Федеральное агентство по образованию
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Факультет информатики
Кафедра прикладной информатики

УДК 681-3

ДОПУСТИТЬ К ЗАЩИТЕ В ГАК
Зав. кафедрой, д.т.н., проф.
_____ С.П. Сущенко
«___» _____ 2005 г.

Бадаев Евгений Николаевич

РАЗРАБОТКА КЛИЕНТА-ПРОИГРЫВАТЕЛЯ ДЛЯ СИСТЕМЫ МЕДИАВЕЩАНИЯ

Дипломная работа

Научный руководитель Бирюков О.В.

Исполнитель,
студент гр. 1402 Бадаев Е.Н.

Электронная версия дипломной работы помещена
в электронную библиотеку. Файл

Администратор

Томск – 2005

Реферат

Дипломная работа 36 с., 11 рис., 4 табл., 8 источников, 2 приложения

СИСТЕМА МЕДИАВЕЩАНИЯ, MEDIA BROADCAST SYSTEM, ПРОИГРЫВАТЕЛЬ, DIRECTSHOW, TCP, UDP, RTP, RTCP, RTSP, MPEG-1, MPEG-2, МОДЕЛЬ COM

Объект разработки – проигрыватель мультимедиа потоков системы медиавещания

Цель работы – разработать и реализовать клиент-проигрыватель системы медиавещания, позволяющий принимать и воспроизводить мультимедиа потоки, передаваемые сервером системы, на клиентской станции

Метод разработки – экспериментальный на ЭВМ

Результаты работы: разработан и реализован клиент-проигрыватель для системы медиавещания, удовлетворяющий поставленным целям и реализующий удобное управление процессом воспроизведения мультимедиа потоков, передаваемых сервером системы; проигрыватель реализован отдельной динамической библиотекой, что позволяет встраивать его в другие подобные системы.

Содержание

Перечень условных обозначений

ПО – программное обеспечение

API – Application Programming Interface (программный интерфейс приложения)

COM – Component Object Model (модель компонентных объектов)

GUID – Globally Unique Identifier (глобально уникальный идентификатор)

UDP – User Datagram Protocol (протокол передачи дейтаграмм пользователя)

IP – Internet Protocol (протокол межсетевого взаимодействия)

RTP – Real-Time Transport Protocol (транспортный протокол реального времени)

RTCP – Real-Time Control Protocol (протокол управления передачей в реальном времени)

QoS – Quality of Service (качество и класс предоставляемых услуг передачи данных)

TCP – Transmission Control Protocol (протокол управления передачей)

ISN – Initial Sequence Number (начальный номер последовательности)

RTSP – Real-Time Streaming Protocol (потоковый протокол реального времени)

HTTP – HyperText Trasfer Protocol (протокол передачи гипертекста)

FTP – File Transfer Protocol (протокол передачи файлов)

MPEG – Motion Pictures Experts Group (экспертная группа по кинематографии)

HTML – HyperText Markup Language (язык гипертекстовой разметки)

Введение

В наше время все более широкое распространение получают цифровые технологии в области видео и аудио информации. С каждым годом все больше телевизионных каналов переходят на новый более качественный цифровой формат. Все новые видео фильмы выпускаются в цифровом формате на DVD. Эпоха аналоговых видео записей и телевизионных трансляций заканчивается.

Остается нерешенным вопрос о доставке высококачественного цифрового сигнала до абонентов. Линии связи кабельного телевидения позволяют передавать видео и аудио сигнал лишь в аналоговом виде. Это означает, что при использовании этих линий требуется обратно приводить сигнал к аналоговому виду, уничтожая таким способом все преимущества цифрового телевидения. Альтернативой может служить установка спутниковой антенны на стороне абонента, но на данный момент это очень дорогой вариант.

Но ведь можно передавать мультимедиа данные по оптоволоконной сети, по ADSL и прочим цифровым каналам. В этом случае становится возможным использование коллективных ресурсов. Например, можно использовать одну спутниковую тарелку, оцифровщики в местных телерадиокомпаниях, а также использовать единые серверы фильмов.

Целью данной работы является разработка клиента-проигрывателя для системы медиавещания, который должен по запросу устанавливать соединение с вещающим сервером, принимаю мультимедиа поток, отправляемый сервером, и воспроизводить его на компьютере клиента.

1 Анализ предметной области

1.1 Системы медиавещания

Мультимедиа информация поступает из различных источников: телевидение, радио, носители информации (DVD). Концепция построения систем медиавещания заключается в объединении этих источников медиаданных и предоставления удобного способа доставки мультимедиа до пользователя.

В мире уже существует довольно много систем позволяющих передавать аудио и видео через сеть Интернет. Большинство таких систем направлены на живое вещание, что удобно в случае предоставления актуальной информации без задержек. Однако, предоставление фильмов в таком виде довольно неудобно, т.к. пользователь может пропустить время трансляции, что впрочем может быть отнесено и к трансляции телепередач в сети. Стоит пользователю опоздать на несколько минут, он может не получить интересующую его новость или пропустить интересный эпизод фильма.

Поэтому не так давно зародилось новое направление в системах медиавещания: временное сохранение содержимого живого вещания на КЭШ-серверах для последующей доставки клиенту по его запросу, а также система видео-по-запросу, позволяющее пользователю просматривать интересующие его видео фильмы, клипы в любой момент с возможностью приостановки воспроизведения и позиционирования в потоке.

Специфичность таких систем, как правило, заключается в использовании их в пределах локальной сети (например, в гостиницах), когда абоненты экономят на средствах связи, таких как антенны, спутниковые тарелки, а также на видео дисках.

1.2 Выбор архитектуры проигрывателя

Разрабатываемая система ставит своей главной задачей доставку и презентацию мультимедиа информации клиенту. Поэтому очень важно правильно выбрать архитектуру клиентского проигрывателя. Важным аспектом данной архитектуры является гибкость, универсальность и относительная простота построения систем воспроизведения медиаданных на ее основе.

На данный момент существуют, пожалуй, две наиболее общие концепции построения программных медиа проигрывателей:

- *Монолитная*. При данном типе архитектуры все работы по воспроизведению мультимедиа информации производит одна программа/компонента. Она читает медиапотоки из файла или принимает их из сети, обрабатывает заголовки потоков, осуществляет позиционирование в потоке, разделяет смешанные потоки, декодирует сжатые данные и выводит полученные несжатые мультимедиа потоки на устройства вывода, синхронизируя различные потоки. Таким образом, в одной компоненте представлена довольно обширная функциональность, но из-за такой перегруженности система теряет гибкость и расширяемость, становится очень трудно внести какие-либо исправления и новую функциональность в готовый продукт.
- *Компонентная*. Эта архитектура основана на разделении процесса воспроизведения мультимедиа информации на отдельные семантически единые работы. Так, например, могут быть построены отдельные компоненты для чтения потока из файла, разделения смешанных потоков, декодирования какого-либо формата данных, визуализации несжатых потоков. В таких системах, как правило, существует внешняя компонента управляющая связями компонент осуществляющих основные работы и предоставляющая механизм синхронизации различных потоков.

Одной из таких компонентных архитектур является DirectShow. Он предназначен для построения систем мультимедийной обработки на базе платформы Microsoft® Windows® и уже содержит в себе массу реализованных компонент и структур данных, облегчающих разработку массивных систем.

2 Стандарты и технологии

2.1 Мультимедийный интерфейс DirectShow

2.1.1 Обзор архитектуры

DirectShow API - универсальная архитектура потоковой передачи мультимедиа данных, обеспечивающая оцифровку, обработку и воспроизведение мультимедиа информации, как на локальном компьютере, так и в глобальной сети Интернет в режиме реального времени.

Этот API позволяет приложениям, построенным для платформы Microsoft® Windows®, управлять различными устройствами ввода аудио- и видеоинформации. Он обеспечивает программную поддержку множества форматов хранения мультимедиа информации. DirectShow, предоставляет унифицированные интерфейсы управления выводом аудио и видео, а также управлением воспроизведением.

DirectShow упрощает процессы воспроизведения мультимедиа информации, преобразования форматов данных и захвата с внешних аудио- и видеоустройств.

Основное достоинство архитектуры DirectShow – ее расширяемость, что позволяет сторонним разработчикам поддерживать специализированные устройства, форматы данных и компоненты обработки информации.

Приведем типичные задачи, решаемые с использованием технологии DirectShow:

- видео и аудио захват
- прием и воспроизведение мультимедиа данных с различных источников таких как:
    - локальные файлы
    - сети Интернет вещания
    - различные внешние устройства
- обработка видео и аудио информации
- вывод мультимедиа данных на внешние устройства, а также вещание в сеть Интернет

DirectShow API основана на модели компонентных объектов (COM).

Модель COM.

COM – архитектура компонентного программного обеспечения, обеспечивающая возможность построения приложений и систем на основе компонент различных разработчиков.

Модель компонентных объектов:

- определяет бинарный стандарт взаимодействия компонент
- независима от языка программирования
- поддерживается на разнообразных платформах (Microsoft® Windows®, Microsoft Windows NT™, Apple® Macintosh®, UNIX®)
- предусматривает интенсивное развитие приложений, основанных на COM
- является расширяемой

COM обеспечивает механизмы взаимодействия компонент вплоть до распределенных в сетевом пространстве, управления разделяемой компонентами памяти, извещения о состоянии и ошибках, а также механизм динамической загрузки компоненты.

В отличие от C++ объектов компонентный объект никогда не получает прямого доступа к другому компонентному объекту. Вместо этого один объект получает доступ к другому объекту через указатели интерфейсов (наборов функций).

Интерфейс COM представляет собой строго определенный *контакт* между программными компонентами, предназначенный для предоставления небольшого семантически единого набора операций.

Свойства интерфейсов, перечисленные ниже, формируют основу взаимодействия компонент в модели COM:

- *любой клиент, использующий функциональность компонентного объекта, имеет дело только с указателями на интерфейсы, предоставляемые компонентным объектом*
- *компонентный объект может реализовывать множество интерфейсов* (более новые версии данного объекта могу дополняться новыми интерфейсами)
- *интерфейсы строго определены*, каждому интерфейсу соответствует уникальный идентификатор (GUID), при создании нового интерфейса разработчик также создает новый GUID, а клиент для получения указателя на интерфейс компонентного объекта использует GUID, соответствующий данному интерфейсу
- *интерфейсы неизменны*, интерфейс COM не имеет версии и, таким образом, он не подвержен конфликтам версий, вместо этого на базе старого интерфейса создается новый с помощью дополнения новых функции и/или модификации старых

Архитектура COM определяет специальный интерфейс IUnknown, предназначенный для унификации всех COM объектов, каждый компонентный объект должен реализовывать данный интерфейс. Данный интерфейс содержит лишь функции контроля количества ссылок на объект, это позволяет выгружать объект из памяти, когда никакой другой объект в нем уже не нуждается. Также интерфейс IUnknown содержит функцию QueryInterface, позволяющую клиенту динамически определить, поддерживается ли тот или иной интерфейс объектом, а также получить указатель на данный интерфейс.

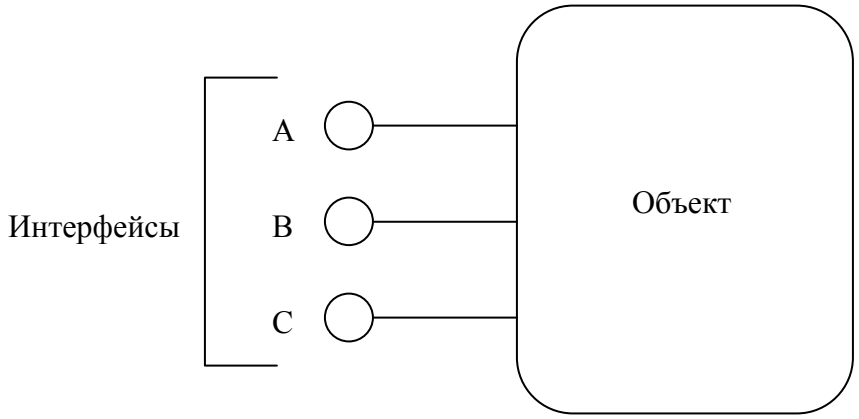

Рисунок 1. Компонентный объект, поддерживающий три интерфейса A, B и C

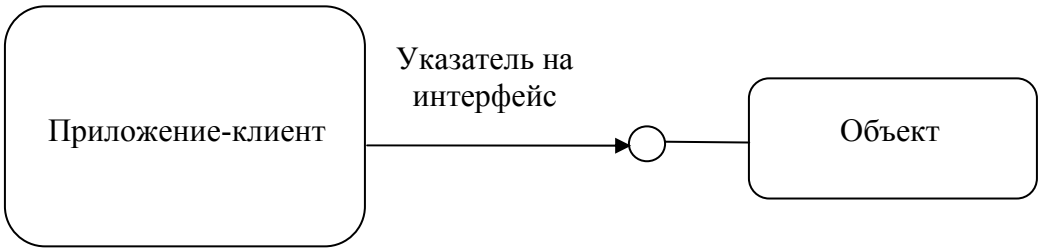

Рисунок 2. Взаимодействие клиента с компонентным объектом

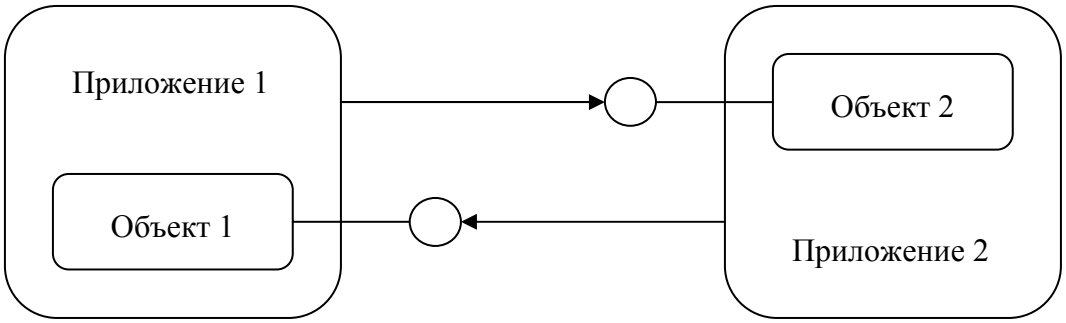

Рисунок 3. Использование объектов в других приложениях

2.1.2 Основные объекты DirectShow

DirectShow API основывается на разделении процесса обработки потоковых мультимедиа данных на отдельные стадии, выполняемые различными компонентными объектами, именуемыми фильтрами (filters).

Фильтры могу выполнять разнообразные задачи, перечислим наиболее общие и распространенные из них:

- чтение мультимедиа данных с локальных или распределенных носителей, а также прием мультимедиа вещания из сети Интернет
- захват мультимедиа потоков с внешних устройств
- разделение смешанных мультимедиа потоков
- смешение нескольких мультимедиа потоков в один
- сжатие и разжатие потоков
- шифрование данных
- обработка мультимедиа потоков
- вывод потоков на внешние устройства и в сеть Интернет

Фильтр может иметь несколько входов и несколько выходов, которые в терминологии DirectShow называются контактами (pins). Каждый контакт фильтра имеет свой список поддерживаемых форматов данных, к примеру, один контакт может принимать только видео поток в строго определенном формате, а другой может передавать аудио поток в нескольких различных представлениях.

Контакты служат для связи фильтров в графе фильтров (filter graph), который строится для выполнения какой-либо задачи: воспроизведения, аудио/видео захвата, монтажа мультимедиа данных.

Фильтры в зависимости от его роли в графе можно отнести к одной из трех категорий:

- *источники (source)* – фильтры, принимающие/читающие мультимедиа данные и представляющие их в граф
- *преобразователи (transform)* – фильтры, занимающиеся обработкой мультимедиа потоков
- *визуализаторы (renderer)* – фильтры, передающие мультимедиа потоки на устройства видео и аудио вывода, а также на внешние устройства и в сеть Интернет

Для управления графами фильтров в DirectShow существует COM объект менеджер графа фильтров (filter graph manager). Менеджер графа фильтров контролирует состояние графа фильтров, устанавливает эталонное время для всех фильтров (одним из применений которого является синхронизация различных потоков) и служит посредником во взаимодействии между приложением и фильтрами.

Менеджер графа фильтров может автоматически построить граф для воспроизведения. В случае более сложной структуры графа он обычно используется для построения части графа, тем самым, уменьшая объем работы, необходимый выполнить приложению.

Любой граф фильтров является:

- *ориентированным*, т.к. данные движутся по фильтрам в заданном направлении
- *ациклическим*, в графе фильтров не должно быть пути назад для обрабатываемых данных
- *несвязным*, так, к примеру, фильтр, занимающийся обработкой видео потока, не связан с фильтром, выводящим аудио на внешние устройства

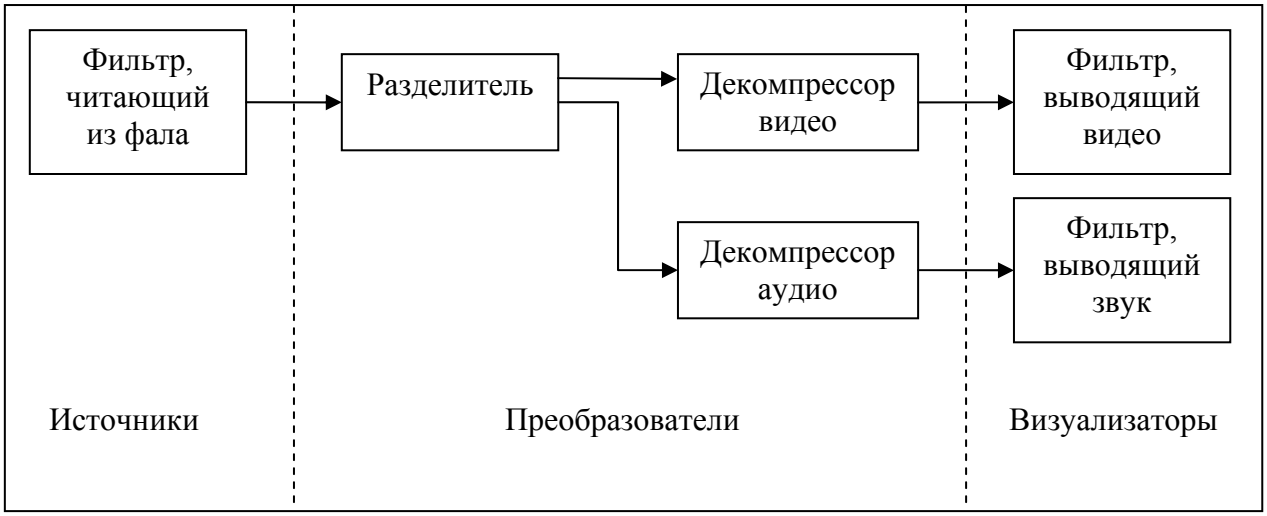

Рисунок 4. Типичный граф воспроизведения видео файла

2.2 Протоколы передачи мультимедиа данных

2.2.1 Протокол UDP

UDP предназначен для обмена дейтаграммами между процессами компьютеров входящих в единую сеть с коммутацией пакетов. Протоколом нижнего уровня для UDP является Internet (IP).

UDP предоставляет прикладному программному обеспечению процедуру посылки сообщений другим программам, при которой механизм протокола минимален. Данный протокол ориентирован на транзакции, получение дейтаграмм и защита от дублирования не гарантированы.

Заголовок UDP содержит следующие поля:

- *порт отправителя*: может содержать номер порта, с которого был отправлен пакет
- *порт получателя*: порт компьютера, на который пакет должен быть доставлен
- *длина*: длина дейтаграммы в байтах, включая заголовок и данные
- *контрольная сумма*: вычисляется по данным пакета, заголовку UDP-пакета, псевдозаголовку (информации от IP-протокола) и полям выравнивания по 16-битной границе (нулевым)

0 8	16 24	32 40	48 56	64
Порт отправителя	Порт получателя	Длина	Контрольная сумма	Данные

Таблица 1. Формат заголовка UDP

Основное преимущество протокола UDP – требование минимума установок и параметров соединения двух процессов между собой.

С точки зрения передачи мультимедиа данных в сети Интернет, не смотря на гарантию доставки пакетов протоколом управления передачей (TCP), приемлемым является протокол UDP, т.к. в отличие от TCP имеет быструю скорость передачи данных и практически не испытывает задержек, что весьма важно для зависимых от времени процессов.

2.2.2 Протокол RTP

RTP является транспортным протоколом, предназначенным для передачи мультимедиа данных, базирующимся на протоколе нижнего уровня IP. Данный протокол не включает функции маршрутизации, т.к. для этого применяется дейтаграммная служба UDP из стека протоколов TCP/IP. Вместе с протоколом RTCP он способен обеспечить сквозную доставку и необходимый уровень QoS для мультимедиа файлов.

RTP выполняет функции идентификации полезной нагрузки, нумерации последовательности пакетов и присвоения временных меток, что позволяет передавать мультимедиа трафик без потерь информации и временных задержек.

Данный протокол был разработан для многоадресной передачи. RTP-сессия устанавливается приложением-сервером, которое определяет набор адресов получателей, состоящих из адреса сети и номеров двух портов: для RTP и для RTCP. Каждому типу мультимедиа трафика соответствует отдельная сессия.

Протокол RTP также выполняет некоторые функции, свойственные протоколам уровня приложений, такие как упорядочение пакетов во времени, их реконструкция и синхронизация. Вся необходимая информация для выполнения этих функций находится в заголовке RTP.

В зависимости от загрузки сети может изменяться метод шифрования полезной нагрузки, информация об используемом методе также хранится в RTP заголовке. При формировании пакета полезная нагрузка «заворачивается» в RTP заголовок, полученный фрейм помещается в UDP заголовок, и все это инкапсулируется в IP заголовок. После чего готовый пакет передается через Интернет.

Протокол RTCP

RTCP выполняет функции управления и передачи управляющей информации протоколу RTP для диагностики и оптимизации производительности. Также он осуществляет контроль качества данных.

Основные функции RTCP:

- мониторинг QoS и управление перегрузками канала
- идентификация источника
- внутренняя синхронизация аудиовизуальной информации
- управление масштабированием

Пакеты пользователей содержат сведения, позволяющие их приложениям работать в сетях с низкой пропускной способностью, с буферизацией или без. Отправитель, анализируя пакеты обратной связи, генерируемые протоколом RTCP получателя, имеет возможность изменить скорость передачи данных.

Идентификация источника проводится за счет преобразования 32-битного значения соответствующего поля заголовка RTP в уникальное глобальное имя, идентифицирующее участника любой сессии.

Для внутренней синхронизации аудиовизуальной информации используются временные метки RTP и соответствующие им значения реального времени. Также RTCP производит масштабирование информации для ограничения сетевого трафика.

Для обеспечения функциональности RTCP генерируется ряд специальных управляющих пакетов, таких как:

- *Receiver Report (RR)*, пакеты статуса получателя с информацией о подтверждении получения пакетов, неустойчивости синхронизации между входящими пакетами, задержке, связанной с подтверждением приема
- *Sender Report (SR)*, пакеты статуса отправителя с данными о внутренней аудиовизуальной синхронизации и количестве отправленных байт
- *Source Description Items (SDES)*, пакеты с информацией об участниках сессии
- *BYE*, прощальный пакет отключения от сессии
- *APP*, пакет сведений о специфических функциях приложения

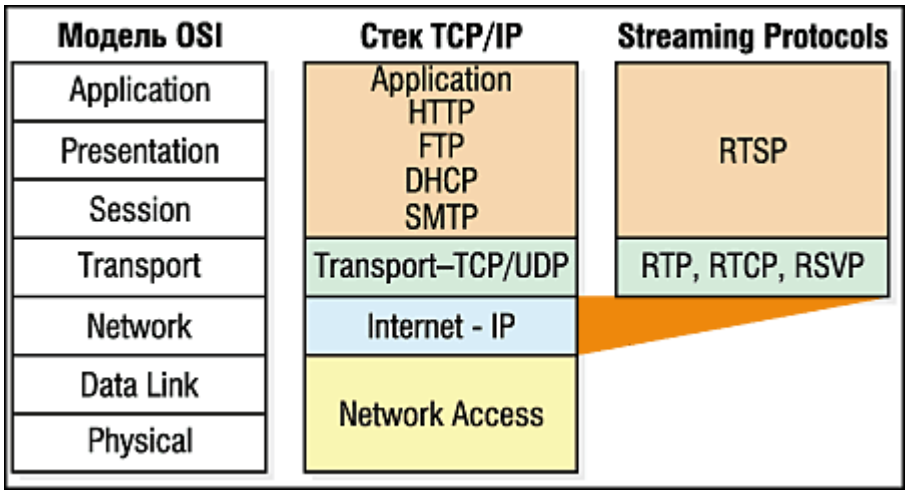

Рисунок 5. Стек потоковых протоколов

Рисунок 6. Структура пакета с полезной нагрузкой RTP

2.3 Протоколы управления передачей мультимедиа потоков

2.3.1 Протокол TCP

Протокол TCP в отличие от протокола UDP осуществляет доставку сегментов с установлением соединения. Он гарантирует доставку сообщения адресату, реализуя два основных механизма. Во-первых, в протоколе используются контрольные суммы пакетов для проверки их целостности, что освобождает прикладные процессы от необходимости использования таймаутов и повторных передач. Во-вторых, для отслеживания доставки сообщения адресату TCP реализует систему передачи подтверждений доставки на основе «окна». Как и UDP протокол TCP работает с портами, которые идентифицируют в системе процесс-отправитель и процесс-получатель.

Алгоритм использования окна передачи основан на передаче в заголовке TCP сегмента номера первого октета в поле данных пользователя. При значении флага SYN=1 в этом поле содержится начальный номер (ISN) последовательности, выбираемый для конкретного соединения (допускается и его случайное задание). Первому байту, передаваемому через установленное соединение, присваивается номер (ISN+1).

32-битовое поле номера следующего октета содержит код на единицу больший номера последнего успешно доставленного байта. Содерживое поля анализируется, только если присутствует флаг ACK.

В TCP предусмотрен режим полнодуплексной передачи, при которой данные могут передаваться в обоих направлениях независимо. В этом случае каждая из сторон отслеживает номера передаваемых и принимаемых байт. При благополучном приеме байтов с номерами (ISN+1)-N отправителю посылается отклик с номером следующего октета равным (N+1). Если же в следующий момент был получен пакет с номером первого октета равным (N+k), то в отклике опять отсылается номер следующего октета равный (N+1). Этот процесс позволяет определить на стороне отправителя потерю пакета.

Поле HLEN определяет длину заголовка в 32-разрядных словах. Это поле необходимо, т.к. в заголовке могут присутствовать опциональные поля переменной длины. Поле размера окна показывает, какое количество байт может послать отправитель без получения подтверждения доставки. Размер окна может варьироваться в течение сессии.

Поле контрольная сумма предназначено для обеспечения целостности сообщения. Как и в случае с протоколом UDP перед суммированием к TCP сегменту добавляется псевдозаголовок, включающий в себя часть заголовка IP.

Поле указателя важной информации представляет собой указатель последнего байта, содержащего информацию, требующую немедленного реагирования. Данное поле обрабатывается только при наличии флага URG, отмечающего сегмент с первым байтом «важной информации».

Протокол TCP подходит для управления процессом передачи мультимедиа потоков, т.к. для управления более важным аспектом является гарантия доставки сообщения адресату, нежели скорость передачи сообщения.

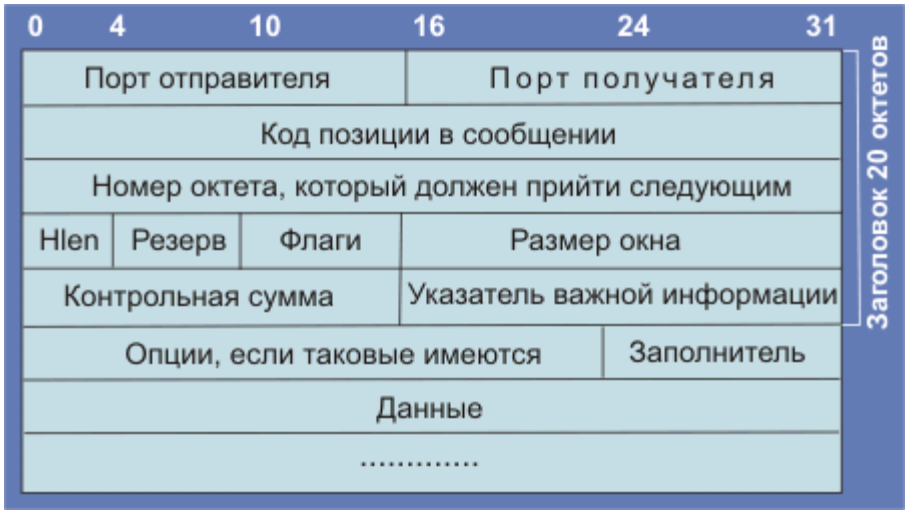

Рисунок 7. Формат TCP сегмента

Рисунок 8. Схема использования окна передачи

2.3.2 Протокол RTSP

RTSP – это протокол прикладного уровня, подобный HTTP и FTP в стеке протоколов TCP/IP. Данный протокол предназначен для управления мультимедиа потоком. Для него протоколами нижнего уровня могут быть RTP, TCP/UDP.

Также как и HTTP протокол RTSP обладает свойствами масштабируемости и взаимодействия. Каждая презентация, каждый мультимедиа поток в нем идентифицируются своим URL. Свойства презентаций и другие спецификации хранятся в файле дескриптора презентации, также имеющем свой URL.

Не смотря на существенное сходство протоколов RTSP и HTTP, они обладают некоторыми различиями. Самое важное отличие заключается в том, что в протоколе RTSP и сервер, и клиент могут генерировать запросы. К примеру, видео сервер может послать запрос на установку параметров воспроизведения определенного видео потока. Также протоколом RTSP предусматривается, что управление состоянием и связью осуществляет сервер. Третье отличие заключается в возможности передачи данных вне основной полосы (out-of-band) другими протоколами (например, протоколом RTP).

Сервис RTSP содержит набор инструкций, которыми обмениваются сервер и клиент, они отсылаются в виде RTSP пакетов, содержащих установочные параметры для мультимедиа потока.

Приведем некоторые из инструкций:

- *DESRIBE*, клиентский запрос на описание презентации/мультимедиа потока
- *ANNOUNCE*, серверная инструкция на обновление описания сессии в режиме реального времени
- *SETUP*, клиент запрашивает у сервера ресурсы и начинает RTSP сессию
- *PLAY*, запрос на начало передачи данных в потоке, выделенном командой SETUP
- *PAUSE*, запрос на временную приостановку доставки данных без освобождения ресурсов
- *TEARDOWN*, клиентский запрос на прекращение передачи данных и освобождение связанных с потоком данных

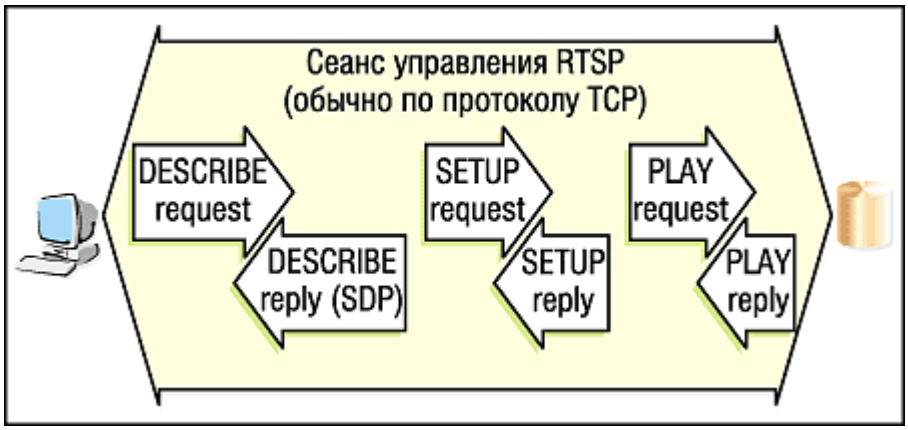

Рисунок 9. Пример сеанса управления по протоколу RTSP

2.4 Формат передачи мультимедиа потока MPEG-2

Предшественник этого формата – MPEG-1 вполне можно назвать революционным, т.к. до него фактически не существовало сжатых форматов видео данных. Как раз в то время был найден подходящий тип носителя, обладавший огромной вместительностью, – CD-ROM диски. Хотя фильм в формате MPEG-1 не вмещался на один диск, но ничто не мешало записывать его на два диска, ведь новинка была недорогой.

Первые CD-ROM проигрыватели были односкоростными и этим обусловлено ограничение скорости пересылки потока данных в формате MPEG-1 в размере 150Кб/сек. И несмотря на то что сам формат поддерживал сжатие видеоинформации с разрешением до 4095x4095 и частотой смены кадров 60Гц ограничение скорости передачи данных фактически определила потолок. Наиболее распространенными оказались оптимизированные форматы: SIF 352x240 с 30 кадрами в секунду и PAL/SECAM 352x288 с 25 кадрами в секунду.

Рассмотрим принципы сжатия видеоинформации в формате MPEG-1. в данном формате в качестве цветовой схемы используется YCbCr, где Y – яркостная компонента, а Cb и Cr – хроматические. Как правило, хроматические компоненты кодируются с меньшим разрешением, чем яркостная компонента, это обусловлено тем, что глаз человека лучше воспринимает слабое изменение яркости, нежели слабое изменение цвета. В MPEG-1 используются только варианты 4:2:0, 4:1:1 и 4:1:0.

Вариант кодирования	Отношение разрешений Cb/Y (Cr/Y) по горизонтали	Отношение разрешений Cb/Y (Cr/Y) по вертикали
4:4:4	1:1	1:1
4:2:2	1:2	1:1
4:2:0	1:2	1:2
4:1:1	1:4	1:1
4:1:0	1:4	1:4

Таблица 2. Варианты кодирования цветовой схемы

В процессе кодирования получаются три вида кадров:

1. *кадры типа I (Intra frame)* – ключевые кадры, сжимаемые без зменений
2. *кадры типа P (Predicted frame)* – кодируются с удалением части информации, а при воспроизведении используется информация предыдущих I или P кадров
3. *кадры типа B (Bidirectional frame)* – кодируются с еще большей потерей информации, а восстанавливаются на основе двух предыдущих I или P кадров

Цепочки формируемых кадров могут быть различными, наиболее распространенная последовательность выглядит так: IBBPBBPBBIBBPBBPBB…, в этом случае при воспроизведении следование кадров будет следующим: 1423765…

Все кадры при кодировании в формате MPEG-1 разбиты на блоки размером 8x8 пикселей. В случае кодирования I фрейма простого разбиения кадра на блоки для последующего сжатия достаточно, а при кодировании P и B фреймов применяется алгоритм предсказания движения. Он выделяет похожие блоки предыдущих I или P фреймов и текущего фрейма и вычисляет вектора движения блоков и разницу между текущим и предыдущими блоками, которая далее подвергается сжатию.

Само кодирование состоит из трех фаз:

- *Discrete Cosine Transformation (DCT) – дискретное преобразование косинусов* (преобразование Фурье)
- *Quatization – квантование* (перевод данных из непрерывной формы в дискретную)
- *Преобразование полученных данных в последовательность* (преобразование матричного вида в линейный)

DCT, используя коррелирующие эффекты (т.к. обычно пиксели в блоке и сами блоки связаны между собой), преобразует блоки в частотные Фурье-компоненты. Часть информации теряется за счет выравнивания некоррелирующих участков. Далее формируется матрица квантования, представляющая собой преобразованные в дискретную форму амплитуды частотных Фурье-компонентов. Происходит разбивка частотных коэффициентов на конкретное число значений, их точность фиксирована и составляет 8 бит. Многие коэффициенты в результате квантования обнуляются, а далее матрица переводится в линейный вид.

Звук в формате MPEG кодируется отдельным звуковым кодером. Форматы звука неоднократно эволюционировали, что в итоге привело к появлению трех типов звуковых кодеров формата MPEG-1: MPEG-1 Layer I, Layer II, Layer 3. все они основаны на психоакустической модели, которая достигла своей вершины в алгоритмах Layer 3.

Синхронизации аудио- и видеоданных осуществляется при помощи специального потока данных System stream, содержащего встроенный таймер. Таймер работает со скоростью 90 КГц и содержит два слоя:

- Слой таймера и служебной информации для синхронизации видеокадров с аудиотреком
- Компрессионный слой с аудио- и видеопотоками

Рассмотрим основные отличия потомка MPEG-2, являющегося дальнейшей эволюцией формата сжатия мультимедиа данных.

В MPEG-2 используется нелинейный процесс дискретно-косинусного преобразования. Теперь в процессе кодирования можно задавать точность коэффициентов матрицы квантования, такие как: 8, 9, 10 и 11 бит на значение. В формате MPEG-2 стало возможно загружать отдельную матрицу квантования перед каждым фреймом, что позволяет с меньшими потерями кодировать участки с движением.

Алгоритмы предсказания движения также расширили свою функциональность. Добавились новые режимы 16x8 MC, field MC и Dual Prime. Основной размер блока теперь может быть не только 8x8 точек, но и 16x16 и 16x8. были введены еще два варианта соотношения цветовых и яркостной плоскостей: 4:4:4 и 4:2:2. Всё это позволяет ставить ключевые кадры реже, повышая степень сжатия, с более качественной картинкой, чем в формате MPEG-1.

Новинкой для форматов сжатых видеоданных явились несколько ранее неиспользуемых алгоритмов сжатия:

- *Scalable Modes* – определение уровня приоритетов разных слое потока (больший приоритет = больший битрейт)
- *Spatial Scalability (пространственное масштабирование)* – кодирование базового слоя с меньшим разрешением, для предсказания движения приоритетных слоев
- *Data Partitioning (дробление данных)* – дробление блоков размером в 64 элемента матрицы квантования на два потока: высокоприоритетный, содержащий низкочастотные компоненты, и низкоприоритетный с высокочастотными составляющими
- *Signal to Noise Ratio Scalability (масштабирование соотношения сигнал/шум)* – кодирование разных по приоритету слоев с разным качеством
- *Temporal Scalability (временное масштабирование)* – уменьшение количества ключевых блоков для менее приоритетных слоев и внедрение дополнительной информации в высокоприоритетные слои для восстановления промежуточных кадров

название уровня	разрешение	максимальный битрейт	качественное соответствие
Low	352*240*30	4 Mbps	CIF, бытовая видео кассета
Main	720*480*30	15 Mbps	CCIR 601, студийное TV
High 1440	1440*1152*30	60 Mbps	4x601, бытовое HDTV
High	1920*1080*30	80 Mbps	Hi-End видеомонтажное оборудование

Таблица 3. Уровни приоритетов слоев потока

В области аудиокодирования в MPEG-2 появилось новое семейство форматов MPEG-2 AAC (Advanced Audio Coding). Добавились новые виды частот: 16, 22.05, 24 КГц, а такде появилась поддержка многоканальности.

Синхронизация аудио- и видео потоков теперь состоит из двух этапов:

- *Packetized Elementary Stream (PES)* – разбивка звукового и видео потока на пакеты
- Объединение пакетов в один из двух потоков:
    - *MPEG-2 Program Stream* – используется для локальных передач данных, полностью совместим с MPEG-1 System
    - *MPEG-2 Transport Stream* – предназначен для передачи транспортных пакетов (длиной 188 или 188+16 бит) двух типов (сжатые данные PES и сигнальную таблицу *Program Specific Information – PSI*) в каналах с большим количеством ошибок

Название	Разрешение	Комментарии
VCD	352*480*24 (progressive)	VHS
SVCD	544*480*30 (interlaced)	Laserdisc (LD), D-2, Качество как у PAL
DVD	704*480*30 (interlaced)	Качество CCIR 601.Studio D-1

Таблица 4. Наиболее популярные стандарты

3 Архитектура системы медиавещания

3.1 Общая структура системы медиавещания

Архитектуру разрабатываемой системы медиавещания можно условно разделить на 3 рабочие области:

- получение мультимедиа от источника (которыми могут являться пункты приема спутникового вещания, телерадиостанции, библиотеки видео фильмов и другие внешние источники)
- распределение мультимедиа потоков по промежуточным серверам временного хранения
- доставка запрошенного пользователем содержимого до клиентской станции (в качестве которой может выступать как персональный компьютер, так и телевизионная приставка)

На первом этапе происходит прием сигнала источника мультимедиа данных и его оцифровка (если получаемый сигнал имеет аналоговое представление), либо его перекодировка (если сигнал является цифровым). Если же в качестве источника выступает библиотека фильмов, то вместо приема сигнала имеет место чтение мультимедиа данных с носителей информации цифровой библиотеки.

Далее полученные мультимедиа потоки распределяются по серверам временного хранения. Эти сервера на определенное время сохраняют содержимое мультимедиа потоков живого видео, обеспечивая так называемый *сохраненный контент* (TimeShift), который позволяет пользователю просматривать запись прямой трансляции. Также они кэшируют мультимедиа потоки цифровых видео фильмов, поступающие с серверов хранения.

На заключительном этапе мультимедиа потоки доставляются клиенту серверами временного хранения, а программное обеспечение клиентской станции воспроизводит получаемую информацию.

Рассмотрим компоненты системы медиавещания на схеме [Рисунок 10]:

1. сервера живого видео
2. библиотеки фильмов
3. сервер индексации
4. кэширующие сервера
5. клиентские станции (персональные компьютеры и видео приставки)

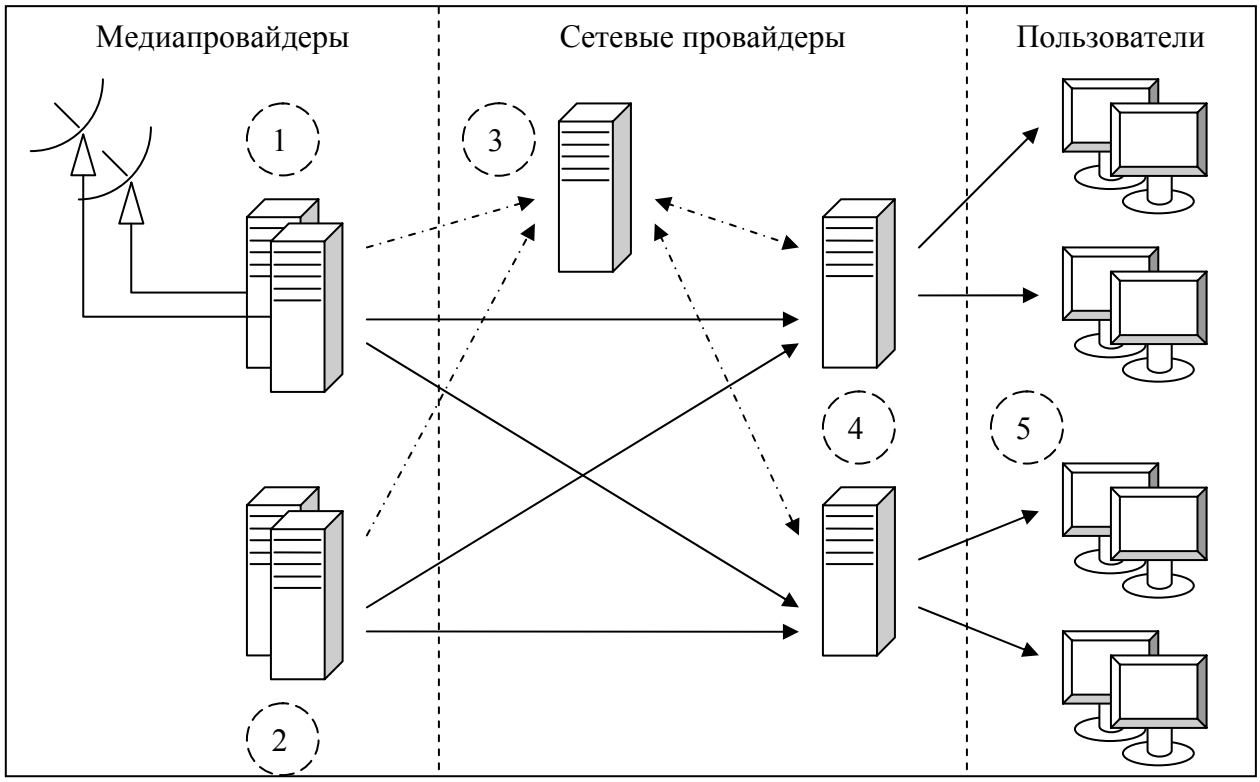

Рисунок 10. Структура системы медиавещания

3.2 Основные компоненты системы медиавещания

Рассмотрим более подробно функциональность основных компонент системы медиавещания.

Сервер живого видео

Данный сервер предназначен для приема живого мультимедиа сигнала в режиме реального времени. Как упоминалось выше, данный сервер производит оцифровку аналогового сигнала источника живого видео и перекодировку цифрового сигнала.

После преобразования формата мультимедиа потока сервер вещает его в сеть в формате MPEG-2.

Отличительной особенностью сервера является его расширяемость, которая позволяет принимать, обрабатывать и вещать в сеть одновременно сигналы нескольких источников.

Сервера живого видео, как правило, находятся у медиапровайдеров вблизи источников мультимедиа трансляций для минимизации расходов на ретрансляцию и во избежание потери качества принимаемого сигнала.

Сервер библиотеки цифровых видео фильмов

Этот сервис производит хранение фильмов в цифровом формате. Для совместимости с видео приставками форматом данных также является MPEG-2, однако, система позволяет хранить и передавать видео фильмы в любом формате. В последнем случае прием и воспроизведение такого видео возможен только на клиенте, установленном на персональном компьютере.

Передача мультимедиа потока с сервера библиотеки фильмов начинается только по запросу пользователя.

Сервер индексации

Сервер индексации является узлом системы медиавещания. Он производит учет всей мультимедиа информации в системе. Сервер хранит данные о содержимом библиотек цифровых фильмов, информацию о каналах живого вещания.

Данный сервис предоставляет информацию серверам временного хранения о местоположении запрашиваемых пользователем мультимедиа данных, а также предоставляет клиентам списки и описания присутствующих в системе ресурсов.

Сервер временного хранения

КЭШ-сервер фактически является источником мультимедиа потоков для клиентов системы медиавещания. По запросу пользователя он осуществляет передачу нужного мультимедиа содержимого на клиентскую станцию.

КЭШ-сервер осуществляет прием и временное хранение мультимедиа потоков с серверов живого видео, а также кэширование видео-по-запросу, принимаемое с серверов библиотеки фильмов. Таким образом, сервер временного хранения является ретранслятором потов мультимедиа данных. Однако, сам сервер является единственным источником сохраненного контента.

Географически такие сервера должны находиться в локальных сетях групп клиентов системы медиавещания. Благодаря такому размещению серверов уменьшается нагрузка на вычислительные сети во время просмотра многими клиентами живого вещания, т.к. в локальных сетях может быть организовано широковещание мультимедиа потоков.

Клиент системы медиавещания

Задача клиента системы медиавещания заключается в приеме мультимедиа потоков от серверов временного хранение и передача их на устройства вывода клиентской станции.

Клиентом может быть как программа на персональном компьютере, так и видео приставка. Однако следует отметить существенные различия клиентского программного обеспечения для компьютера от видео пристаки.

Видео приставка имеет возможность воспроизведения на телевизоре принимаемого мультимедиа потока, а также просмотра и навигации по электронным страницам в формате html. Таким образом, для предоставления клиенту возможности выбора нужного ресурса сервер формирует для видео приставки HTML страницу и обрабатывает HTTP запросы приставки. Также в компетенцию сервере в этом случае входит настройка параметров воспроизведения приставки по протоколу RTCP, передача мультимедиа данных по протоколу RTP и обработка запросов приставки по управлению воспроизведением по протоколу RTSP.

Программный клиент на персональном компьютере освобождает сервер от необходимости формирования HTML страниц, а запрашивает список и описание ресурсов по протоколу TCP, по этому же протоколу клиент посылает управляющие запросы во время воспроизведения. Доставка мультимедиа потока клиенту на компьютере осуществляется по протоколу UDP.

3.3 Работа системы медиавещания

I. Этап инициализации

На этом этапе происходит инициализация сервисов системы медиавещания. После того как какой-либо сервис прошел стадию подготовки, он начинает процедуру регистрации на сервере индексации (если сам сервер индексации готов к работе).

Для серверов источников мультимедиа данных, таких как сервер библиотеки фильмов и сервер живого видео, процесс регистрации заключается в уведомлении сервера индексации о текущих доступных ресурсах сервиса, их описании и статусе.

Серверы временного хранения на этом этапе получают списки доступных в системе ресурсов и начинают кэширование живого видео в соответствии с установленных заданием.

Архитектура системы медиавещания такова, что любой сервис может отключаться и включаться в произвольный момент времени, не влияя на работоспособность остальных компонент системы (кроме конечно клиентов системы: «падение» сервера временного хранения для которых является отказом сервиса).

II. Рабочий этап (описание работы системы с программным клиентом)

По запросу пользователя клиент системы медиавещания связывается с КЭШ-сервером и получает список доступных в системе ресурсов дополненный списком сохраненного содержимого. Данные списки содержат информацию о ресурсе, его идентификатор и сведения об источнике, предоставляющем данный ресурс.

При выборе пользователем того или иного ресурса на проигрывание клиент посылает запрос на КЭШ-сервер. Если в КЭШе сервера временного хранения присутствует данный ресурс или запрошенная его часть, то сервер сразу начинает передавать поток клиенту. Если же ресурс отсутствует на сервере, то он по информации об источнике ресурса связывается с последним и начинает принимать и ретранслировать поток клиенту. В процессе ретрансляции ресурса клиенту сервер временного хранения производит кэширование ресурса.

Сохраненный контент, находящийся на КЭШ-сервере, вторично не кэшируется и передается напрямую клиенту.

В случае с живым вещанием клиент запрашивает сервер временного хранения только о предоставлении доступа к широковещанию мультимедиа канала. При получении сервером первой заявки на живое вещание, он начинает отправлять соответствующий поток в сеть по широковещательному адресу. Если в локальной сети, обслуживаемой КЭШ-сервером не остается ни одного клиента смотрящего какой-либо канал живого видео, то широковещание этого канала в сеть прекращается до следующего запроса.

Сервер временного хранения периодически очищает кэш долго не использующихся ресурсов.

4 Архитектура клиента системы медиавещания

Клиент системы медиавещания состоит из двух основных компонент:

- компонента управления визуальным интерфейсом пользователя
- компонента проигрывателя мультимедиа потоков

Рассмотрим работу клиента.

Первая компонента занимается обработкой событий визуального интерфейса клиента. По запросу пользователя она связывается с сервером временного хранения для получения списка доступных ресурсов в системе и представляет его пользователю. При выборе пользователем какого-либо ресурса на проигрывание компонента визуального интерфейса передает запрос проигрывателю.

Проигрыватель, получив запрос на получение мультимедиа потока, посылает запрос КЭШ-серверу на запуск передачи потока указанного пользователем ресурса. Получив уведомление о готовности сервера к началу передачи, компонента проигрывателя строит граф фильтров DirectShow с использованием специального фильтра-источника, настроенного на текущее соединение с сервером временного хранения. По окончании процесса инициализации компонента запускает граф на проигрывание мультимедиа потока в окне, указанном компонентой визуального интерфейса.

В процессе воспроизведения компонента визуального интерфейса опрашивает компоненты проигрывателя о состоянии графа и текущем времени проигрывания, а также посылает управляющие запросы пользователя.

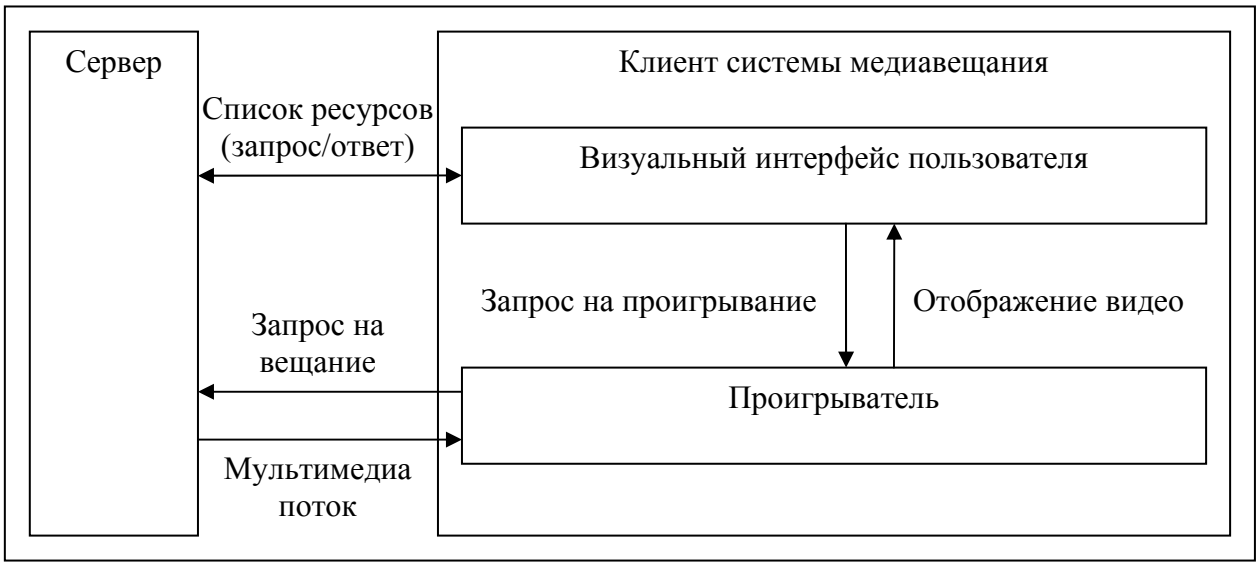

Рисунок 11. Структура клиента медиавещания

Заключение

В результате данной работы реализован программный клиент-проигрыватель для системы медиавещания, устанавливающий по запросу пользователя соединение с мультимедиа сервером, принимающий входящий поток и проигрывающий его на пользовательской системе.

Разработана компонента проигрывателя, выполненная в виде динамической библиотеки, обеспечивающая основную функциональность клиенту системы медиавещания.

Основные особенности компоненты проигрывателя:

- способность проигрывать потоки формата MPEG-2
- возможность быстрого расширения на проигрывание любой мультимедиа потока
- возможность задания произвольного видео окна
- поддержка отношения ширины и высоты видео окна
- поддержка перехода в полноэкранный режим проигрывания и обратно

Компонента проигрывателя была также успешно внедрена в другую медиа систему.

Список использованных источников и литературы

1. MSDN (Microsoft Developer Network) Library – July 2004
2. DirectX 8.1 Documentation for C++
3. Postel J. User Datagram Protocol, RFC 768. – USC/Information Sciences Institute, август 1980.
4. Бараш Л. Видео потоки. Протоколы // Компьютерное Обозрение. – 2003. – № 33.
5. Семенов Ю.А. Протоколы Интернет. Энциклопедия. – М.: Горячая линия-Телеком, 2001. – 1100 с.
6. Дан Гьен Семейство форматов MPEG. Часть первая – MPEG-1. – 20.11.2000 // http://www.3dnews.ru/reviews/multimedia/mpeg/
7. Дан Гьен Семейство форматов MPEG. Часть вторая – MPEG-2. – 14.02.2001 // http://www.3dnews.ru/reviews/multimedia/mpeg2/
8. David Heppe Automating media ingest for broadcasters. - сентябрь 2002 // http://www.telestream.net/news/pdfs/IBE%20Sep2002.pdf

Приложение А. Руководство пользователя

Компонента проигрывателя для системы медиавещания организована в виде димамической библиотеки, предоставляющей набор функций по управлению проигрыванием мультимедиа потока.

Список функций с описанием аргументов приводится в **Листинге структур и функций библиотеки клиента-проигрывателя**.

Для инициализации библиотеки проигрывателя необходимо первой вызвать функцию `SetConfig`, передав ей в качестве аргументов имя сервера временного хранения и дескриптор окна в которое будет выводиться видео изображение.

После инициализации библиотеки становятся доступными функции запуска проигрывания. Для вызова одной из таких функций помимо указания соответствующих параметров (идентификатора канала/фильма, времени смещения в сохраненном контенте и т.п.) необходимо передать указатель на экземпляр структуры `MPVideoConfig`. флаги данной структуры необходимо установить в соответствии с требуемыми параметрами изображения. При удачном выполнении функции запуска проигрывания в данной структуре возвращаются искомые параметры видеопотока: ширина и высота, а также продолжительность при запуске на проигрывание фильма из цифровой библиотеки.

После начала проигрывания медиапотока управление осуществляется тремя функциями: `Pause`, `Play`, `Stop`. Первые две работают в тандеме: одна ставит воспроизведение в состояние паузы, вторая переводит его обратно в состояние воспроизведения. Последняя функция останавливает воспроизведение, разрывается соединение с сервером, уведомляя его об остановке проигрывания медиапотока.

Состояние воспроизведения можно получить с помощью функции `GetState`.

Текущее время воспроизведения получается через функцию `GetCurTime`.

Очень важной для корректного отображения видео в окне проигрывателя является функция `RePaintWnd`, которая должна вызываться обработчиком визуального интерфейса пользователя при изменении размеров и положения родительского окна. Также данная функция используется при переходе в полноэкранный режим проигрывания и возврате из него.

При завершении передачи потока сервером компонента отсылает родительскому окну сообщение `WM_STOPPED`.

При закрытии приложения проигрывателя библиотека автоматически завершает воспроизведение и разрывает связь с КЭШ-сервером.

Приложение Б. Руководство программиста

Компонента проигрывателя системы медиавещания является набором функций и структур данных, отвечающих за прием и воспроизведение мультимедиа потоков, передаваемых клиенту серверов временного хранения. Она реализована в виде динамической библиотеки.

Описание работы «интерфейсных» функций приводится в **Приложении А** и **Листинге структур и функций библиотеки клиента-проигрывателя**.

Компонента основана на архитектуре DirectShow и использует менеджер графа фильтров для воспроизведения медиапотока, принимаемого с КЭШ-сервера.

Приемом вещания сервера и передачей потока в граф занимается специальный фильтр, использующий установленное управляющее соединение клиента с сервером временного хранения для управления передачей потока.

При вызове функции `RePaintWnd` компонента изменяет параметры видео окна, учитывая установленные флаги.

Все экземпляры переменных и структур данных, используемых компонентой во время проигрывания медиапотока глобально определены и начально инициализируются нулевыми значениями.

Все «интерфейсные» функции защищены критическими секциями.

Листинг структур и функций библиотеки клиента-проигрывателя

```
// переключение в полноэкранный режим и обратно
#define     MPVIDEO_FULLSCREEN      0x00000001
// поддержка отношения высоты и ширины видео изображения
#define     MPVIDEO_ASPECTRATIO     0x00000002

// сообщение о прекращении сервером трансляции мультимедиа потока
#define     WM_STOPPED              ( WM_USER + 100 )

// состояние проигрывателя
enum  PLAYSTATE
{
      PS_None     = 0,  // проигрыватель неинициализирован
      PS_Stopped  = 1,  // проигрывание остановлено
      PS_Paused   = 2,  // проигрывание приостановлено
      PS_Running  = 3   // проигрывание запущено

};

// установки видео окна и параметры воспроизведения
struct MPVideoConfig
{
      int   nVideoHeight;     // высота видео окна
      int   nVideoWidth;      // ширина видео окна
      INT64 llVideoDuration;  // длительность видео фильма
      int   nFlags;           // флаги управления видео окном
};

// функция инициализации проигрывателя
//          hWnd        установка видео окна
//          pszCSIP     имя сервера
int SetConfig( HWND hWnd, LPSTR pszCSIP );
```

```
// функция запуска воспроизведения выбранного видео фильма
//          nMovieID          идентификатор фильма
//          nFormatID         идентификатор формата потока
//          pVideoProps       конфигурация видео окна
int StartMovie( int nMovieID, int nIP1, int nIP2, int nIP3, int nIP4,
                int nFormatID, MPVideoConfig* pVideoProps );

// функция запуска просмотра сохраненного содержимого
//          nChannelID        идентификатор канала
//          llStartTime       эталонное время начала воспроизведения
//          nFormatID         идентификатор формата потока
//          pVideoProps       конфигурация видео окна
int StartTS( int nChannelID, int nIP1, int nIP2, int nIP3, int nIP4,
             INT64 llStartTime, int nFormatID, MPVideoConfig* pVideoProps );

// функция запуска просмотра сохраненного содержимого с предустановленного
// эталонного времени
//          nChannelID        идентификатор канала
//          nFormatID         идентификатор формата потока
//          pVideoProps       конфигурация видео окна
int StartTSFromPreset( int nChannelID, int nIP1, int nIP2, int nIP3, int nIP4,
                       int nFormatID, MPVideoConfig* pVideoProps );

// функция запуска воспроизведения живого вещания
//          pszMulticastIP    широковещательный адрес потока
//          pVideoProps       конфигурация видео окна
int StartLive( LPSTR pszMulticastIP, MPVideoConfig* pVideoProps );

// функция приостановки воспроизведения
int Pause( void );

// функция запуска воспроизведения
int Play( void );

// функция остановки воспроизведения и отсоединения от сервера
int Stop( void );
```

```
// функция изменения позиции воспроизведения в потоке
//          llRequest   эталонное время новой позиции
int SeekAtTime( INT64 llRequest );

// функция получения текущей позиции в потоке
//          pllRequest  эталонное время текущей позиции
int GetCurTime( INT64* pllRequest );

// функция перерисовки содержимого видео окна, должна вызываться при изменении
// размеров или местоположения родительского окна
//          pVideoProps       конфигурация видео окна
int RePaintWnd( MPVideoConfig* pVideoProps );

// функция получения состояния проигрывателя
//          pnState     состояние проигрывателя
int GetState( int* pnState );

// функция получения начального и конечного эталонного времени сохраненного
// содержимого
//          pllStart    начальное эталонное время
//          pllStop     конечное эталонное время
int GetTimeShift( INT64* pllStart, INT64* pllStop );

// функция получения начального и конечного эталонного времени сохраненного
// содержимого для заданного канала
//          nChannelID  идентификатор канала
//          pllStart    начальное эталонное время
//          pllStop     конечное эталонное время
int GetTSForChannel( int nChannelID, int nIP1, int nIP2, int nIP3, int nIP4,
INT64* pllStart, INT64* pllStop );
```