Оглавление

[ВВЕДЕНИЕ 3](#_Toc73101246)

[1 ДОСТАВКА ВИДЕО КОНТЕНТА 7](#_Toc73101247)

[1.1 Сценарии доставки 7](#_Toc73101248)

[1.1.1 Просмотр видео в интернете 7](#_Toc73101249)

[1.1.2 Видеосвязь, конференции 8](#_Toc73101250)

[1.1.3 Трансляции с веб-камер 10](#_Toc73101251)

[1.1.4 Стриминг 11](#_Toc73101252)

[1.2 Существующие решения 12](#_Toc73101253)

[1.2.1 Определение области рассматриваемых решений 12](#_Toc73101254)

[1.2.2 Программные энкодеры 13](#_Toc73101255)

[1.2.2.1 Open Broadcast Software 13](#_Toc73101256)

[1.2.2.2 XSplit 15](#_Toc73101257)

[1.2.3 Программы для видеоконференции 15](#_Toc73101258)

[1.2.3.1 Google Meet 15](#_Toc73101259)

[1.2.3.2 Zoom 16](#_Toc73101260)

[1.3 Выводы 16](#_Toc73101261)

[2 СРЕДСТВА ОРГАНИЗАЦИИ ПЕРЕДАЧИ ПОТОКОВОГО ВИДЕО 18](#_Toc73101262)

[2.1 Средства композиции сцены 18](#_Toc73101263)

[2.1.1 Вручную 18](#_Toc73101264)

[2.1.2 Использование библиотек 18](#_Toc73101265)

[2.2 Методы доставки видео контента 18](#_Toc73101266)

[2.2.1 rtsp, rtmp, rtp, sdp 18](#_Toc73101267)

[2.2.2 WebRTC 18](#_Toc73101268)

[2.2.3 hls, mpeg-dash 18](#_Toc73101269)

[2.3 Библиотеки кодеков для транскодирования 18](#_Toc73101270)

[2.3.1 ffmpeg 18](#_Toc73101271)

[2.3.2 vlclib 18](#_Toc73101272)

[2.3.3 gstreamer 18](#_Toc73101273)

[2.4 Выводы 18](#_Toc73101274)

[3 РАЗРАБОТКА РЕШЕНИЯ ДЛЯ СОВМЕСТНОГО ВИДЕО СТРИМИНГА 19](#_Toc73101275)

[3.1 Описание архитектуры 19](#_Toc73101276)

[3.2 Метод использования WebRTC – RTMP 19](#_Toc73101277)

[3.3 Метод использования RTMP – WebRTC 19](#_Toc73101278)

[3.4 Выводы 19](#_Toc73101279)

[4 РЕЗУЛЬТАТЫ РАЗРАБОТКИ 20](#_Toc73101280)

[ЗАКЛЮЧЕНИЕ 21](#_Toc73101281)

[СПИСОК ЛИТЕРАТУРЫ 22](#_Toc73101282)

[Глоссарий 23](#_Toc73101283)

ВВЕДЕНИЕ

Сложно представить жизнь современного человека без интернета и технологий, которые его образуют. Существует огромное количество возможностей, которые предоставляет нам интернет: доступ к свободной информации в любое удобное время, возможность отсылать и получать электронные письма, возможность участвовать в телеконференциях и удаленных встречах, возможность совершать покупки, не выходя из дома, играть в сетевые компьютерные игры и многое другое.

Среди прочего большую значимость для людей имеет потоковое вещание. Потоковое вещание — это мультимедиа контент (аудио-, видео-, а также текстовые данные), которые пользователи непрерывно получают от поставщика услуг. Это в том числе применимо к аналоговым средствам вещания наподобие радио или телевидения, хотя таковые отношение к интернету имеют посредственное.

В рамках данной работы наибольшее значение имеет потоковое видео вещание в режиме реального времени. Наиболее важные решения, интересующие нас в данной работе таковы:

1) Массовый онлайн стриминг. Под данной категорией подразумеваются платформы способные транслировать видео в реальном времени во всемирную паутину обширному, условно неограниченному количеству пользователей (ограничивают количество пользователей технические возможности платформы). Наиболее популярные сервисы — YouTube, Twitch, Facebook Live и прочие.

2) Видеоконференции. Под данной категорией подразумеваются решения, обеспечивающие двустороннюю передачу, обработку, преобразование и представление видеоданных между несколькими пользователями, количество которых ограничено, независимо от их месторасположения в режиме реального времени. Skype, Zoom, Google Hangouts Meet — наиболее яркие решения, которые предоставляют данный формат вещания.

Можно сказать, что данные категории узконаправлены в некотором понимании. Массовый онлайн стриминг предоставляет возможность вести трансляцию только одного медиа-потока, но этот поток доступен неограниченному количеству пользователей. Более того, данный видеопоток может обладать высоким качеством, например иметь высокое разрешение. Это в свою очередь говорит о том, что данный поток будет иметь больший объем данных, и, если мы захотим производить трансляцию видео в реальном времени в высоком качестве будет присутствовать ощутимая задержка.

Видеоконференции так же позволяют производить только один медиа-поток данных для одного пользователя, но так как количество пользователей больше одного, таких потоков может быть несколько. Так же стоит отметить, что количество участников ограничено, а конференция зачастую закрыта для широкой публики. Видео качество же тут сравнимо ниже, чем у массового онлайн стриминга, что дает возможность сократить задержку видео.

Отдельного упоминания достойна проблема создания так называемой сцены (отображения того, как будет отображаться видео, если помимо кадров, захваченных устройством захвата, будут присутствовать дополнительные кадры или элементы) – отсутствует возможность многофункционального администрирования таковой по умолчанию. В видеоконференциях возможен вариант локального упорядочивания медиа-потоков отдельных людей. В общем случае сцену необходимо создавать лично, с использованием сторонних приложений.

Возможность участия нескольких людей в одном медиа-потоке также ограничена и зачастую сводится к одному из следующих вариантов:

1. непосредственное нахождение людей в пределах видимости камеры захвата медиа-потока;
2. комбинирование меда-потоков в один прим помощи стороннего программного обеспечения, при этом возможны следующие варианты захвата этих потоков:

a. производится запись на одном аппаратном устройстве с использованием разных устройств захвата кадров (например, разные камеры на одном персональном компьютере; захват экрана параллельно с захватом изображения камерой так же можно отнести к данной категории, несмотря на то что участник может быть один),

b. производится запись на различных аппаратных устройствах внутри локальной сети, при этом производится вещание одного потока в аппаратное устройство поставщика основного медиа-потока,

c. аналогично предыдущему за исключением нахождения устройств в сети интернет.

Очевидны недостатки данных подходов: для первого необходимо физическое присутствие людей в одном месте, для второго необходима установка стороннего программного обеспечения, а также сложности с динамическим количеством участников (выходом может быть использование различных предустановленных сцен для определенного количества участников, что так же неудобно).

Среди существующих платформ нет такой, которая сочетала бы свойства обеих категорий платформ потокового вещания в режиме реального времени: возможность участия нескольких пользователей, медиа-поток которых бы мог собираться в единый, возможность администрирования потоков пользователей и создания сцены (например, динамическое размещение потоков в одной сцене при изменении количества потоков).

Более того, среди существующих решений существует строгая привязанность к

Целью данной работы является обеспечение возможности динамической композиции транслируемого видеопотока под управлением нескольких пользователей.

Для достижения данной цели были поставлены следующие задачи:

1. проанализировать существующие средства организации передачи потокового видео в реальном времени (массовый онлайн стриминг и конференции);
2. изучить способы организации совместных трансляций с помощью существующих средств и выявить недостатки;
3. проанализировать средства, используемые для организации передачи потокового видео (композиция сцены, библиотеки кодеков для транскодирования, протоколы передачи);
4. разработать решение для совместного видео-стриминга.

1. ДОСТАВКА ВИДЕО КОНТЕНТА

1.1 Сценарии доставки

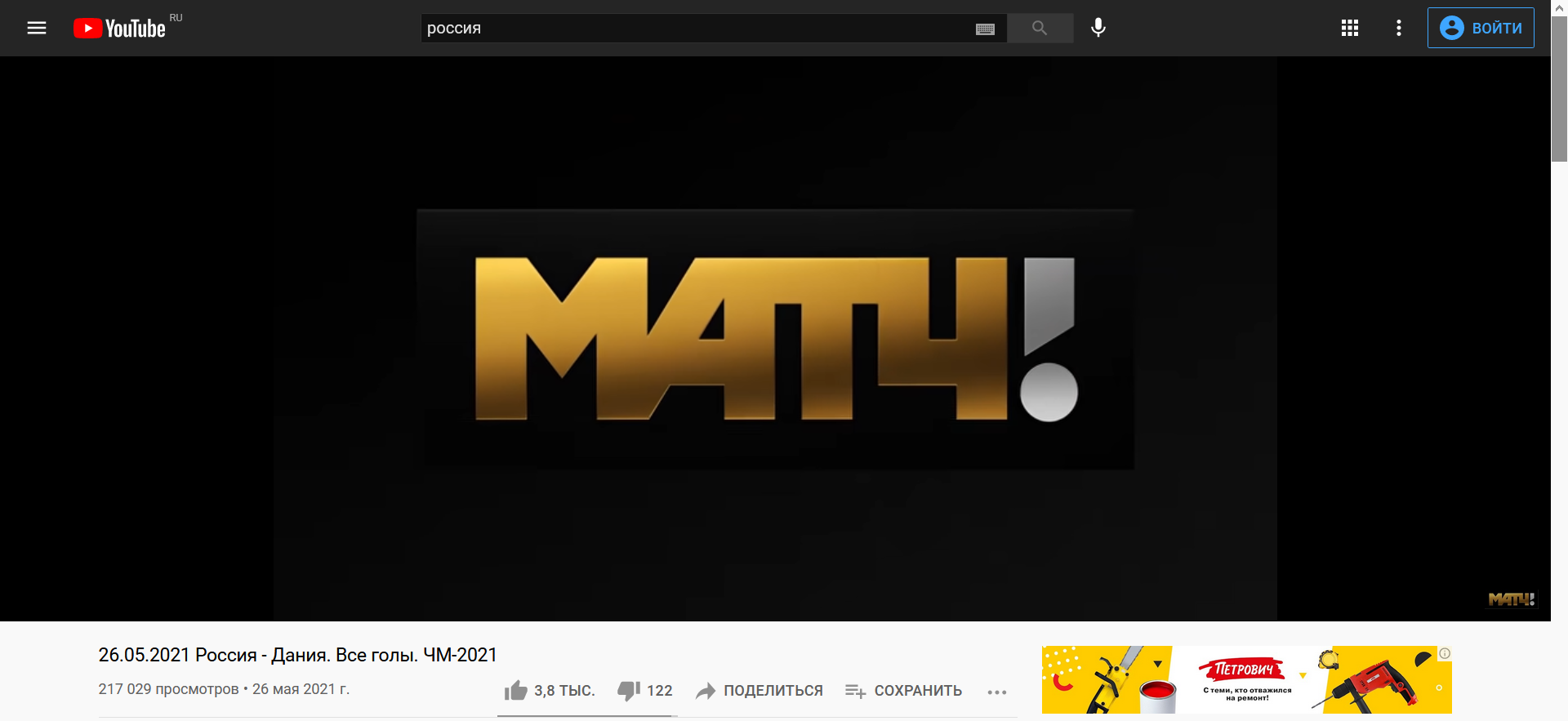
1.1.1 Просмотр видео в интернете

Как известно, в интернете найдется все — нужно лишь хорошенько поискать. Пользователи интернета имеют доступ к миллионам видео файлов, расположенных в глобальной сети. В любое удобное время можно посмотреть любимый фильм или сериал, смешной видеоролик именно благодаря данному сценарию доставки видео контента.

Данный сценарий доставки контента является наиболее распространенным и востребованным. Он представляет из себя предоставление видео файлов по запросу по распространенным протоколам передачи, например по UDP или TCP. Наиболее простой пример – HTML-5 видео, который используется в веб приложениях, где видео интегрировано при помощи тега <video> с указанием расположения видеофайла. Количество решений, предоставляющих данный сценарий доставки видео контента, довольно обширно. Самый популярный поставщик данного сценария – YouTube.

Среди прочих также популярны стриминговые сервисы и онлайн кинотеатры, предоставляющие доступ к видео по подписке. Примерами таких сервисов являются Netflix, Okko, Ivi, Megogo и прочие.

На рисунке 1 приведен пример данного сценария.



1. Сценарий доставки видео «просмотр видео» на примере видео платформы YouTube

Имеет смысл рассмотреть возможную организации данного сценария. Имеется сервер, который предоставляет доступ к видео материалу, например к файлу формата mp4 который хранится где-то на сервере. Клиентом данного сервера является любой имеющий доступ к серверу пользователь.

1.1.2 Видеосвязь, конференции

Видеосвязь — это сессия переговоров между ее участниками, во время которой участники могут слышать и видеть друг друга. Данное понятие очень схоже с видеоконференцсвязью, но значения у них разительные, в основном из-за сферы их применения. На данный момент видеосвязь является более публичным, массовым сценарием, и используется преимущественно в личных целях. Когда мы общаемся по видеозвонку со своими друзьями или родственниками в непринужденной обстановке, нам нужно слышать и видеть их в первую очередь. Качество звука обеспечивается посредством хороших микрофона и динамиков, а для четкой картинки мы выбираем веб-камеру, чего вполне достаточно. Для сеанса видеоконференцсвязи же такого оборудования будет не хватать. Видеоконференцсвязь является инструментом ведения бизнеса, поэтому требования у нее гораздо выше.

Существует множество сервисов, которые предоставляют видеосвязь. Те же социальные сети, например Facebook, VK или WhatsApp, уже давно завоевали доверие пользователей в данной области. Стоит отметить, что существуют такие решения, которые не требуют наличия другого ПО кроме браузера для обеспечения видеосвязи. Одно из таких решений – это Google Hangout Meet, пример которого приведен на рисунке 2.



1. Сценарий доставки видео «видеосвязь, конференции» на примере сервиса Google Duo

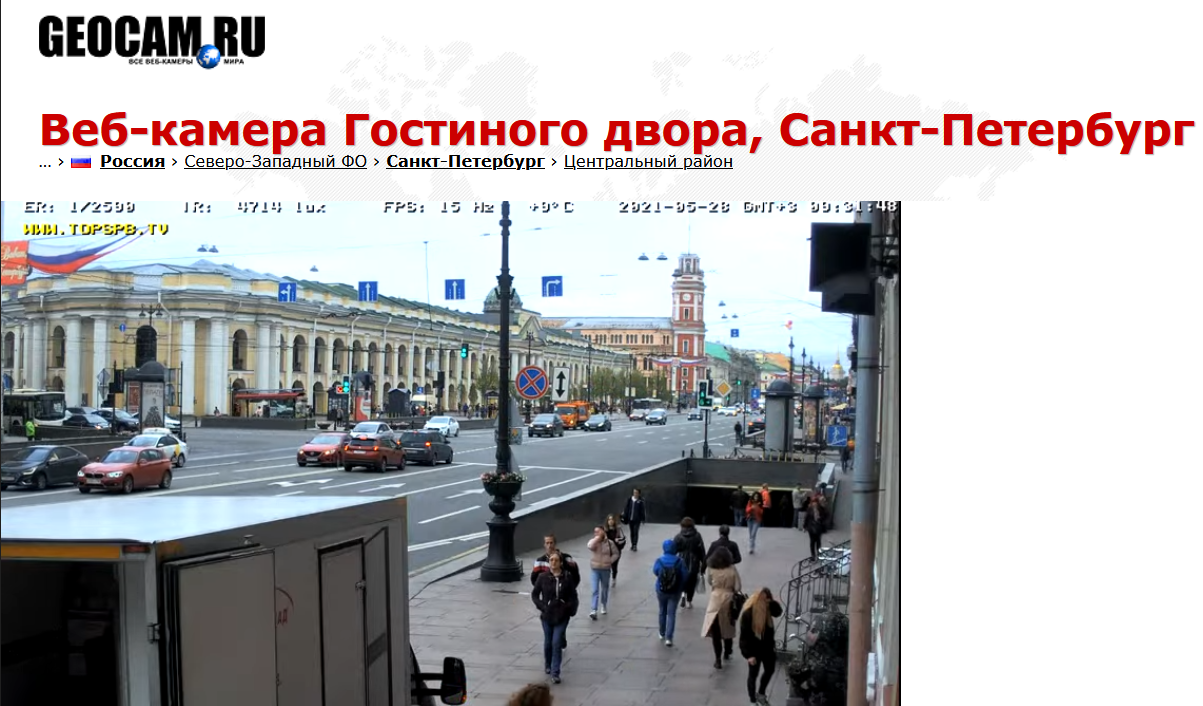
Рассмотри один из способов организации данного сценария. Клиентом является пользователь, который участвует в видеосвязи. Медиа поток клиента отправляется на сервер, в роли которого может выступать как другой клиент, так и специализированная под такой тип задач ЭВМ, после чего медиа поток рассылается другим клиентам конференции.

1.1.3 Трансляции с веб-камер

Не покидая собственной квартиры, можно побывать на известных достопримечательностях, в далеких городах или даже посмотреть на нашу планету с борта МКС. Все это доступно благодаря возможности просматривать трансляции с веб-камер, хотя и не ограничивается только развлечением: в последнее время все большую актуальность обретает возможность наблюдения за заседанием государственной думы или, например, наблюдение за избирательными участками во время выборов, что помогает обнаружить нарушения.

Пример такого сценария приведен на рисунке 3.

В роли сервера в данном случае выступает веб-камера, доступ к видео у которой будет просить клиент. Клиентом может быть как конечный интернет-пользователь, так и другой сервер, который далее будет оперировать данным потоком неким образом.



1. Сценарий доставки видео «трансляции с веб-камер» на примере сервиса geocam.ru

1.1.4 Стриминг

Благодаря современным технологиям любому пользователю глобальной сети доступна возможность создания собственной трансляции при наличии соответствующего оборудования. Более того, ощутимого числа людей эта возможность стала основным источником дохода: зритель может материально поддержать стримера либо прямым пожертвованием, либо подпиской на данного стримера, если стриминговая платформа поддерживает монетизацию данной функции, либо самым обычным просмотром рекламы. Стоит отметить, что основное количество людей собраны вокруг киберспортивного контента, который представляет из себя трансляцию игровой деятельности.

Пример сценария приведен на рисунке 4.



1. Сценарий доставки видео «стриминг» на примере видео кадра, взятого с платформы Twitch у пользователя yogscast

Несмотря на то, что в данном сценарии так же генерируется медиа-поток в реальном времени, как и в предыдущем сценарии, сервером в данном случае является специализированное ПО/ЭВМ, на который публикуется генерируемый медиа поток. Так же клиентом сервера считается пользователь, который запрашивает медиа поток стримера.

Еще одно отличие состоит в том, что веб-камеры могут поставлять медиа поток одного типа — захватываемое данной камерой видео, а в данном сценарии медиа поток может быть сгенерирован как камерой стримера, так и специализированным ПО, которое может комбинировать различные видео элементы, видео захвата экрана, видео с камеры стримера, посторонние изображения и тому подобное.

1.2 Существующие решения

1.2.1 Определение области рассматриваемых решений

Как уже упоминалось ранее, существует огромное количество решений, которые осуществляют вышеперечисленные сценарии. Это могут быть серверы, оперирующие как статичными файлами, так и динамическим контентом пользователя.

В рамках поставленной цели имеет смысл рассматривать только те решения, которые позволяют генерировать медиа поток. Категорий таких решений 2:

* решения видеосвязи и конференция, которые в основном генерируют медиа поток основываясь на камерах (usb-камерах, web-камерах, а также камерах захвата экрана);
* программные энкодеры медиа потока, которые могут генерировать медиа поток основываясь не только на камерах, но и других внутренних и внешних источниках (например, внешний rtmp сервер или внутреннее статичное видео).

Для ясности и дальнейшего понимания стоит определиться с понятием программные энкодеры. Под данным понятием подразумевается класс программного обеспечения, которое способно генерировать медиа поток из других исходных данных и затем публиковать его на удаленные медиа сервера.

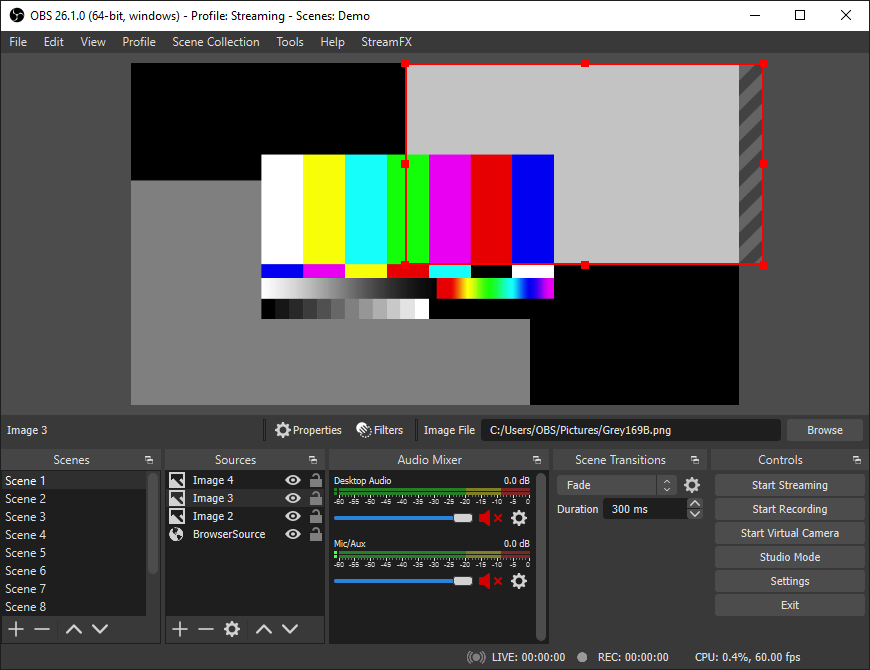
1.2.2 Программные энкодеры

1.2.2.1 Open Broadcast Software

Наиболее популярным решением является Open Broadcaster Software (OBS). OBS – это бесплатная программа с открытым исходным кодом для записи видео и потокового вещания. Доступна на многих операционных системах, среди которых присутствуют Windows, Linux и Mac. Среди конкурентов решения выделяют:

* возможность стримить на любые стриминговые платформы,
* большой выбор плагинов, число которых постоянно растет,
* локализация для 40 языков.

Пример решения приведен на рисунке 5.



1. Решение по реализацию сценария «стриминг» на примере ПО OBS

Среди минусов можно выделить отсутствие возможности одновременно стримить на несколько платформ из коробки, а также отсутствие возможности смены битрейта на лету.

В целом, данное решение позволяет собирать локальные элементы и комбинировать их в единый медиа поток и публиковать его на rtmp сервер из коробки, но может быть расширено плагинами. Например, можно вещать по другим транспортным протоколам или подключать внешние источники данных. Стоит упомянуть плагин [], который способен получать данные из видеоконференций, основанных на стандарте WebRTC реализованных в медиа сервере Janus. С подобными плагинами возможность создания виртуального центра управления медиа потоков становится реальной, хотя и требует большого количества накладных ресурсов и сторонних серверов.

Стоит также отметить наличие сцен и возможности переключаться между ними не прерывая поток. Это дает возможность создания пред настроенных сцен для решения разного рода задач. Например, в процессе генерации потока видеосвязи нескольких людей можно использовать разные сцены, отвечающие за композицию разного количества входящих потоков. Возможность динамической смены сцен и вариантов композиции отсутствует, поэтому переключаться между ними придется вручную.

В случае, если мы захотим создать виртуальный центр управления медиа потоков, в котором будет присутствовать возможность генерации сцен удаленно или посредством нескольких людей, нам придется создавать поток на каждый необходимый сценарий по отдельности и публиковать на промежуточный сервер. Таким образом будет возможность выбора потока, который генерируется отдельным человеком, который размещен на отдельном от остальных потоков адресе, с возможностью смены данного потока другим у любой момент времени.

Данный подход, очевидно, накладывает накладные расходы. Так, например, чтобы иметь возможность выбирать из 3 сцен, нам придется кодировать и декодировать данные как минимум 4 раза, что в 4 раза больше в сравнении с решением, которое предоставило бы возможность не использовать отдельные сервера, а динамически переключаться, используя только один набор входных данных.

1.2.2.2 XSplit

XSplit Broadcaster и XSplit Gamecaster – ПО, предназначенное для трансляции видеопотоков, генерируемых преимущественно посредством захвата окна компьютерной игры. Данное ПО заточено под геймеров и позволяет начать трансляцию буквально одной кнопкой вне зависимости от вида ЭВМ – будь то персональный компьютер или игровая консоль. Для изменения битрейта или других настроек останавливать данное решение не требуется. Из минусов можно отметить отсутствие использования данного по на операционных системах кроме Windows, а также малое количество плагинов.

Композиция сцены также присутствует, но обладает меньшей гибкостью из-за меньшего количества плагинов.

1.2.3 Программы для видеоконференции

1.2.3.1 Google Meet

Google Meet – решение, которое предоставляет возможность участия в аудио- и видеоконференциях, а также предоставляет возможность мгновенного обмена сообщениями. Решение основано на стандарте WebRTC и не требует стороннего ПО. Примечательно тем, что использует скрытые для широкого пользования возможности WebRTC, являясь своего рода входной точкой для новых возможностей данного стандарта в широкое использование. Так, например, именно в Google Meets впервые были использованы возможности по захвату экрана при помощи функций, описанных стандартом WebRTC.

1.2.3.2 Zoom

Zoom, как и многие другие решения, предоставляет возможность участия в видео конференциях, причем приложение обеспечивает возможности использования HD видеопотока и подключения до 100 участников к одной конференции в бесплатной версии. Среди минусов можно выделить обширное количество уязвимостей безопасностей (например, отсутствие сквозного шифрования), а также ограничение длительности конференции в 40 минут для бесплатной версии.

Данное решение интересно в рамках данной работы тем, что здесь присутствует минимальные возможности управления сценой, хоть таковые возможности и локальны. В Zoom возможно перемещать видео блоки с участниками по своему усмотрению, а также возможно управлять входным медиа потоком (включать и выключать аудио и видео), а также наличием залов, которые помогают разбивать участников на изолированные группы.

Касательно композиции, в Zoom существуют фильтры видео, которые могут изменять фон позади участника или накладывать объекты поверх участника (например, шляпа в качестве головного убора или усы), хотя практической пользы от этого в рамках данной работы нет.

1.3 Выводы

После анализа сценариев доставки видео контента была составлена таблица 1, приведенная ниже, которая отображает ключевые особенности каждого сценария. По результатам анализа были выделены интересующие в рамках данной работы сценарии:

* видеосвязь, конференции,
* стриминг.

Данные сценарии были выделены так как только в них присутствует генерация потока клиента, во время которой возможно применение композиции сцены.

1. результаты анализа сценариев поставки меди потока

|  |  |  |  |
| --- | --- | --- | --- |
|  | Клиент | Сервер | Генерация потока клиентом |
| Просмотр видео в интернете | Пользователь сети интернет | Специализированное ПО/ЭВМ | Нет |
| Видеосвязь, конференции | Пользователь сети интернет | Пользователь сети интернет Специализированное ПО/ЭВМ | Да |
| Трансляции с веб-камер | Пользователь сети интернет  Другой сервер | Веб-камера | Нет |
| Стриминг | Пользователь сети интернет | Специализированное ПО/ЭВМ | Да |

После анализа решений, использующих интересующие нас сценарии, была составлена таблица 2, приведенная ниже, в которой отображены ключевые особенности каждого решения.

Среди рассмотренных решений наиболее близким к обеспечению поставленной цели является OBS с использованием плагинов. Однако данное решение не является оптимальным, так как требует внешних серверов либо для подключения к ним и сбора данных, либо для публикации одной из возможных композиций для обеспечения возможности удаленного администрирования каждой из них.

Можно отметить, что использовать композицию сцен в рамках видеоконференций возможно, при условии композиции сцены программным энкодером и генерации виртуальной камеры, на основе которой будет генерироваться поток для видеосвязи.

1. анализ решений, использующих интересующие нас сценарии

|  |  |  |  |
| --- | --- | --- | --- |
|  | Тип сценария | Особенности | Возможности композиции сцен |
| OBS | стриминг | большое количество плагинов | присутствуют |
| XSplit | стриминг | ориентированность на геймеров | присутствуют |
| Google Meets | видеосвязь | ранний доступ к WebRTC функциям | отсутствуют |
| Zoom | видеосвязь | до 100 (500 в платной версси) участников | несущественные |

По результатам анализа существующих сценариев и сцен, было выявлено отсутствие возможности динамической композиции транслируемого видеопотока под управлением нескольких пользователей, что подтверждает целесообразность данной работы.

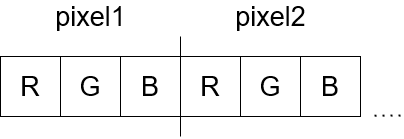
1. СРЕДСТВА ОРГАНИЗАЦИИ ПЕРЕДАЧИ ПОТОКОВОГО ВИДЕО

2.1 Средства композиции сцены

2.1.1 Вручную

В процессе обработки видео потока, получив сырые, или как их еще называют, декодированные данные, можно получить доступ к данным о пикселях на кадре посредством вызовов соответствующих методов используемого API.

Существует несколько вариантов расположения битов и их значения для одного пикселя. Наиболее распространенные варианты это RGB и I420. На рисунке 6 приведен приме организации порядка байтов в формате RGB



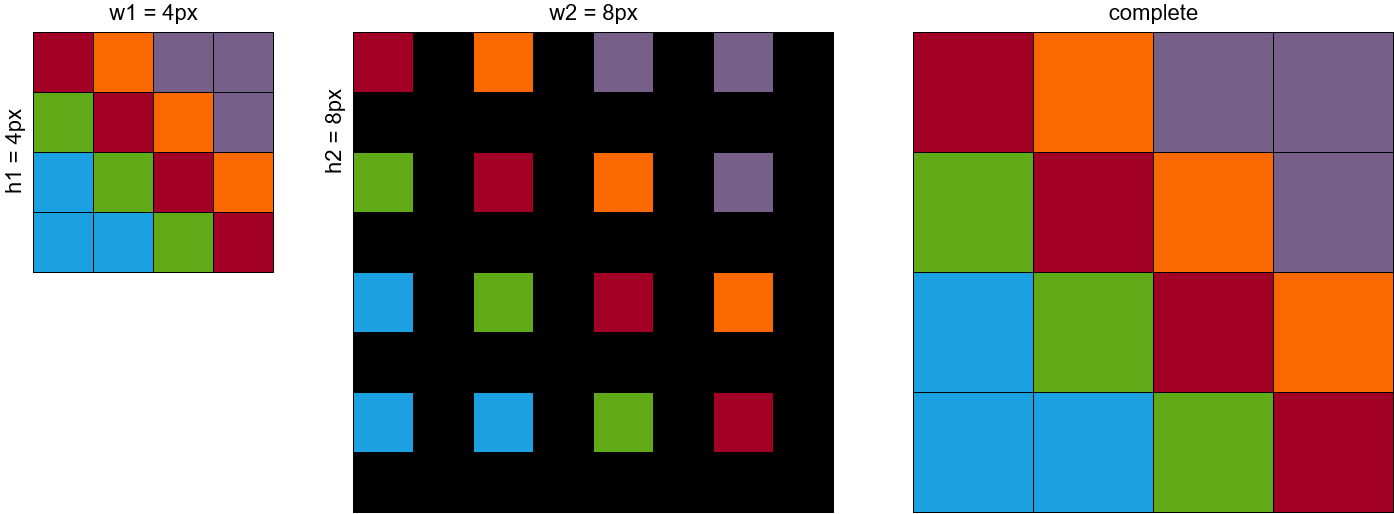
1. пример порядка байтов в формате RGB

Зная то, как расположены биты для соответствующих пикселей, и что они означают, можно с легкостью манипулировать ими:

* масштабировать,
* изменять размер добавляя или обрезая граничные пиксели,
* заменять пиксели другими с целью создания наложения видео компонентов друг на друга

Для наглядности, рассмотрим один из методом манипуляции над данным кадром. В качестве метода манипулирования изображением был выбран метод масштабирования «метод ближайшего соседа».

На рисунке 6 приведен пример работы метода масштабирования. Из изначального кадра 4 на 4 пикселя, мы получаем новый размером 8 на 8. Черным цветом помечены новые пиксели, которые появятся в кадре.



1. иллюстрация работы метода ближайших соседей

Ниже приведен пример кода 1, в котором описан процесс масштабирования кадра методом ближайшего соседа.

1. пример метода масштабирования кадра методом ближайшего соседа на языке Java

public int[] resizePixels(byte[] pixels,int w1,int h1,int w2,int h2) {

    byte [] temp = new int[w2\*h2] ;

    double x\_ratio = w1/(double)w2, y\_ratio = h1/(double)h2;

    double px, py;

    for (int i=0;i<h2;i++) {

        for (int j=0;j<w2;j++) {

            px = Math.floor(j\*x\_ratio) ;

            py = Math.floor(i\*y\_ratio) ;

            temp[(i\*w2)+j] = pixels[(int)((py\*w1)+px)] ;

        }

    }

    return temp ;

}

2.1.2 Использование библиотек

В целом, подход в операциях изменения размера и замены пикселей кадра схожи с подходом из примера указанного выше. Единственное отличие заключается в том, что нет необходимости самим работать с пикселями, ведь ошибиться в способе организации байтов очень легко, а понять в чем ошибка бывает достаточно сложно.

2.2 Методы доставки видео контента

2.2.1 Протоколы управления потоком, передачи и описания данных

В общем случае протоколы управления потоком описывают то, как два участника видеосвязи будут обмениваться данными с целью получения данных. В рамках обмена медиа контента подразумевается имплементация хотя бы команды Play, но иногда встречаются и расширенные варианты: пауза, стоп, продолжение и другие. Примерами таких протоколов являются RTSP, RTMP.

Протоколы передачи или транспортные протоколы описывают то, как сжатое видео упаковывается в так называемый контейнер, который представляет из себя байтовое отображение данных, который далее будет отправлен от одного участника другому. Примерами таких протоколов являются RTMP, RTP.

Можно заметить, что RTMP выступает в роли двух видов протоколов. Альтернативой ему является связка двух других – RTP + RTSP.

Протокол описания данных предназначен для описания сессии и потоковых данных, которые передаются в ее контексте. Примером такого протокола является SDP.

Сессия SDP может реализовывать потоков данных, среди которых могут быть аудио- и видеоданные, а также данные управления и поточные приложения.

В общем случае SDP сообщение содержит в себе

* адрес места назначения,
* номера UDP портов отправителя и получателя,
* описание медиа форматов (кодеки),
* описание атрибутов сессии (например, дополнительные атрибуты кодека или режим приема-передачи),
* в случае, если сообщение описывает не прямую трансляцию может содержать время старта и остановки.

2.2.2 WebRTC

WebRTC – это стандарт, который описывает тот, как необходимо организовывать передачу потоковых аудио- и видеоданных, а также данных произвольного типа (например текст) между браузерами или другими поддерживающими его приложениями в режиме реального времени без обязательного использования посредников. Ключевой особенностью является отсутствие необходимости в стороннем ПО: набор стандартов, которые включает в себя WebRTC, позволяет проводить пиринговые конференции и обмен данными без необходимости пользователю устанавливать плагины.

Эта особенность возможна благодаря набору функций Adapter.js, которые обеспечивают бесшовную стыковку различных браузеров и приложений.

WebRTC работает поверх протоколов RTP/RTCP.

Ключевым звеном в стандарте WebRTC является PeerConnection. Именно этот API позволяет устанавливать пиринговое соединение между браузерами. Настроить общение между оконечными пользователями довольно просто. Необходимо лишь передать сообщение о намеренности одного пользователя начать общаться с другим. Реализуется это в большинстве случаем при помощи простых сторонних серверов, часто называемых сигнальными, задача которых состоит лишь в предоставлении возможности пользователям отправлять и получать сообщения.

На рисунке 8 представлена вариация того, как может выглядеть общение между двумя пользователями. Упрощенная (без использования STUN и TURN серверов) же последовательность действий для инициализации обмена медиа данными такова:

1. PeerA хочет инициировать общение с пользователем PeerB,
2. PeerA посылает SDP сообщение Offer, то есть описание включающие всю информацию для соединения, о предлагаемой конфигурации вызывающего абонента,
3. PeerB в может согласиться или отказаться инициировать сообщение,
4. В случае, если PeerB хочет инициировать сообщение, он посылает SDP сообщение Answer, аналогичное сообщению Offer.
5. После этого пользователи начинают обмениваться так называемыми IceCandidate сообщениями, в которых заключается информация о сетевом соединении (Internet Connectivity Establishment) которая может быть использована для установки соединения. В данном сообщении описываются протокол и путь сообщения.



1. Пример общения между двумя пользователями WebRTC

2.2.3 Технологии адаптивной потоковой передачи данных передачи данных

Основной концепт данных технологий заключается в разбиении контента на последовательность небольших файловых сегментов, каждый из которых содержит небольшой отрывок содержимого. Данные сегменты могут описывать разными способами, например при помощи m3u8-плейлистов или MPD-манифеста XML. Основные представители данной категории это HLS и MPEG-DASH.

HLS строится поверх HTTP по которому передаются MPEG-TS контейнеры, в которых содержится сжатые закодированные данные, с использованием H264-видео и AAC- или MP3-аудио. Был разработан в компании Apple, в связи с чем изначально работал только в браузер Safari под управлением операционной системы семейства Mac. Это дало сложности в использовании стандарта сторонними производителями. Так, например, были реализованы собственные способы передачи разных аудиодорожек, из-за чего теперь приходится отдавать разные форматы плейлистов для разных плееров.

MPEG-DASH обычно строится с использованием контейнеров MP4 и H264/H265-видео и AAC-аудио внутри него или с контейнером WEBM и VP/VP9 видео внутри, хотя строгой привязки к технологиям нет. MPEG-DASH хорош тем, что в большинстве браузеров работает нативно, без использования стороннего ПО.

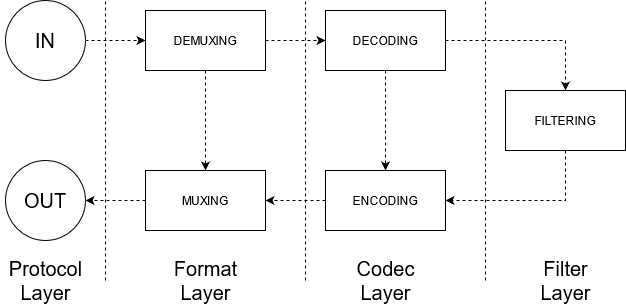
2.3 Библиотеки кодеков для транскодирования

2.3.1 libav (FFmpeg)

С помощью FFmpeg можно выполнять большое количество задач по обработке видео: кодирование, декодирование, транскодирование, мультиплексирование и демультиплексирование. Использование данной программы очень просто — FFmpeg представляет из себя консольное приложение, а процесс работы над видео задается при помощи флагов и параметров.

Библиотека libav представляет кроссплатформенные инструменты и API для конвертирования, манипуляции и передачи потокового медиа в широком спектре форматов по распространенным протоколам для программного кода. Библиотека libav является ответвлением программы FFmpeg и разрабатывается отдельно.

В общем случае процесс работы с видео сводится к общему шаблону, который представлен на рисунке 9.



1. Шаблон рабочего процесса работы с видео в libav

Protocol Layer оперирует но не ограничивается протоколами HTTP, RTMP. Вместо оперирования протоколами допустимо оперирование файлами или сырыми данными из приложения.

Format Layer оперирует но не ограничивается форматами видео потока MKV, MP4.

Codec Layer оперирует но не ограничивается видами кодировки AV1, VP9, H264.

Минусом libav можно отметить, что может помечать свои старые API как устаревшие, что может вести к модифицированию кода.

2.3.2 vlclib

vlclib – это главное ядро и интерфейс на котором основан фреймворк VLC Media Player. Под libvlc написано большое количество плагинов, которые могут быть загружены во время выполнения кода.

Данная библиотека поддерживает все распространенные форматы файлов, а также все кодеки и протоколы медиа вещания. Имеет API для всех популярных языков программирования.

Тем не менее, шаблон процесс работы с видео не отличается от шаблона libav, и предоставляет ту же функциональность.

2.3.3 gstreamer

Gstreamer – мультимедийный фреймворк, который использует собственную систему объектов GObject, является основой для многих мультимедийных приложений от видеоредакторов до потоковых серверов и медиа проигрывателей.

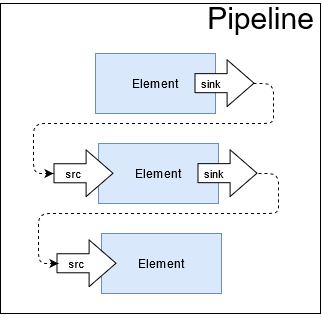
Данный фреймворк позволяет создавать так называемые pipeline-ы (далее конвейер), которые позволяют гибко работать с видео, изменять его, передавать и принимать. Внутри трубы содержатся GObject, каждый из которых может изменять поступающие в него данные и передавать на обработку следующему элементу. Основной смысл конвейера заключен в инкапсуляции работы на данными. Все что интересует пользователя в момент работы с конвейером – его состав и порядок элементов внутри него.

Элементы GObject являются программным кодом, который обрабатывает данные неким образом. Существует огромное количество реализаций данного класса под любые задачи: будь то посылание сырых кадров из приложения в конвейер, разветвление данных или отправка данных на удаленный сервер. Если есть какая-то стандартная задача работы с видео данными под нее скорее всего есть готовое решение в виде элемента GObject.

На рисунке 10 представлен пример простого конвейера. Как видно на рисунке у каждого элемента имеется два шлюза:

* src – входная точка данных в элемент
* sink – выходная точка данных из элемента

Можно заметить, что у конвейера нет шлюзов вовсе, а у оконечных элементов отсутствует по одному. Это обусловлено назначением этих шлюзов, а именно передачей данных от одного элемента к другому.



1. Простой пример конвейера в фреймворке Gstreamer

2.4 Выводы

По результатам анализа средств организации передачи потокового видео был определен набор инструментов, которые будут задействованы в реализации решения. В качестве ядра программы решено использовать фреймворк Gstreamer из-за его универсальности и гибкости. Так как решение, разрабатываемое в рамках работы, планируется быть максимально гибким и иметь возможность собирать медиа потоки из всевозможных источников, среди которых могут быть и источники видеоконференцсвязи, конкуренты фреймворка Gstreamer отпадают. Как упоминалось в ходе анализа, во фреймворке Gstreamer существуют реализации элементов на любую стандартную задачу. К сожалению, реализация конференцсвязи не является стандартной проблемой. Среди доступных плагинов не было найдено такого, который позволил бы получать медиа поток от каждого нового участника видеосвязи, в связи с чем возникает необходимость в удобном решении видеосвязи, у которого есть простой и надежный API, который желательно был бы широко используемым и унифицируемым. Среди всех проанализированных решений только одно подходит под эти условия – WebRTC стандарт. Таким образом были определены используемые решения для данной работы, основанное на комбинации двух технологий:

* Gstreamer – для удобного манипулирования видео,
* WebRTC – для поддержки видеоконференцсвязи.

1. РАЗРАБОТКА РЕШЕНИЯ ДЛЯ СОВМЕСТНОГО ВИДЕО СТРИМИНГА

3.1 Описание архитектуры

3.2 Метод использования WebRTC – RTMP

3.3 Метод использования RTMP – WebRTC

1. РЕЗУЛЬТАТЫ РАЗРАБОТКИ

ЗАКЛЮЧЕНИЕ

СПИСОК ЛИТЕРАТУРЫ

<https://github.com/CoSMoSoftware/OBS-studio-webrtc>

Глоссарий

Стриминговая платформа

Виртуальный центр управления медиа потоков

API