. Deve ser capaz de escolher e dar suporte aos serviços escolhidos e oferecer o mesmo conteúdo para diferentes dispositivos terminais, considerando limitações de recursos desses dispositivos.

É necessário ter controle sobre o congestionamento da rede e sobre o fluxo de dados para melhor eficiência dos serviços. Deve ser dado suporte à conectividade da rede móvel e suportar as operações de múltiplos compartimentos para aumentar a taxa de dados. Deve ser possível: transmitir simultaneamente um serviço usando múltiplos modos de transmissão; apoiar a mobilidade do terminal com a continuidade do serviço; apoiar robustez na entrega de conteúdos para as altas taxas de perdas de pacotes e a prestação contínua de serviços em diversas regiões de serviço.

3.2.3 Requisitos funcionais de QoS e QoE

Ao estabelecer uma sessão, há uma troca de parâmetros de QoS entre dispositivos e servidores, e nessa troca deve haver suporte a aspectos como estado da rede, capacidade do dispositivo e requisitos do usuário. Ao se conectar, o usuário recebe informações de QoS e quando uma mudança de QoS é solicitada por esse usuário, o servidor de gerenciamento verifica a possibilidade e, quando possível, faz a mudança solicitada. Para isso, deve-se existir classes diferentes de QoS. No caso de vários fluxos de mídia ou vídeo codificado por camadas, a reserva de recursos deve ser feita para cada fluxo e camada de forma

independente.

Algumas tecnologias são usadas para aumentar a taxa de dados, e de acordo com essa taxa e a prioridade necessária, a estação base aloca recursos de rádio.

Os servidores de transmissão e as estações base mantêm a sincronização do conteúdo, amenizando as interferências que podem ser de dois tipos: Nível de quadro ou nível de símbolo. No primeiro tipo é transmitido o mesmo conteúdo em um período de quadro de transmissão de rádio e no segundo é transmitido o mesmo conteúdo usando o mesmo recurso de rádio em um símbolo de transmissão de rádio.

Na arquitetura de IPTV móvel, para cada classe de serviços e para cada mídia dentro dessas classes devem ser atribuídos níveis de QoS diferentes e, no caso de fluxos separados, definir a QoS de cada fluxo de maneira diferente. Deve-se controlar erros, alocar e gerenciar recursos de rádio e liberar esses recursos quando a sessão terminar. Alocar recursos na rede de acesso para decidir sua disponibilidade entre o dispositivo terminal e a estação base. Negociar o acesso às redes visitantes durante handoff horizontal e vertical. Gerenciar prioridades de agendamento de tráfego associados ao serviço. Reconfigurar a região de serviço conforme a qualidade do canal de rádio. Deve-se permitir que o serviço de transporte seja gerenciado em vários domínios de provedores de rede e possibilitar o ajuste de parâmetros de QoS e QoE devido a alterações em um canal. Para superar jitter na rede, o dispositivo deve ter espaço suficiente no buffer. Roteador localizado em uma posição que exige a medição de qualidade deve ser capaz de medir parâmetros de transmissão. Se a avaliação de QoS é feita no dispositivo, o cálculo deve ser menor que a capacidade de cálculo do dispositivo.

3.2.4 Requisitos funcionais de dispositivos terminais

Os dispositivos terminais precisam receber e processar dados, gerenciar e minimizar o consumo de energia, selecionar uma rede móvel e recuperar as preferências do usuário para seleção dessa rede. É necessário que eles meçam a qualidade dos recursos de rádio e o desempenho da rede e informem essas medidas, que serão usadas pra reconfigurar a região de serviço, ao provedor de serviços ou provedor de rede. Além disso, devem informar aos provedores de serviços sobre mudanças na capacidade, nos recursos de rede

e na localização dos dispositivos. Pesquisar e selecionar serviços de IPTV e identificar a região do serviço a que está associado. Receber o serviço, independentemente do modo de gerenciamento de energia no modo de serviço broadcast e multicast. Suporte a entrega de conteúdo via múltiplas interfaces de rede e fazer a sincronização e fusão do conteúdo dessas interfaces.

3.2.5 Requisitos funcionais de codificação de mídia

Na arquitetura de IPTV móvel é recomendado o uso de diferentes parâmetros de QoS para cada serviço e codecs multimídia para se adaptar a esses parâmetros e para suportar múltiplas taxas e resoluções de vídeo. Pode ser usada Codificação de Vídeo Escalável (SVC).

3.2.6 Requisitos funcionais de mobilidade

Na arquitetura de IPTV móvel são necessárias estações base na mesma região de broadcasting para compartilhar os mesmos recursos de radio broadcasting. As informações de perfil podem ser obtidas entre as diferentes tecnologias de acesso. Em caso de roaming entre os provedores de rede, deve haver políticas de cobrança entre os mesmos para a entrega de conteúdo de radiodifusão para serem determinados com antecedência.

3.2.7 Requisitos funcionais de segurança

Devem ser fornecidos mecanismos que protejam contra interceptação, geração, alteração e/ou exclusão de dados, bem como impeçam a saída ilegal de conteúdo, e caso ocorra, seja possível rastrear o dispositivo terminal responsável. Devem ser fornecidos mecanismos de controle de acesso, assegurando a privacidade dos dados, a distribuição das chaves e a negociação das políticas de segurança. O usuário deve poder baixar softwares com segurança. Deve ser dado suporte à autenticação de usuários e de dispositivos, sendo, geralmente, essa autenticação uma combinação de um token com uma senha ou uma autenticação biométrica. É necessário garantir a segurança entre o provedor de serviços e dispositivos terminais e o controle de acesso aos recursos.

4 Reportando o desempenho de sessões RTP

O RTCP (SCHULZRINNE et al., 2003) é o protocolo de controle do RTP. Fornece feedback sobre a qualidade das transmissões dos dados, carrega o identificador CNAME (identificador único que permite identificar a fonte mesmo que seu identificador SSRC mude, mais detalhes na sessão 4.1.3), controla a taxa de envio dos pacotes e, opcionalmente, fornece informações de controle, como nome de usuário. Usa o mesmo mecanismo de distribuição usado no RTP, normalmente UDP, ficando a cargo do protocolo subjacente multiplexar os pacotes de dados e de controle.

Existem alguns tipos de pacote RTCP como SR (Relatório do Remetente), RR (Relatório do Receptor), SDES (Descrição da Fonte), BYE (Termino de Fonte) e APP (Específico de Aplicação), cada qual com sua função, conforme descrito na próxima Seção. Se o tipo for desconhecido o mesmo deve ser ignorado a menos que esse novo tipo seja registrado no Internet Assigned Numbers Authority (IANA).

Vários pacotes RTCP podem ser fundidos em um único pacote RTCP composto, devendo esse, ter o seguinte formato: Primeiramente, somente no caso do pacote ser criptografado, são necessários 32 bits aleatórios e se for necessário preenchimento, este deve ser adicionado no último pacote individual do pacote composto. Em seguida, obrigatoriamente, deve haver um pacote SR ou RR. Depois, pacotes RR adicionais, caso haja mais de 31 fontes. Logo após, um pacote SDES obrigatório contendo o CNAME ou outros tipos de pacotes podem vir em qualquer ordem, com exceção do pacote BYE que, se houver, deve ser o último do pacote. Cada participante pode enviar apenas um pacote RTCP composto por intervalo de relatório.

4.1 Tipos de pacote RTCP

Como dito anteriormente, existem alguns tipos pré definidos de pacotes RTCP. Cada um desses pacotes possui uma função, um formato e vários campos. Embora a maioria desses campos sejam diferentes em cada pacote, os campos versão, preenchimento,

comprimento e tipo de pacote existem em todos os pacotes, tendo o mesmo significado em todos. Devido a isso, eles só serão vistos na próxima Subseção, ficando subentendido que estão presentes nas demais.

Nas próximas cinco Subseções, será visto cada um dos cinco pacotes com mais detalhes.

4.1.1 Pacote de relatório do remetente (SR)

O pacote SR envia estatísticas de transmissão e recepção de participantes que são remetentes ativos.

Consiste de quatro seções. Três presentes em todos os pacotes SR e uma opcional, sendo essa uma extensão específica de perfil. A primeira seção, o cabeçalho, apresenta os seguintes campos: versão, preenchimento, contagem do relatório de recepção, tipo de pacote, comprimento e SSRC.

A versão identifica a versão do RTCP, que é a mesma do RTP. O preenchimento indica que o pacote contém octetos adicionais que não fazem parte da informação, onde o último destes define quantos devem ser ignorados, inclusive ele próprio. A contagem do relatório de recepção guarda o número de blocos de relatório de outros transmissores no pacote. O tipo de pacote contém um número inteiro que informa qual o tipo do pacote, no caso do SR esse número é 200. O comprimento informa o comprimento do pacote em palavras de 32 bits menos uma. O SSRC é o identificador da fonte de sincronização dos timestamps RTP.

A segunda seção contém as informações de transmissão dos dados do remetente. Possui os campos NTP timestamp, RTP timestamp, número de pacotes do remetente e contagem de octetos do remetente.

O NTP timestamp captura o tempo (horário local da máquina fonte) em que o relatório foi enviado e pode ser utilizado para calcular o tempo de propagação de ida e volta utilizando o NTP timestamp retornado pelo relatório de recepção. O RTP timestamp corresponde ao mesmo tempo que o NTP timestamp, entretanto utiliza as mesmas unidades e offsets que os pacotes de dados RTP. A correspondência entre os dois timestamps permite fazer sincronização intermédias. O número de pacotes do remetente é o

total de pacotes de dados transmitidos. A contagem de octetos do remetente é o total de octetos (bytes) de carga útil transmitidos, sendo utilizado para estimar a taxa média de dados de carga útil.

A terceira seção contém os blocos de relatório de recepção, um para cada fonte ouvida pelo remetente. Transmite estatísticas sobre a recepção de pacotes RTP a partir de uma fonte de sincronização. Possui os campos SSRC_n, fração perdida, número acumulado de pacotes perdidos, maior número de sequência estendida recebido, jitter entre chegadas, último timestamp SR, atraso desde o último SR.

O SSRC_n é o identificador SSRC da fonte da qual as informações contidas nesse relatório pertencem. A fração perdida é a fração de pacotes de dados RTP perdidos desde o último relatório e é calculada pelo número de pacotes perdidos dividido pelo número de pacotes esperados, que por sua vez é calculado subtraindo o número de sequência inicial recebido do maior número de sequência estendida recebido. O número acumulado de pacotes perdidos é o total de pacotes perdidos e é calculado como sendo o número de pacotes esperados menos o número de pacotes recebidos. Maior número de sequência estendida recebido contém o maior número de sequência recebido em um pacote. O jitter entre chegadas (J) é a estimativa da variância estatística de tempo entre chegadas dos pacotes RTP, medido em unidades de timestamps e expressa como um inteiro positivo. Pode ser visto como a diferença D correspondente ao espaço de tempo entre a chegada de um par de pacotes, comparado com os tempos estampados neste par de pacotes no momento do envio. Conforme definição na especificação do protocolo (SCHULZRINNE et al., 2003) : seja Si o timestamp RTP do pacote i, e Ri o tempo de chegada em unidades de timestamp RTP do pacote i, então para dois pacotes i e j, D pode ser expresso como:

$$D(i,j) = (R_j - R_i) - (S_j - S_i) = (R_j - S_j) - (R_i - S_i).$$

E usando seu valor pode-se calcular o jitter por:

$$J(i) = J(i-1) + (|D(i-1, i)| - J(i-1))/16.$$

O jitter é calculado continuamente. Sempre que um relatório de recepção é emitido, o valor atual de J é amostrado. Atraso desde o último SR é o atraso, expresso em unidades de 1/65536 segundos, entre o último pacote SR recebido da fonte SSRC_n e o envio deste bloco de relatório de recepção.

A quarta seção, a extensão, é opcional. Caso haja informações adicionais a serem comunicadas com frequência sobre remetentes ou receptores em um perfil específico, elas devem ser inseridas nessa extensão.

4.1.2 Pacote de relatório do receptor (RR)

O pacote RR envia estatísticas de recepção dos participantes que não são remetentes ativos.

Esse pacote é semelhante ao pacote SR, alterando apenas o número do tipo de pacote, nesse caso 201, e excluindo as informações do remetente. Um pacote RR vazio deve ser colocado em um pacote RTCP composto quando não há nenhuma transmissão de dados ou recepção de relatórios.

4.1.3 Pacote de descrição da fonte (SDS)

Os pacotes SDS enviam itens que descrevem a fonte. Recomenda-se que no máximo 20% da largura de banda RTCP atribuída a um único participante seja utilizada por pacotes SDS e essa porcentagem deve ser atribuída estaticamente.

É composto por um cabeçalho e zero ou mais blocos, um para cada fonte. O cabeçalho do SDS é semelhante ao do SR e RR, porém o campo contagem de fontes, que identifica o número de fontes no pacote, substitui o campo contagem de relatório de recepção. O número que representa o tipo de pacote SDS é 202. Cada bloco possui um SSRC/CSRC identificando a respectiva fonte, seguido por uma lista de itens que descrevem essa fonte. Itens são contíguos e cada um possui um campo tipo, um contador de octetos que descreve o comprimento do texto e o texto em si.

CNAME, NAME, EMAIL, PHONE, LOC e TOLL são alguns desses itens, sendo que somente o CNAME é obrigatório. Ele fornece uma ligação entre identificadores SSRC e identificadores para a fonte e precisa ser gerado algoritmicamente, devendo ser único entre todos os participantes, fixo para cada participante, adequado para que se localize a fonte e ter o formato "user@host" ou "host".

4.1.4 Pacote BYE

O pacote BYE indica que uma ou mais fontes vão deixar a sessão. Como no SDES, apresenta o campo contagem da fonte (SC) que indica o número de fontes no pacote. O número que representa o tipo de pacote BYE é 203.

Um multiplexador, ao receber um pacote BYE, o encaminha com o SSRC/CSRC inalterado e se for “desligado”, deve enviar um pacote BYE listando os SSRC das fontes que tratava, inclusive o seu próprio. O pacote BYE pode incluir um contador de octetos, seguido por um texto indicando a razão da saída.

4.1.5 Pacote definido pela aplicação (APP)

Pacote que define funções específicas da aplicação, sem que haja a necessidade de se registrar o valor do tipo de pacote. Pacotes APP com nomes não reconhecidos devem ser ignorados. Possui os campos: Subtipo, nome e dados dependentes do aplicativo. O número que representa o tipo de pacote APP é 204.

O nome define o conjunto de pacotes APP para ser único e o subtipo permite que esse conjunto seja definido sob um único nome. Dados dependentes do aplicativo podem ou não aparecer em um pacote APP. O pacote é interpretado pelo aplicativo e não pelo RTP em si.

4.2 Intervalo de transmissão RTCP

Em cada sessão, o envio de dados é limitado pela largura de banda da sessão, que é dividida entre os participantes, devendo estes usar o mesmo valor. Em muitos casos, o tráfego de dados é constante, independentemente do número de participantes. Porém o controle do tráfego cresce ou diminui linearmente com o número de participantes. Assim, faz-se necessário um meio para impedir que o controle do tráfego ultrapasse a largura de banda disponível. Para isso deve-se calcular o intervalo de transmissão dos pacotes de controle RTCP. Deve-se dedicar uma pequena e conhecida fração da largura de banda da sessão para o controle do tráfego. Recomenda-se 5% adicionais da largura de banda da sessão (SCHULZRINNE et al., 2003).

Como o controle do tráfego depende do número de participantes, deve-se criar uma tabela de membros, adicionando o SSRC/CSRC do participante na tabela quando o mesmo é ouvido e retirando-o quando recebe um pacote BYE ou quando o tempo de transmissão expira por um tempo determinado. Da mesma forma é útil ter-se uma tabela que guarde o SSRC/CSRC dos remetentes, adicionando-o quando um pacote RTP é recebido de um participante e excluindo-o quando ele fica um tempo determinado sem enviar um pacote.

O primeiro pacote composto enviado por um participante deve ser adiado por uma variação aleatória de metade do intervalo mínimo.

Para encontrar o intervalo de transmissão deve-se ter algumas constantes bem definidas. Tp informa a última vez que um pacote RTCP foi transmitido. Tc informa o tempo atual. Tn informa a próxima transmissão programada de um pacote RTCP. $Members$ fornece o número atual de membros da sessão. $Pmembers$ representa o número estimado de membros no momento Tn calculado por último. $Senders$ informa o número de remetentes ativos e we_sent informa se um participante é um remetente ativo ($we_sent = true$) ou não ($we_sent = false$). O tamanho médio de um pacote RTCP composto é guardado em avg_rtcp_size e o valor da largura de banda RTCP em $rtcp_bw$. $Initial$ informa se o participante já enviou ($initial = false$) ou não ($initial = true$) um pacote RTCP. Além disso, ainda há outras variáveis que serão úteis: C e n , Td (intervalo calculado determinista) e $Tmin$. Para encontrar o intervalo de transmissão Tn , deve-se proceder da seguinte forma:

Se we_sent é verdadeiro e $senders$ é menor ou igual a $0,25 * members$, C recebe $avg_rtcp_size / (0,25 * rtcp_bw)$ e n recebe $senders$. Se $senders$ é maior do que $0,25 * members$, remetentes e receptores são tratados juntos, C recebe $avg_rtcp_size / rtcp_bw$ e n recebe $members$. Se we_sent é falso, C recebe $avg_rtcp_size / 0,75 * rtcp_bw$ e n recebe o número de receptores ($members - senders$). Se $initial$ é verdadeiro, $Tmin$ é definido como 2,5 segundos, caso contrário $Tmin$ é definido como 5 segundos.

O máximo entre $Tmin$ e $(C*n)$ representa o intervalo calculado determinístico Td que multiplicado por um número entre 0.5 e 1.5 nos dá o intervalo de transmissão (Tn), que por sua vez é dividido por 1,21828, resultando no valor procurado. Isso geralmente garante

no mínimo 25% da largura de banda para os remetentes e o resto para os receptores. Caso os remetentes constituam mais de 1/4 dos membros, a banda é dividida igualmente entre todos os participantes.

Essas variáveis, além de serem importantes no cálculo do intervalo de transmissão dos pacotes, são importantes em outras situações como quando um participante entra ou sai de uma sessão, por exemplo, e por isso devem ser constantemente atualizadas.

4.3 Atualização das variáveis do protocolo

Ao aderir a sessão, o participante adiciona seu SSRC na tabela de membros, define *tp*, *tc* e *senders* como 0, *pmembers* e *members* como 1, *we_sent* como falso, *rtcp_bw* como a fração da largura de banda da sessão, *initial* como verdadeiro, e *avg_rtcp_size* como o tamanho provável do primeiro pacote a ser enviado. Calcula-se o intervalo de transmissão T e o primeiro pacote é agendado para esse tempo $T_n = T$, ou seja, o temporizador expira no tempo T .

Se um participante envia um pacote RTP e seu *we_sent* = *false*, acrescenta-se o seu SSRC na tabela de remetentes, incrementa-se *senders* e define-se *we_sent* como *true*.

Quando um pacote é recebido, há dois tipos de procedimentos que se pode fazer com base no tipo de pacote. Caso um pacote RTP seja recebido consulta-se a tabela de membros e em seguida a tabela de remetentes, caso o SSRC do participante não esteja presente em alguma das duas, ele é adicionado à respectiva tabela e *members* e/ou *senders* é atualizado. Caso um pacote RTCP é recebido verifica-se o seu tipo: Se for um pacote BYE, o SSRC do pacote é procurado na tabela de membros e em seguida na tabela de remetentes, caso esteja presente em alguma das duas ele é removido e *members* e/ou *senders* são atualizados. Caso seja um pacote RTCP de outro tipo, consulta-se a tabela de membros, caso seu SSRC não esteja presente, ele é adicionado e *members* é atualizado. Para cada pacote RTCP composto recebido deve-se atualizar *avg_rtcp_size* com a fórmula: $avg_rtcp_size = packet_size/16 + (15*avg_rtcp_size)/16$; onde *packet_size* é o tamanho do pacote RTCP recebido.

Além de enviar um pacote BYE, há outra maneira de um participante sair de uma sessão que é expirando o tempo de transmissão. De tempos em tempos os parti-

participantes devem verificar se algum outro participante atingiu o limite de tempo máximo para transmissão. Para isso, deve-se calcular o intervalo calculado determinístico Td sem o fator de randomização para um receptor ($we_sent = false$). Se algum outro membro não enviou um pacote RTP ou RTCP desde o tempo $tc - MTd$ (M é multiplicador de tempo limite, sendo 5 o valor recomendado) ele é excluído. No caso de ser um remetente, qualquer outro remetente que não tenha enviado um pacote RTP desde o tempo $tc - 2T$ é removido da tabela de remetentes e seu we_sent é definido como *false*.

Se o temporizador expirar o participante deve calcular o intervalo de transmissão T . Se $tp + T$ é menor ou igual a tc , um pacote RTCP é transmitido, tp é definido para tc e o temporizador é definido para expirar no tempo tn . Se $tp + T$ é maior que tc , tn recebe $tp + T$, um pacote é enviado e o temporizador é definido para expirar no tempo tn . $Pmembers$ é definido como $members$, se um pacote RTCP é transmitido *initial* é definido como *false* e o valor de *avg_rtcp_size* é atualizado pela fórmula $avg_rtcp_size = packet_size/16 + (15*avg_rtcp_size)/16$.

Quando um pacote BYE é enviado, se $members$ é maior que 50, tp é definido para tc (o tempo atual), $members$ e $pmembers$ são definidos como 1, *initial* é definido como *true*, we_sent é definido como *false*, $senders$ é definido como 0, e *avg_rtcp_size* é definido como o tamanho do pacote BYE composto. O intervalo de transmissão T é computado. O pacote BYE é programado para o tempo $tn = tc + T$.

Se a estimativa de $members$ é menor que 50, o participante pode enviar um pacote BYE imediatamente. Um participante que nunca enviou um pacote RTP ou RTCP não deve enviar um pacote BYE quando deixar o grupo.

4.4 Análise dos relatórios

A análise dos relatórios permite que os membros envolvidos avaliem o desempenho das transmissões, identificando problemas e fazendo modificações para solucioná-los.

Através de contagens cumulativas pode-se calcular diferenças entre dois relatórios e, assim, fazer medições a curto e longo prazo. A diferença nos dois relatórios mais recentes ajuda a estimar a qualidade recente da distribuição.

Pode-se calcular a taxa de perda de pacotes entre dois relatórios. Essa taxa deve

ser igual à fração perdida caso esses dois relatórios sejam consecutivos. Pode-se também achar a taxa média perdida por segundo, dividindo a fração perdida pelo timestamp (em segundos). Subtraindo-se o número de pacotes perdidos do número de pacotes esperados obtém-se o número de pacotes efetivamente recebidos.

Pode-se também calcular a taxa média de dados de carga útil e a taxa média de pacotes sobre um intervalo. A relação dos dois dá o tamanho médio de carga útil. O número de pacotes recebidos multiplicado pelo tamanho médio de carga útil dá o rendimento para o receptor.

Uma outra medida a curto prazo é o jitter entre chegadas, que pode indicar congestionamento antes que este leve à perda de pacotes. Destina-se à comparação entre vários relatórios de um mesmo receptor ou entre vários receptores ao mesmo tempo. Assim, é importante usar a mesma fórmula para calcular o jitter para todos os receptores.

5 Proposta de streaming adaptativo para IPTV Móvel

Como vem sendo mencionado nos capítulos anteriores, a presente monografia tem foco no serviço de IPTV para os ambientes móveis, uma vez que esses ambientes possuem várias limitações e se apresentam suscetíveis a variações de conectividade, muito comuns nesses ambientes, além do handover, podendo esse conjunto de fatores gerar muitas perdas de pacotes, atrasos, entre outros problemas que podem afetar a qualidade da visualização do conteúdo multimídia. Entretanto, também pode ser usado em ambientes fixos onde ocorra os mesmos problemas, embora nesses ambientes isso seja menos comum.

A figura abaixo oferece uma amostra do cenário reproduzido para a implementação da presente monografia.

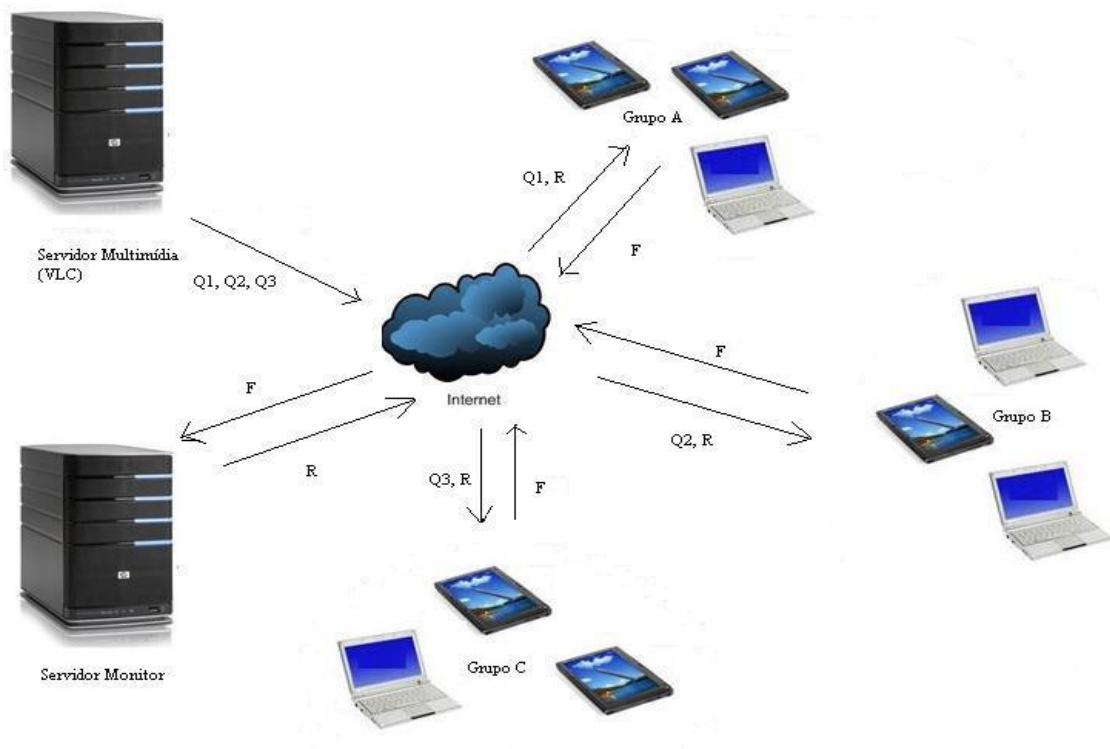


Figura 5.1: Cenário proposto na implementação

Na figura é possível observar os elementos básicos que compõem a arquitetura. O servidor multimídia utilizado é o VLC que, de acordo com o site Videolan.org, é um

player multimídia rápido e simples capaz de converter mídias e criar múltiplos fluxos simultâneos. Além disso, pode ser executado em diversos sistemas operacionais e é capaz de reproduzir a maioria dos arquivos de vídeo existentes na atualidade.

No cenário mostrado, esse servidor envia vídeos para grupos multicast, um vídeo com uma qualidade diferente para cada grupo. Em outros cenários imagináveis, esses outros grupos multicast poderiam receber diferentes camadas do vídeo, caso tenha sido feita codificação escalável, entretanto isso não se aplica nessa implementação. Na figura estão ilustrados três grupos multicast, compostos de dispositivos terminais móveis (como no IPTV móvel), recebendo do servidor de mídia o mesmo vídeo, com uma qualidade diferente (Q1, Q2, Q3) para cada um dos grupos. Cada dispositivo terminal no grupo, ao receber os pacotes RTP, emitem pacotes RTCP (pacote de relatório do receptor - RR) contendo informações sobre a recepção dos dados multimídia a um Componente de Adaptação, implementado como um servidor à parte (servidor monitor). O Componente de Adaptação analisa esses pacotes e dá uma resposta ao cliente, informando se a transmissão está com uma qualidade aceitável e satisfatória ou se o cliente precisará se conectar a outro grupo multicast com conteúdo em uma qualidade diferente. Caso seja constatada a necessidade de mudança, o cliente a faz, passando a receber o conteúdo na qualidade indicada na resposta do Componente de Adaptação, se conectando ao respectivo grupo multicast que recebe o vídeo com essa qualidade.

O Zapper é o player inserido no dispositivo terminal do cliente que proporciona que o mesmo visualize os conteúdos enviados pelo servidor multimídia. Ele é um protótipo já em execução de uma aplicação para o consumo do serviço de TV linear e é capaz de exibir áudio e vídeo nos formatos recomendados, além de permitir a troca de canais por numeração, incremento e decremento (GT-IPêTeVê, 2014). Sua especificação segue a recomendação (ITU-T H.721, 2009). Consiste em uma aplicação cliente que usa basicamente FFmpeg e SDL na sua implementação.

Segundo o site ffmpeg.org, o FFmpeg é um framework capaz de codificar, decodificar, transcodificar, multiplexar, demultiplexar, filtrar e reproduzir diversos formatos de arquivos multimídia. A segurança é prioridade e as revisões de código são sempre feitas com isso em mente. FFmpeg lê arquivos de entrada e escreve em arquivos de saída,

podendo cada arquivo conter um número limitado de fluxos de diferentes tipos. A transcodificação no FFmpeg acontece da seguinte forma: Um demultiplexador lê arquivos de entrada obtendo pacotes contendo dados codificados, mantendo os mesmos sincronizados. Esses pacotes são passados para o decodificador que produz quadros sem compressão, que são transmitidos para o codificador, que codifica os pacotes que, finalmente são passados para o multiplexador, que grava os pacotes codificados no arquivo de saída.

Segundo o site libsdl.org, o SDL é uma biblioteca de desenvolvimento multiplataforma, usada por softwares de reprodução de vídeo, emuladores e jogos, projetada para fornecer acesso de baixo nível ao áudio, teclado, mouse, joystick e aos gráficos de hardware via OpenGL e Direct3D. É escrito em C e por ser distribuído sob a licença zlib, pode ser usado livremente. SDL oferece gráficos 3D, criação e gerenciamento de múltiplas janelas para vídeos. Possui um conjunto de eventos de entrada, cada um podendo ser ativado ou desativado, e esses são passados através de uma função de filtro especificada pelo usuário antes de serem enviados para a fila de eventos internos. Define vários formatos de reprodução de áudio, além de executá-lo de forma independente em uma thread separada. Suporta criação de threads simples, threads de armazenamento local, mutexes, semáforos e variáveis de condição. Pode obter o número de milissegundos decorridos, esperar um número especificado de milissegundos e criar temporizadores que funcionam em uma thread separada. Pode também consultar o número de CPUs e detectar os recursos e conjuntos de instruções suportados por eles, além de poder consultar o estado de gerenciamento de energia.

Assim, no cenário de implementação, o VLC envia um mesmo vídeo em três qualidades diferentes via três canais multicast. Um dos vídeos com qualidade alta (Q3), um com qualidade média (Q2) e um com qualidade baixa (Q1). Para fazer a adaptação, o cliente alterna entre esses três vídeos, mudando a qualidade do vídeo exibido de acordo com a qualidade da transmissão.

A estratégia adotada foi a de implementar o Componente de Adaptação como um servidor separado, também conhecido como servidor de terceiros (servidor monitor). Esse servidor é responsável por receber os pacotes de relatório dos receptores e, através das informações contidas nesses relatórios, tomar decisões e enviá-las aos clientes para que os

mesmos façam a adaptação.

Optou-se por implementar a tomada de decisões nesse servidor à parte pensando no desempenho, uma vez que desse modo, toda a responsabilidade de se tomar as decisões sobre mudar, ou não, a qualidade fica a cargo do servidor monitor, tirando isso tanto do cliente quanto do servidor multimídia. O fluxo principal só é interrompido no cliente quando há a necessidade de modificação. Tal desacoplamento permite, também, que futuras implementações posicionem o Componente de Adaptação junto a qualquer um dos lados da comunicação ou outros pontos intermediários.

Embora, no cenário de implementação, o servidor monitor tome a decisão de modificação, isso poderia ser feito tanto no cliente quanto no servidor multimídia.

5.1 Implementação

A implementação foi desenvolvida no ambiente de desenvolvimento Code Blocks em linguagem C e C++, sendo que sua maior parte seguiu os conceitos vistos em (SCHULZ-RINNE et al., 2003). Foi usado também comunicação por sockets, tabelas hash, entre outros conceitos importantes. O socket utilizado realiza a entrega sobre TCP, pois, assim a entrega é garantida, uma vez que a perda de pacotes RTCP pode influenciar negativamente na qualidade da transmissão, uma vez que sem esses pacotes torna-se impossível a adaptação da transmissão às características da rede.

5.1.1 Implementação do Servidor

Já foi mencionado anteriormente que o Componente de Adaptação encontra-se no servidor responsável por monitorar a qualidade das transmissões, o servidor monitor. Pelo fato desse servidor poder ter que lidar com um grande número de clientes, dependendo do cenário da aplicação, é necessário que se tenha um controle para que se saiba o número de membros na sessão e para que os mesmos possam ser identificados com facilidade. O principal identificador de um participante, o seu SSRC, não é o suficiente para identificá-lo, uma vez que, se detectado algum conflito ou o participante for reiniciado, esse SSRC pode ser modificado durante a sessão. Sendo assim, além de armazenar o SSRC do cliente, faz-

se necessário também armazenar o seu CNAME. Pensando nisso, criou-se uma estrutura para guardar o SSRC e o CNAME de cada participante, que por sua vez são armazenados em uma tabela Hash. Foi criada uma variável para guardar o número de membros da sessão, sendo incrementada quando o servidor recebe um pacote RTCP (com exceção dos pacotes BYE) e decrementada quando o servidor recebe um pacote BYE ou quando a conexão expira devido a um tempo sem receber nenhum pacote.

Uma série de métodos são responsáveis por manipular os dados nessa tabela Hash. O cálculo da posição exata onde armazenar os dados é feito dividindo-se o SSRC do cliente a ser armazenado pelo tamanho da tabela e pegando-se o resto dessa divisão, que será a posição onde serão inseridos os dados. É desejável, para fins de desempenho, que o tamanho da tabela seja um primo e nessa implementação foi usado o 7, já que só existirá uma fonte e um cliente receptor, sendo desnecessário definir um tamanho maior. Caso essa implementação seja usada em um cenário com muitos membros (remetentes e/ou receptores) na sessão, esse valor deve ser aumentado.

É possível também inicializar todas as posições da tabela com NULL, inserir, excluir, buscar e imprimir os dados. Antes de qualquer inserção ou exclusão na tabela é feita uma busca para se ter certeza de que os dados não estejam presentes na mesma (no caso de inserção) ou que existam na mesma (no caso de exclusão).

O servidor é executado e fica esperando indefinidamente até que algum cliente se conecte a ele. Quando isso acontece, uma conexão é estabelecida e ambos (servidor monitor e cliente) estão prontos para iniciar uma comunicação através de sockets.

Tal comunicação se inicia efetivamente, com o cliente enviando um pacote RTCP ao servidor que, quando recebido, tem algumas informações de cabeçalho analisadas para verificar se o mesmo é um pacote válido. Essa verificação é feita para cada pacote recebido do cliente e se baseia em três campos que são: versão do protocolo (version), tipo de pacote (type) e comprimento do pacote (length). Para a versão do protocolo aceita-se somente a versão 2, que é a versão mais atual e utilizada na implementação. O tipo de pacote aceitável varia de 201 a 203 e o comprimento varia de acordo com o tipo de pacote.

No cenário proposto, não foram implementados os pacotes de relatório do remetente (SR) uma vez que, como os pacotes RTCP são enviados para um servidor à parte,

julgou-se que a implementação do mesmo não seria necessária, sendo possível conseguir ótimos resultados utilizando apenas os pacotes de relatório do receptor (RR), como visto nessa implementação. Os pacotes APP também não foram implementados.

Os pacotes RTCP são implementados como estruturas onde cada campo do pacote é representado por um campo na estrutura. Foi criada uma estrutura para os pacotes RR, uma para os pacotes SDES, uma para os pacotes BYE e uma estrutura composta, constituída de um pacote RR e um pacote SDES.

Quando o servidor recebe um pacote RTCP do cliente, e este é um pacote válido e não é um pacote BYE, o SSRC e o CNAME do cliente são inseridos na tabela de membros e é feita uma análise da transmissão dos pacotes RTP do servidor multimídia ao Zapper, levando-se em consideração a qualidade de cada vídeo em particular. A idéia é simples: o servidor vai analisar as informações contidas no pacote RTCP e, a partir dessa análise, enviar um valor tipo char ao cliente através de um socket. Cada valor tipo char enviado tem um significado definido e conhecido pelo cliente e a partir dele o cliente saberá como proceder, mudando ou não a qualidade do vídeo que está sendo transmitido. O vídeo sendo executado é representado por um inteiro, onde 1 é a qualidade mais baixa, 2 é a qualidade intermediária e 3 é a melhor qualidade. Definiu-se que a transmissão sempre comece com a qualidade mais baixa (vídeo 1), assim todos os clientes serão beneficiados num primeiro momento, podendo visualizar o conteúdo. É importante ressaltar também que, se o servidor receber um pacote RTCP fora dos padrões aceitáveis, indicando que a transmissão está ruim, a qualidade do vídeo poderá, ou não, ser alterada de imediato. Foi criada uma variável (count) que permite controlar quantos pacotes indicando qualidade ruim na transmissão serão precisos para que a qualidade na transmissão se modifique, isso permite adequar a implementação a diversos tipos de cenário. O valor analisado que leva à alteração da qualidade da transmissão é o número de pacotes perdidos (lost) por intervalo de transmissão, e os valores que levam à alteração são estabelecidos no próximo capítulo.

Quando o servidor recebe um pacote RTCP BYE do cliente, o SSRC e o CNAME do cliente são retirados da tabela de membros, a variável members é decrementada e a conexão com o cliente é interrompida. Comportamento semelhante se observa quando

o servidor monitor fica um determinado intervalo de tempo sem receber nenhum pacote RTCP do cliente, após a conexão e a comunicação entre ambos tiverem sido estabelecidas. Nesse caso, o servidor age exatamente como se tivesse recebido um pacote BYE.

5.1.2 Implementação do Cliente

Como mencionado anteriormente, o Zapper funcionará como o dispositivo terminal do cliente IPTV móvel.

Para que se alcançasse o objetivo proposto, foi necessário fazer adições e modificações na implementação do Zapper. Na sua implementação original, era necessário o pressionamento de teclas especiais para que o conteúdo começasse a ser exibido, entretanto com as modificações feitas, o conteúdo é exibido automaticamente quando se inicia o Zapper, começando sempre a exibir o conteúdo com a qualidade mais baixa primeiro e, dependendo das condições da rede no momento, essa qualidade do conteúdo pode ir aumentando.

Foi necessário, também, fazer alterações no código fonte do FFmpeg e recompilá-lo. Assim, foi inserida uma função que permite a utilização de informações contidas nos pacotes RTP e outras informações de cada fonte de dados, que de outra forma não poderiam ser utilizadas, já que ficam encapsuladas no FFmpeg. Tais informações são, por exemplo, o SSRC da fonte, o tempo em que os dados são amostrados nos pacotes (timestamp), o número de seqüência dos pacotes, entre outras informações. Com isso, foi possível calcular o número de pacotes esperados, recebidos e perdidos, o atraso entre a chegada dos pacotes, o jitter, entre outros cálculos importantes. A cada pacote RTP que chega ao cliente, essas informações são atualizadas e, em intervalos de tempo calculados dinamicamente (intervalo de transmissão) os dados necessários são enviados em pacotes RTCP ao servidor monitor e zerados para que sejam novamente calculados.

Como no servidor, é importante que o cliente mantenha o controle do número de participantes e, nesse caso, também do número de remetentes de dados da sessão. Assim, foram criadas duas variáveis para guardar essas informações e duas tabelas hash para guardar o SSRC dos remetentes e de todos os membros da sessão. Essas informações são importantes para que se tenha certo controle sobre a sessão e seus participantes e

para que se possa calcular o intervalo de transmissão dos pacotes RTCP, uma vez que esse intervalo aumenta, ou diminui, com o número de participantes, controlando assim o congestionamento, e evitando uma inundação de pacotes RTCP na rede. Assim, no cliente também foram implementadas funções que manipulam as tabelas hash, permitindo calcular onde armazenar a informação nas tabelas, inicializar as tabelas, inserir, buscar e excluir o SSRC das tabelas e imprimi-las na tela.

Assim como o número de remetentes e de membros na sessão são importantes para o cálculo do intervalo de transmissão, outras informações são importantes para esse cálculo, tais como: se o participante é, ou não, um remetente, se ele já enviou, ou não, um pacote RTCP (ambos definidos por uma variável booleana), o tamanho médio dos pacotes RTCP, a largura de banda destinada ao envio desses pacotes, entre outras informações. As variáveis que guardam essas informações são inicializadas assim que o participante é executado. Além disso, é calculado o primeiro intervalo de transmissão e dois números inteiros aleatórios e únicos entre todos os participantes são gerados, nesse caso entre 0 e 99, os quais serão o SSRC e o CNAME do cliente, que serão inseridos na tabela de membros e/ou de remetentes (conforme o caso), incrementando a respectiva variável que guarda seu total.

O intervalo de transmissão é calculado a cada pacote gerado, com base nas variáveis acima mencionadas, assim se faz importante estar sempre atualizando as mesmas. O mecanismo utilizado para o cálculo desse intervalo se baseia num conjunto de regras definidas em (SCHULZRINNE et al., 2003).

Ao ser iniciado, o cliente faz a conexão com o servidor monitor, que estava executando, esperando algum cliente se conectar a ele. Logo após, ambos estão prontos para trocar mensagens, o cliente enviando pacotes RTCP ao servidor e recebendo dele a resposta informando se é, ou não, necessário mudar a qualidade do conteúdo transmitido. Essa conexão e troca de mensagens é feita através de sockets, como já mencionado.

Após o envio de cada pacote RTCP ao servidor, o cliente recebe a resposta do mesmo e a analisa. Essa resposta é uma variável do tipo char que pode ser '0', '1', '2', '3' ou '9'. Caso seja '0', nada será feito, pois esse valor indica que a qualidade do conteúdo não precisa ser alterada. Se for '1' o cliente fechará o fluxo de dados atual e se conectará

ao vídeo com a menor qualidade disponível. Caso seja ‘2’, o cliente fecha o fluxo de dados e se conecta ao vídeo com a qualidade intermediária. Caso seja ‘3’, o cliente fecha o fluxo de dados atual e se conecta ao vídeo com a melhor qualidade disponível. Caso seja ‘9’ o cliente está sendo informado que os pacotes por ele enviados não estão sendo validados. Vale ressaltar que, quando a necessidade de mudança na qualidade do conteúdo transmitido é detectada, o cliente a faz mudando o endereço multicast e a porta.

6 Resultados Obtidos

Para que fosse verificada a eficácia do algoritmo e medidos os valores aceitáveis e críticos de perda de pacotes na transmissão dos mesmos, realizou-se modificações no FFmpeg inserindo-se um trecho de código que faz com que um número aleatório de pacotes sejam propositalmente perdidos, não sendo recebidos pelo Zapper, simulando assim uma situação real. Isso foi feito porque os testes foram realizados localmente, e não em rede.

Entende-se por valores aceitáveis de perda de pacotes, os valores que, mesmo se atingidos, não geram muitos problemas na visualização do conteúdo, ou seja, mesmo que esses pacotes sejam perdidos, o cliente ainda consegue visualizar o conteúdo. Já os valores críticos são aqueles que, quando atingidos, faz-se necessário a adaptação mudando a qualidade do conteúdo transmitido, de modo que o mesmo possa continuar a ser visualizado.

A variável que se pretende medir é o número de pacotes RTP perdidos (lost) por intervalo de transmissão. Inicialmente foi observada a perda de pacotes para diversos valores randômicos, destacando-se cinco de maior relevância: P1, P2, P3, P4, P5.

P1 representa a fórmula “ $\text{rand()} \% 100) < 2$ ” o que significa que, a cada pacote a ser enviado é sorteado um número entre 0 e 99 e, caso este número seja 0 ou 1, o pacote não é entregue, simulando assim uma perda. Isso dá uma probabilidade de 2% de chances de se perder um pacote. Seguindo-se a mesma ideia, P2 representa a fórmula “ $\text{rand()} \% 100) < 1$ ”, o que dá uma probabilidade de 1% de se perder um pacote. P3 representa a fórmula “ $\text{rand()} \% 1000) < 5$ ”, o que dá uma probabilidade de perda de 0,5% por pacote. Vale notar que o valor do divisor agora é 1000. P4 representa a fórmula “ $\text{rand()} \% 1000) < 4$ ” e dá uma probabilidade de 0,4% de se perder um pacote. Finalmente, P5 representa a fórmula “ $\text{rand()} \% 1000) < 3$ ” dando uma probabilidade de perda igual a 0,3%.

O algoritmo foi executado para cada probabilidade de perda citada acima, uma vez para cada qualidade transmitida pelo servidor. O intervalo de transmissão de cada pacote RTCP foi calculado dinamicamente a cada pacote enviado. Para cada execução do algoritmo observou-se 40 pacotes consecutivos.

Após realizados esses testes, foram colhidos e analisados os resultados, que são mostrados nos gráficos abaixo:

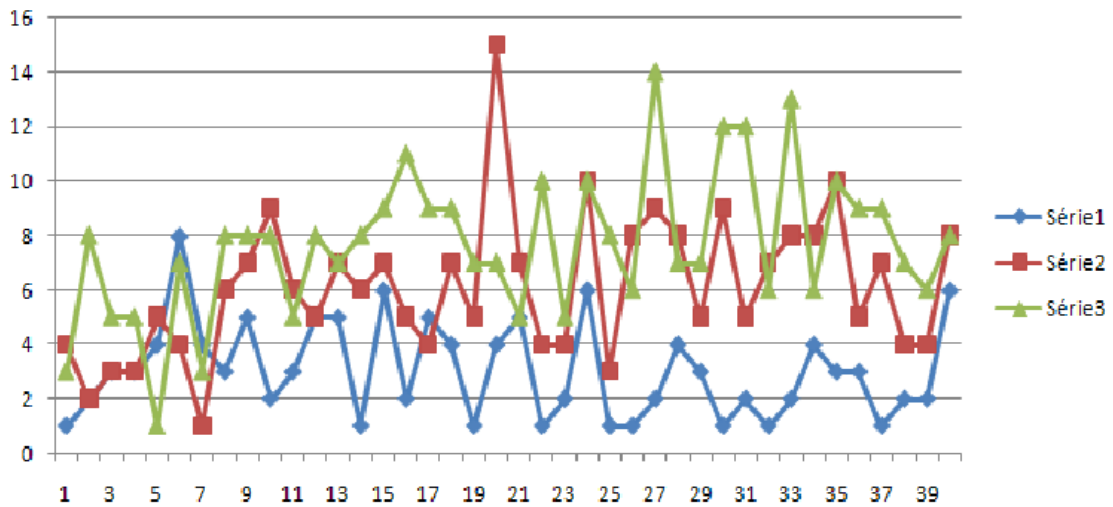


Figura 6.1: Gráfico para Probabilidade P1

A figura 6.1 nos mostra a perda de pacotes por intervalo de transmissão quando se executa o algoritmo com probabilidade P1 (probabilidade de perda de 2% por pacote) para as transmissões com a melhor qualidade (série 3), qualidade média (série 2) e baixa qualidade (série 1). Os valores na vertical mostram o número de pacotes RTP perdidos a cada intervalo de transmissão, representado na horizontal. Assim, observou-se uma distribuição de perda de pacotes, para a melhor qualidade, como se segue: 1 = 1; 3 = 2; 5 = 5; 6 = 4; 7 = 7; 8 = 8; 9 = 5; 10 = 3; 11 = 1; 12 = 2; 13 = 1; 14 = 1. O que indica que, em 1 intervalo de transmissão foi perdido 1 pacote RTP, em 2 intervalos de transmissão foram perdidos 3 pacotes RTP, e assim por diante. Da mesma forma, para a qualidade média, obteve-se a seguinte distribuição dos valores: 1 = 1; 2 = 1; 3 = 3; 4 = 7; 5 = 7; 6 = 3; 7 = 7; 8 = 5; 9 = 3; 10 = 2; 15 = 1. E finalmente, para a pior qualidade, observou-se a seguinte distribuição dos valores de perda: 1 = 9; 2 = 9; 3 = 7; 4 = 6; 5 = 5; 6 = 3; 8 = 1.

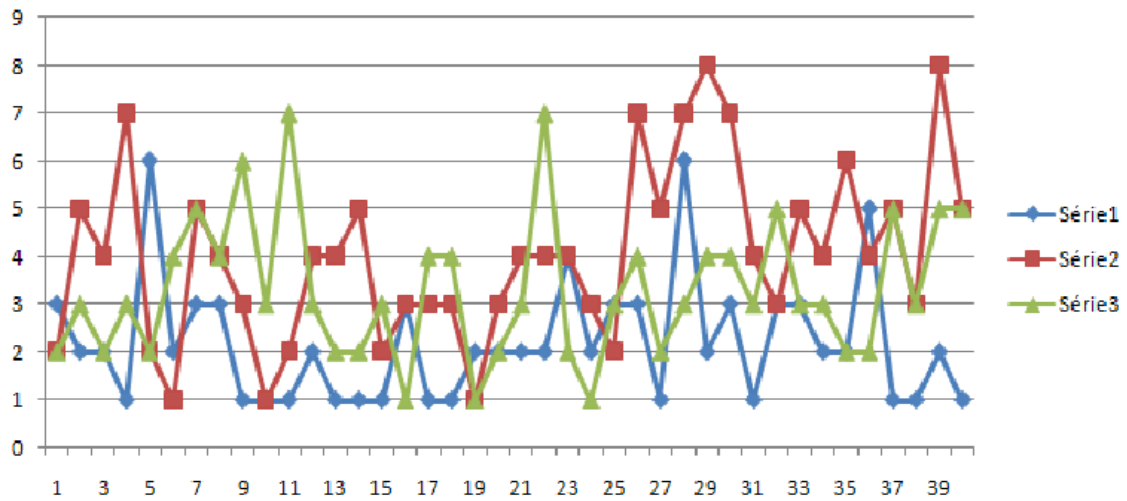


Figura 6.2: Gráfico para Probabilidade P2

Seguindo-se essa linha de raciocínio, a figura 6.2 nos mostra a perda de pacotes por intervalo de transmissão quando se executa o algoritmo com probabilidade P2 (probabilidade de perda de 1% por pacote) para as transmissões com qualidade alta (série 3), média (série 2) e baixa (série 1). A distribuição para a melhor qualidade foi: 1 = 3; 2 = 10; 3 = 12; 4 = 7; 5 = 5; 6 = 1; 7 = 2. A distribuição para a qualidade intermediária foi de: 1 = 3; 2 = 5; 3 = 8; 4 = 10; 5 = 7; 6 = 1; 7 = 4; 8 = 2 e a distribuição para a qualidade baixa foi: 1 = 14; 2 = 13; 3 = 9; 4 = 1; 5 = 1; 6 = 2.

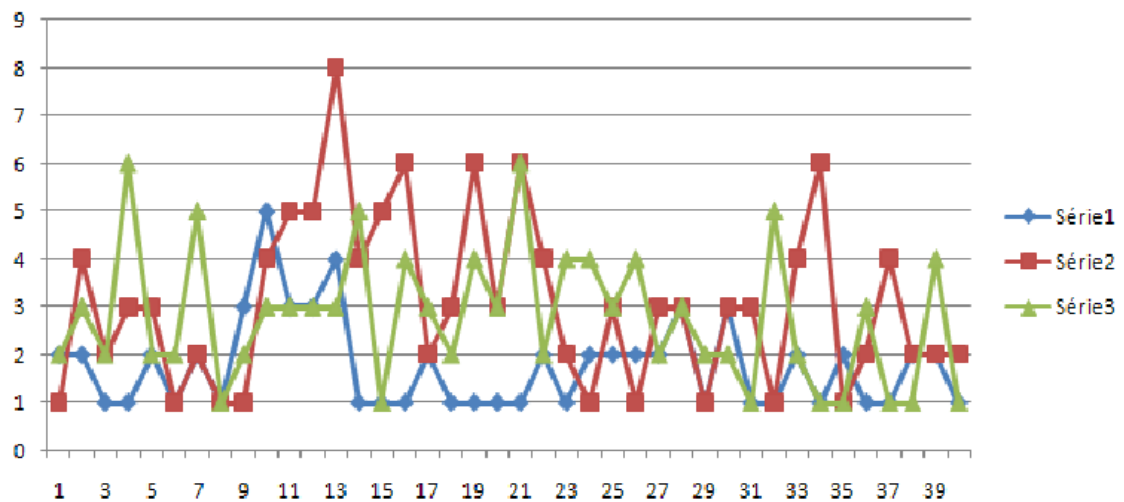


Figura 6.3: Gráfico para Probabilidade P3

A figura 6.3 mostra a perda de pacotes por intervalo de transmissão quando se executa o algoritmo com probabilidade P3 (probabilidade de perda de 0.5% por pacote)

para as transmissões com a melhor qualidade (série 3), qualidade média (série 2) e qualidade baixa (série 1). As distribuições observadas foram: 1 = 8; 2 = 11; 3 = 10; 4 = 6; 5 = 3; 6 = 2 para a melhor qualidade. 1 = 9; 2 = 8; 3 = 9; 4 = 6; 5 = 3; 6 = 4; 8 = 1 para a qualidade média e 1 = 19; 2 = 14; 3 = 5; 4 = 1; 5 = 1 para a qualidade mais baixa.

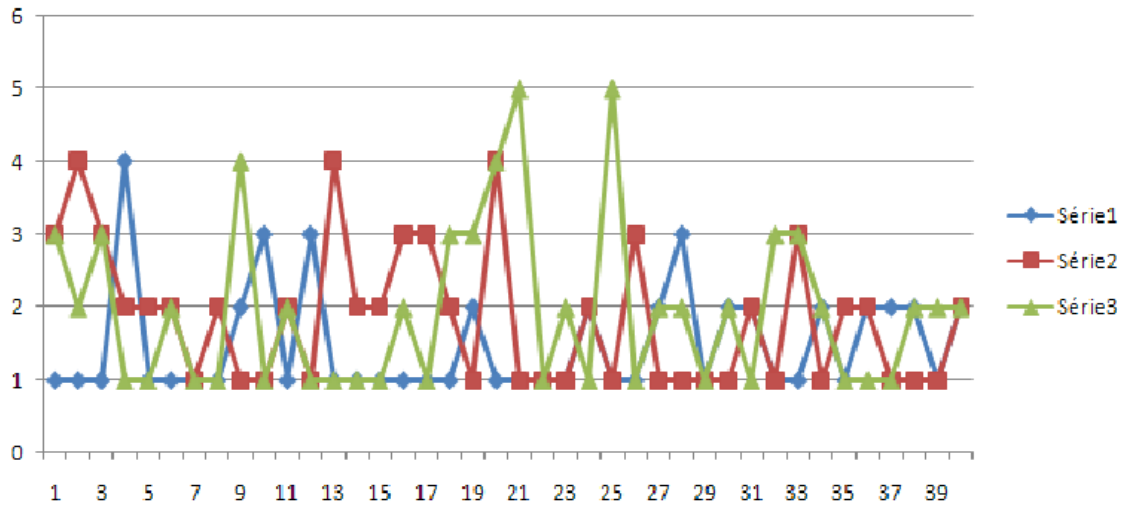


Figura 6.4: Gráfico para Probabilidade P4

A figura 6.4 mostra a perda de pacotes por intervalo de transmissão quando se executa o algoritmo com probabilidade P4 (probabilidade de perda de 0.4% por pacote) para as transmissões com qualidade alta (série 3), média (série 2) e baixa (série 1). A distribuição de perda de pacotes por intervalo de transmissão observada para a melhor qualidade foi: 1 = 18; 2 = 12; 3 = 6; 4 = 2; 5 = 2. Para a qualidade média foi: 1 = 18; 2 = 13; 3 = 6; 4 = 3. E para a qualidade mais baixa foi: 1 = 25; 2 = 11; 3 = 3; 4 = 1.

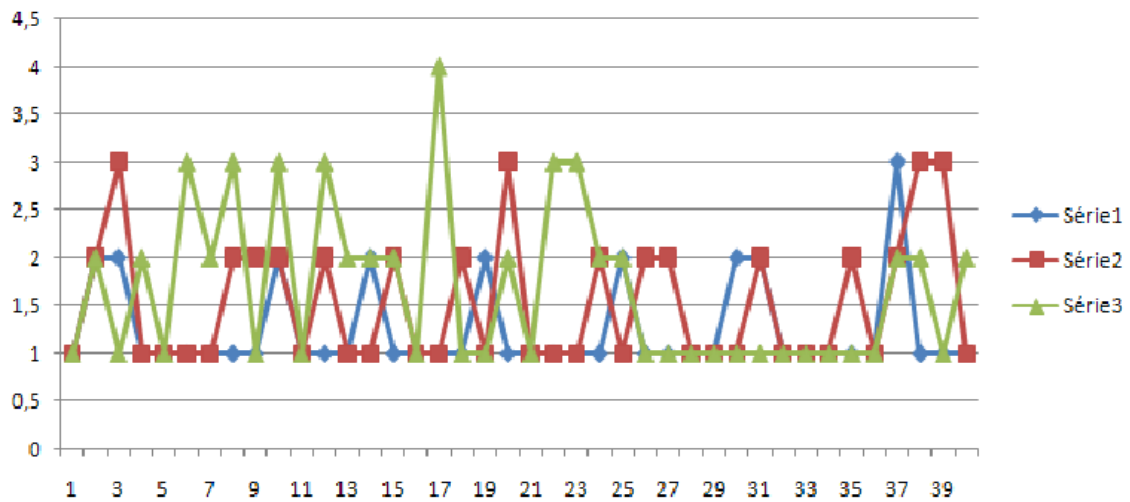


Figura 6.5: Gráfico para Probabilidade P5

A figura 6.5 mostra a perda de pacotes por intervalo de transmissão quando se executa o algoritmo com probabilidade P5 (probabilidade de perda de 0.3% por pacote) para as transmissões com qualidade alta (série 3), média (série 2) e baixa (série 1). A distribuição de perda de pacotes foi de: 1 = 21; 2 = 12; 3 = 6; 4 = 1 para a melhor qualidade. 1 = 23; 2 = 13; 3 = 4 para a qualidade média e 1 = 31; 2 = 8; 3 = 1 para a pior qualidade.

Como dito anteriormente, foi preciso fazer modificações no FFmpeg a cada probabilidade de perdas de pacotes diferente que se inseria e recompilá-lo.

Após executados todos esses testes e observando, nos gráficos, os valores de perdas gerados por intervalo de transmissão e, no Zapper, as transmissões dos vídeos em suas diferentes qualidades, concluiu-se que a melhor probabilidade a se usar é a P4, ou seja, a cada pacote a ser enviado, é escolhido um número inteiro randômico entre 0 e 999 e todos os pacotes que tiverem esse número entre 0 e 3 serão perdidos.

Na probabilidade P1, notou-se que na maioria das vezes um número expressivo de pacotes RTP estava sendo perdido por intervalo de transmissão, o que gerava uma transmissão ruim, com muitas falhas na imagem e no áudio, causada pela grande perda de pacotes, o que foi constatado visualizando o vídeo.

Na probabilidade P2 ocorre algo parecido com o que ocorre na P1. Embora o número de pacotes perdidos tenha diminuído consideravelmente, eles ainda continuam

altos, mostrando muitos intervalos de transmissão com perdas de 4 e 5 pacotes, o que gera efeitos parecidos com os gerados no primeiro caso.

Na probabilidade P3 os números se mostraram bem idênticos ao anterior, com grande incidência de intervalos com perdas de 4 e 5 pacotes.

Na P5, há um número muito pequeno de perdas de pacote por intervalo, se mantendo praticamente em um máximo de 3. Isso gera uma transmissão boa e estável, que permite que, mesmo com essa perda de pacote, o conteúdo seja visualizado. O problema dessa probabilidade é que nela é mais difícil mostrar a adaptabilidade do programa.

A probabilidade P4 apresenta um ótimo desempenho, apresentando perdas baixas o que permite uma transmissão de boa qualidade, com algumas perdas um pouco elevadas o que permite demonstrar a adaptabilidade do algoritmo.

Com a conclusão de qual probabilidade de perda usar, o programa foi executado mais uma vez para cada qualidade de vídeo. Foram observados os 40 primeiros pacotes de relatório RTCP informando as perdas de pacotes RTP por intervalo de transmissão.

O gráfico na figura a seguir nos mostra o resultado:

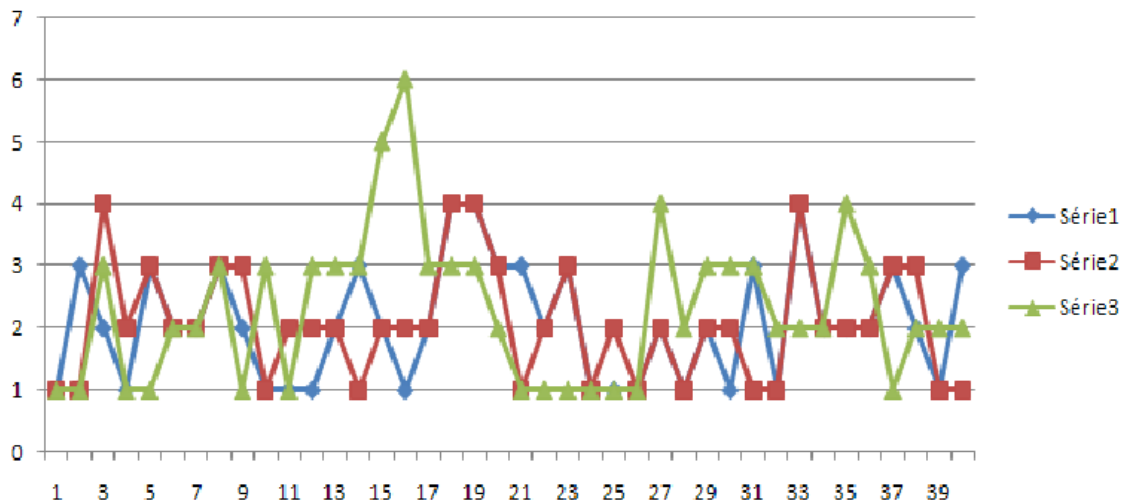


Figura 6.6: Gráfico Melhor Probabilidade

Pela figura 6.6, podemos observar uma distribuição para a melhor qualidade (série 3) de: 1 = 13; 2 = 10; 3 = 13; 4 = 2; 5 = 1; 6 = 1. Para a qualidade média (série 2) a distribuição fica sendo: 1 = 12; 2 = 17; 3 = 7; 4 = 4. E, finalmente, para a qualidade baixa (série 1) a distribuição é: 1 = 13; 2 = 14; 3 = 10; 4 = 3.

Observando-se esses valores, é possível ver que um valor ótimo para o número de pacotes perdidos na transmissão do conteúdo com a melhor qualidade é 4, ou seja, se a perda de pacotes for menor que esse valor não é necessário realizar adaptação, pois mesmo com a perda de pacotes, a transmissão ainda segue numa qualidade razoável. Valores maiores ou iguais a 4 e menores que 8 exigem uma mudança na qualidade do conteúdo transmitido, fazendo-o se adaptar, conectando-se ao conteúdo de qualidade média. Caso esse valor seja igual ou maior que 8 o conteúdo transmitido passará a ser exibido, automaticamente, com a qualidade mais baixa.

Para o conteúdo com qualidade média observou-se um valor ótimo para a perda de pacotes novamente igual a 4, ou seja, se a perda for maior ou igual a 4 o conteúdo passa a ser exibido na qualidade mais baixa. Por outro lado, se o número de pacotes perdidos for menor ou igual a 1 por duas vezes consecutivas, o conteúdo de melhor qualidade passa a ser exibido.

Se o conteúdo estiver sendo exibido com a qualidade mais baixa e o número de pacotes perdidos for igual a 1 por 3 vezes consecutivas, o conteúdo será exibido com qualidade média. Se o número de pacotes perdidos for igual a 0, o conteúdo passará a ser exibido na melhor qualidade disponível.

Desse modo, o Zapper pode alternar entre os grupos multicast, mudando a qualidade do conteúdo transmitido de acordo com esses valores, proporcionando sempre ao cliente a melhor qualidade disponível para o momento. Isso se mostra muito útil, principalmente em dispositivos terminais que possuem conexões muito instáveis, assim o cliente estará sempre se adaptando a essa instabilidade.

Vale lembrar que a probabilidade de perda de pacotes é um valor aleatório. Assim sendo, esses valores podem variar muito a cada execução do algoritmo. Portanto não são exatamente os melhores valores absolutos para a implementação, mas mostram a questão da adaptabilidade zelando também pela qualidade.

7 Considerações Finais

Levando-se em consideração todos os temas levantados nessa monografia, os quais contribuíram de forma essencial para se atingir o objetivo esperado, conclui-se que o IPTV representa um avanço no modo de se consumir conteúdo multimídia, seja ele ao vivo ou sob demanda, tendendo a substituir, gradativamente, a TV convencional. Apresenta uma série de vantagens em relação às formas convencionais de se assistir vídeos na internet: os fluxos são contínuos, diferentemente dos modos convencionais onde são gerados vários segmentos discretos de conteúdo, apresenta centenas de canais de programação, enquanto no modo convencional existem milhões de arquivos de conteúdo, apresenta um ou dois formatos de conteúdo selecionados pelo fornecedor, enquanto o modo normal apresenta dezenas de formatos e é distribuído por uma rede IP gerenciada, o que lhe garante qualidade enquanto no modo convencional a distribuição acontece na Internet pública, de melhor esforço.

Concluiu-se também que o RTP se mostrou eficiente na transmissão de streams, pois é um protocolo de tempo real que gera fluxos contínuos, diferentemente dos modos convencionais (como o padrão DASH), onde é feito vários segmentos discretos de conteúdo. Assim o RTP traz o conceito de um fluxo contínuo de pacotes, enquanto outros métodos usam o conceito de downloads sucessivos de segmentos discretos de conteúdo. Além do mais, as informações transportadas pelos pacotes RTP, como número de sequência e timestamp, são de suma importância para se mensurar a qualidade da transmissão.

Juntamente com o RTP, e talvez sendo a parte mais importante para o objetivo da presente monografia, os pacotes RTCP, principalmente os pacotes de relatório do receptor (RR), mostraram eficácia ao reportar ao servidor a qualidade da transmissão em curso, permitindo que sejam feitas rápidas adaptações visando melhorar a QoS da aplicação, proporcionando ao cliente uma melhor QoE, ou seja, permitindo que ele visualize os conteúdo multimídia da melhor forma possível.

O uso de um servidor separado para a implementação do Componente de Adaptação proposto, para monitorar as transmissões também teve bons resultados, uma vez que tira grande parte do processamento do cliente, o que é de suma importância, uma vez que esta-

mos lidando com dispositivos móveis, que possuem capacidade limitada de processamento e energia.

Como trabalhos futuros, vislumbra-se confeccionar e inserir uma implementação do protocolo RTSP (Real-Time Streaming Protocol) à essa atual implementação. Dessa forma, além de haver adaptação da transmissão dos streams, o cliente terá uma maior liberdade sobre o conteúdo exibido podendo pausar, retroceder, entre outras opções de métodos presentes nesse protocolo. Outra proposta é, diferentemente do que foi feito nesse trabalho, desenvolver uma implementação que proporcione adaptações em unicast para vídeos sob demanda no IPTV móvel. Assim, para que se mude a qualidade do conteúdo transmitido não basta mais apenas mudar o endereço multicast e a porta, uma vez que só estará sendo transmitido um fluxo. Assim, deve-se guardar o local exato no fluxo onde ocorrerá a adaptação e se iniciar esse novo conteúdo desse mesmo local o mais rápido possível, de modo a tentar fazer essa troca ser o mais imperceptível possível aos clientes de IPTV.

Referências Bibliográficas

- AKNOWLTON, C. Adaptive streaming comparison. 2010. Disponível em: <<http://www.iis.net/learn/media/smooth-streaming/adaptive-streaming-comparison>>. Acesso em: 12 set. 2014.
- APPLE. HTTP Live Streaming. 2014. Disponível em: <<https://developer.apple.com/streaming>>. Acesso em 30 set. 2014.
- FFMPEG. About FFmpeg. Disponível em: <<https://www.ffmpeg.org/about.html>>. Acesso em: 20 out. 2014.
- FROJDH, P. et al. Adaptative streaming within the 3GPP packet-switched streaming service. In: IEEE Network. V. 20, p. 34-40. IEEE Communications Society, 2006.
- GALVIS V. J. R. Qualidade de Serviço e de Experiência na Distribuição de Vídeo Escalável em Redes Par-a-Par (P2P). 2013. 179 f. Dissertação (Mestrado em Engenharia Elétrica) - Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília/DF. 2013.
- GT-IPêTeVê: Serviço de televisão IP de Alcance Global. 2014. Disponível em: <<https://sites.google.com/a/ice.ufjf.br/gt-ipeteve/home>>. Acesso em: 28 set. 2014.
- ITU-T. Draft Recommendation ITU-T Y.miptv-req - Functional requirements of mobile IPTV. Temporary Document 136 (WP 1/13). Novembro, 2013.
- ITU-T Y.1910. IPTV Functional Architecture. União Internacional de Telecomunicações. Setembro, 2008.
- ITU-T H.721. IPTV terminal devices: Basic model. União Internacional de Telecomunicações. Março, 2009.
- LIBSDL. About SDL. Disponível em: <<https://www.libsdl.org/>>. Acesso em: 15 out. 2014.
- MICROSOFT. Smooth Streaming. 2014. Disponível em: <<http://www.iis.net/downloads/microsoft/smooth-streaming>>. Acesso em: 01 out. 2014.
- OZER, J. What is Adaptive Streaming? 2011. Disponível em: <<http://www.streamingmedia.com/Articles/Editorial/What-Is-.../What-is-Adaptive-Streaming-75195.aspx>>. Acesso em: 10 set. 2014.
- ROCHA, R. M. Um Framework para Simulação da Negociação de Serviços em Redes Sem Fio de Nova Geração. 2008. 105 f. Dissertação (Mestrado em Informática) - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro. 2008.

SCHULZRINNE, H. et al. RTP: A transport protocol for Real-Time applications. In: Network Working Group, 2003.

SODAGAR, I. The MPEG-DASH Standard for Multimedia Streaming Over the Internet. IEEE MultiMedia. V. 18, issue 4, p. 62-67. IEEE Computer Society, 2011. DOI: 10.1109/MMUL.2011.71

VIDEOLAN. Disponível em: <<http://www.videolan.org/vlc/>>. Acesso em: 10 out. 2014.

WILSON, T. Encontrando e executando streaming de áudio e vídeo. 2014. Disponível em: <<http://tecnologia.hsw.uol.com.br/streaming-video-e-audio1.htm>>. Acesso em: 29 set. 2014.