УДК 004.522

А.И. САВЕЛЬЕВ

ОПТИМИЗАЦИЯ АЛГОРИТМОВ РАСПРЕДЕЛЕНИЯ ПОТОКОВ МУЛЬТИМЕДИЙНЫХ ДАННЫХ МЕЖДУ СЕРВЕРОМ И КЛИЕНТОМ В ПРИЛОЖЕНИЯХ ВИДЕОКОНФЕРЕНЦСВЯЗИ

Савельев А.И. **Оптимизация алгоритмов распределения потоков мультимедийных данных между сервером и клиентом в приложениях видеоконференцсвязи.**

Аннотация. Оптимизация работы с мультимедийными ресурсами с целью сокращения передаваемого объема данных между пользователями является одной из проблем приложений видеоконференцсвязи. В статье описаны алгоритмы и программные средства, позволившие провести оптимизацию разработанного кроссплатформенного приложения видеоконференцсвязи. Основными этапами работы средств видеоконференцсвязи являются: создание и удаление аудио- и видеопотоков данных, их передача от сервера к клиенту и обратно, создание цепочек потоков и их поиск на сервере. Вышеперечисленные этапы присутствуют в любом приложении видеоконференцсвязи и требуют оптимизации в связи с присутствием в них главных процессов и сложностью архитектуры самого приложения. Поэтому в ходе работы было выполнено упрощение клиентской части приложения и реорганизация структуры серверной части приложения. В режиме приема-передачи данных разработанное приложение после проведенной оптимизации по сравнению с программой «Skype» стало потреблять почти в 10 раз меньше оперативной памяти и в 2 раза меньше процессорного времени.

Ключевые слова: видеоконференцсвязь, кроссплатформенные приложения, многомодальные интерфейсы, мультимедийные потоки, клиент-серверная архитектура, оптимизация программных ресурсов.

Saveliev A.I. **Optimization algorithms distribution streams of multimedia data between server and client in videoconferencing application.**

Abstract. Optimization of working with multimedia resources on purpose reduce transmitted amount of data between users is one of the problems videoconferencing applications. The paper describes algorithms and software, that allows to made optimization of cross-platform videoconferencing application. The main stages of videoconferencing application are: creation and deletion of audio and video streams of data, their transmission from the server to the client and back, creation chains of streams and their search on the server. The above stages are presents in any videoconferencing application and they have to be optimized due to the presence that they contain key processes and the complexity of the architecture of the application. Therefore, in the course of work has been done simplification of the client part of the application and the reorganization of the structure of the server-side application. In transmit-receive mode of data developed application after optimization, compared with the program "Skype" was consuming almost 10 times less RAM and 2 times less CPU.

Keywords: videoconferencing, cross-platform applications, multimodal interfaces, multimedia streams, client-server architecture, optimizing applications.

1. Введение. На сегодняшний день существует множество различных приложений видеоконференцсвязи [4, 10], некоторые из них малоизвестны, другие являются крупными коммерческими проектами,

широко используемыми в различных сферах жизнедеятельности человека. Потребность приложений видеоконференцсвязи высока и это связанно с одним из важнейших аспектов жизни человека – обменом информацией. Компании, разрабатывающие приложения видеоконференцсвязи, используют собственные подходы при их проектировании и разработке, поэтому готовое приложение обладает собственными достоинствами и недостатками. Особенности каждого приложения определяют возможность его использования на различных платформах и устойчивость к различному виду нагрузок, связанных с передачей, обработкой и выводом данных, предназначенных для пользователей.

Увеличение числа участников одна из главных проблем всех приложений видеоконференцсвязи. С ростом количества участников, возрастает нагрузка на само приложение: увеличивается количество входящих и исходящих потоков данных и возрастает число обрабатываемых данных, выводимых устройством. Приложения видеоконференцсвязи, работающие на настольных компьютерах, в большинстве случаев располагают необходимыми ресурсами для обработки данных, но при большом количестве входящих и исходящих потоков возможны проблемы из-за перегрузки центрального процессора и оперативной памяти устройства. На мобильных устройствах ситуацию усугубляет отсутствие ресурсов необходимых для обработки больших объемов данных и маленькие дисплеи, не способные корректно отображать более четырех участников одновременно. Частичное решение данных проблем возможно с помощью разработки и создания хорошо оптимизированной архитектуры приложения.

Все приложения видеоконференцсвязи по своей архитектуре, делятся на два типа: клиент-серверные [9, 12, 16] и пиринговые (peer-to-peer) [11]. Данные архитектуры сильно различаются по способам распределения нагрузки на устройства и системой обмена данными. Сначала рассмотрим клиент-серверную архитектуру. В ее основе лежит сервер, который выполняет основные задачи: авторизация клиентов, обработка потоков данных и распределение потоков данных между клиентами. Клиентская часть такой архитектуры способна отображать, передавать серверу и принимать потоки данных. Таким образом, нагрузка на устройство конечного пользователя снижается за счет выполнения основных операций по обработке потоков данных на сервере. Также следует отметить перспективность применения автоматических средств анализа речи и других естественных модальностей, обеспечивающих сокращение объема передаваемых данных и возможность

построения речевых и многомодальных интерфейсов, для данного типа телекоммуникационных приложений [1, 3, 5].

Пиринговая (peer-to-peer) архитектура подразумевает обмен информацией напрямую между конечными пользователями. В случае приложений видеоконференцсвязи даже для данной архитектуры необходимо наличие сервера, служащего для авторизации и координации работы клиентов. Основные функции все же выполняются на устройстве конечного пользователя, поэтому вся нагрузка, связанная с обработкой и отображением данных приходится на устройство конечного пользователя.

Приведем примеры известных приложений видеоконференцсвязи с различными архитектурами. Skype (peer-to-peer архитектура) [2] – проприетарное программное обеспечение с закрытым кодом, обеспечивающее шифрованную голосовую связь и видеосвязь через интернет между компьютерами, а также платные услуги для звонков на мобильные и стационарные телефоны. Google Hangouts (клиент-серверная архитектура) – бесплатный сервис групповой видеосвязи в социальной сети Google+. Позволяет получать общий доступ к экрану, совместно использовать программу рисования, редактировать документы, проводить широковещательные вебинары.

Adobe Connect (клиент-серверная архитектура) – система, основанная на технологии Flash, позволяет проводить онлайн встречи, презентации с использованием Power Point, совместно работать с установленными приложениями. видеофайлы. Может поставляться как вебсервис, не требующий установки, или как самостоятельное приложение. В таблице 1 представлены основные преимущества и недостатки использования архитектур данных приложений.

Таблица 1. **Архитектуры приложений видеоконференцсвязи**

	Преимущества	**Недостатки**
Клиент-серверная	1.Отсутствие дублирования кода программы-сервера программами-клиентами. 2.Снижение требования к устройствам конечного пользователя. 3.Данные хранятся на сервере, что позволяет защитить их лучше, чем на клиенте. 4.Упрощенное управление полномочиями пользователей, подключенных к серверу.	1. Неработоспособность сервера приводит к остановке передачи любых данных. 2. Для крупных систем необходимо дорогостоящее серверное оборудование, способное обрабатывать большой объем данных.

	Преимущества	Недостатки
Пиринговая (peer-to-peer)	1. Передача данных происходит непосредственно между клиентами, даже при отключении сервера регистрации, данные продолжат передаваться. 2.Невысокие требования к серверному оборудованию.	1. Высокие требования к устройству конечного пользователя в связи с высокой нагрузкой на клиентскую часть приложения. 2. Проблемы работы на мобильных платформах при воспроизведении более двух аудио- и видеопотоков.

Вне зависимости от типа архитектуры основными задачами приложения видеоконференцсвязи являются обработка аудио- и видеопотоков и их передача от одного пользователя к другому [6]. В таблице 2 представлены ведущие приложения видеоконференцсвязи с различными архитектурами.

Таблица 2. **Примеры приложений видеоконференцсвязи**

	Название приложения		
	Skype	**Google Hangouts**	**Adobe Connect**
Достоинства	Существуют клиентские части как для персональных компьютеров, так и для мобильных платформ. Поддержка видеоконференций до 10 абонентов. Обеспечивается передача текстовых сообщений (чат) и передача файлов. Есть возможность вместо изображения с веб-камеры передавать изображение с экрана монитора.	Бесплатные сеансы видеосвязи одновременно до 10 участников, центральный видеопоток переключается автоматически в зависимости от того, кто в данный момент громче говорит. Используются открытые технологии, поэтому есть возможность интеграции со сторонними приложениями и веб-сервисами видеосвязи.	Кроссплатформенность, можно свободно выбирать, использовать ли простой телефон, IP-телефон, видеотелефон, микрофон с веб-камерой или профессиональную конференц-систему для трансляции аудио/видео. Существуют мобильные версии для платформ Android, iOS, BlackBerry PlayBook.
	Skype	**Google Hangouts**	**Adobe Connect**
Недостатки	Использование проприетарного протокола - несовместимого с открытыми стандартами (такими, как SIP или H.323), интенсивное использование антиотладочных приёмов и обфусцированного кода, возможны вирусные эпидемии из-за пиринговой архитектуры.	Требуется наличие аккаунта в социальной сети Google+, несмотря на то, что работает в браузере имеет мало поддерживаемых платформ и требует установки дополнительного плагина; адаптирован всего под одну мобильную платформу - Android.	Платное использование необходимого количества лицензий на решение Adobe Connect для компании или приобретение услуги хостинга. Очень высокая стоимость, доступная только крупным организациям.

Из вышеперечисленных приложений стоит отметить систему видеоконференцсвязи “Skype”. Архитектура “Skype” является проприетарной в отличие от клиент-серверной архитектуры “Google Hangouts” и “Adobe Connect”, что позволяет приложению экономить интернет трафик и уменьшить задержки при передаче информации от пользователя к пользователю. На данный момент “Skype” является самым развитым и востребованным приложением видеоконференцсвязи, поэтому сравнение результатов оптимизации мы будем проводить именно с этой программой.

2. Основные характеристики разработанного приложения видеоконференцсвязи. В данном исследовании было использовано приложение видеоконференцсвязи, описанное в работе [7]. На его основе была проведена оптимизация ресурсов, необходимых для обработки аудиовизуальных данных. Клиентская часть приложения[15] написана на языке программирования Action Script 3 [8], а серверная часть состоит из приложения, написанного на языке программирования JAVA [13] и сервера RED 5 [14]. За счет чего приложение обладает свойством кроссплатформенности, что позволяет ему конкурировать с аналогами.

В начале работы приложения необходимо ввести логин, пароль и адрес сервера, к которому будет подключен клиент. Возможность варьировать сервер, к которому происходит подключение, обеспечивает работу приложения в локальной сети, что дает преимущество использования приложения там, где нет доступа к интернету.

После успешного подключения к серверу и проверки логина и пароля происходит переключение в панель участников видеоконференцсвязи. Сверху области располагаются основные кнопки управления: «Participants» – осуществляет переход к панели отображения участников, «Camera» - осуществляет переход к панели управления камерами, «Friends» – выводит справа панель управления друзьями, где можно подключиться к пользователю, удалить его и найти нового, две последние кнопки – «Up» и «Down» позволяют прокручивать область отображения пользователей, тем самым обеспечивая вывод на дисплей участников, которые расположены за областью видимости. Ниже кнопок управления располагается область отображения участников, она способна отображать до восьми участников одновременно, чтобы увидеть остальных участников, нужно воспользоваться кнопками «Up» и «Down».

Последняя панель – это панель управления камерами. Она отображает кнопки навигации, рассмотренные выше и области управления

камерами и микрофонами. На каждой из областей управления камерами отображается видео, захватываемое камерами устройства и кнопки управления аудио- и видеопотоками данных: «On» включение камеры и микрофона, «Off» - отключение камеры и микрофона и «Publish» - публикация аудио-видео потока на сервер.

Далее рассмотрим основные преимущества приложения. Первым достоинством преимуществом является возможность работы приложения как на мобильных платформах: Android, Iphone, так и на платформах, преимущественно используемых на стационарных компьютерах и ноутбуках: Windows, Linux, причем без существенных потерь функционала и возможностей. Из-за различия между мобильными и настольными платформами, было разработано две версии клиентской части приложения. Различие между версиями приложения заключается в количестве отправляемых потоков с камер и микрофонов, подключенных к устройству. В мобильной версии всего одна камера и один микрофон могут отправлять поток на сервер, а в стационарной версии ограничения по количеству камер и микрофонов не накладываются, но рекомендуемое их количество: 10 камер и 10 микрофонов (зависит от возможностей стационарного компьютера или ноутбука). Второе преимущество приложения – это возможность проводить видеоконференции более, чем для двух человек с использованием стационарных компьютеров, ноутбуков и мобильных устройств, таких как планшетные компьютеры. Тем не менее, усложнение архитектуры клиент-серверного приложения, связанное с поддержкой кроссплатформенности и одновременным обслуживанием двух и более пользователей, приводит к существенному увеличению нагрузки на клиентские устройства.

Для решения этой проблемы была проведена оптимизация ресурсов клиент-серверных приложений видеоконференцсвязи. В результате был предложен специальный функционал сервера для упрощения обработки данных и увеличения скорости обмена данными между частями приложения. Это позволило одновременно снизить нагрузку на клиентскую часть приложения путем уменьшения количества обрабатываемых данных, а также уменьшить количество обращений от клиентских приложений к серверу и от сервера к клиентским приложениям для снижения нагрузки на серверную и клиентскую части. Далее рассмотрим особенности функционирования предложенного клиент-серверного приложения видеоконференцсвязи и результаты его экспериментальной проверки более подробно.

3. Описание функций взаимодействия клиентской и серверной частей приложения видеоконференцсвязи. Обработка и вывод аудио- и видеоданных – один из самых ресурсоемких процессов для любого устройства. В случае с приложениями видеоконференцсвязи процесс обработки и вывода усложняется тем, что аудио- и видеопотоки не хранятся на устройстве, а постоянно поступают с сервера, тем самым, увеличивая нагрузку на устройство, которое их принимает. Так как мобильные устройства располагают весьма ограниченными ресурсами для обработки и вывода данных, а стационарные компьютеры и ноутбуки обычно выполняют одновременно несколько приложений, то для уменьшения нагрузки на клиентскую часть приложения, комплекс функций, отвечающих за распределение и фильтрацию мультимедийных потоков, был перенесен на серверную часть приложения. Далее рассмотрим классы, необходимые для фильтрации и распределения аудио- и видеопотоков.

Для серверной части приложения был создан класс, отвечающий за распределение и фильтрацию потоков – «ControlConnections», который применяет два вспомогательных класса: «User» и «ClientStream». Класс «User» отвечает за создание пользователя, который является прототипом клиентской части приложения на сервере. Данный класс хранит внутри себя информацию: о пользователях, которые подключены к нему, информацию о потоках, которые поступают от других пользователей и информацию о собственных потоках, которые исходят из клиентского приложения. Экземпляр класса «ClientStream» создается при поступлении нового аудио- и видеопотока данных от клиента на сервер и содержит: имя пользователя, которому принадлежит данный поток, собственное имя и массив, в котором содержатся имена пользователей подключенных к данному потоку.

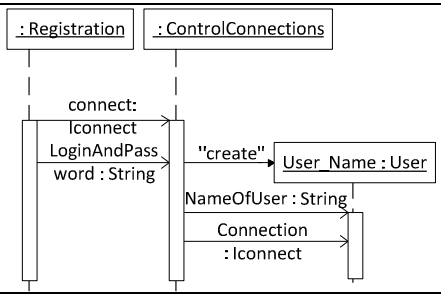
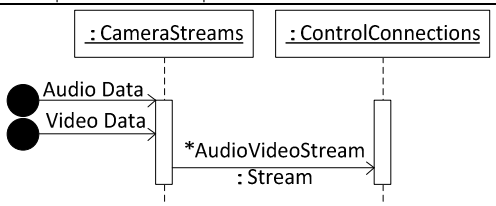
Созданием и удалением экземпляров классов «User» и «ClientStream» занимается класс «ControlConnections». Также класс «ControlConnections» содержит ряд функций, с помощью которых осуществляется: выбор нужных потоков, фильтрация потоков, рассылка потоков, удаление потоков, поиск пользователя, подключение к пользователю, удаление пользователя. Это далеко не весь ряд функций, которые содержатся в классе, но эти функции играют основную роль в оптимизации архитектуры клиент-серверного приложения.

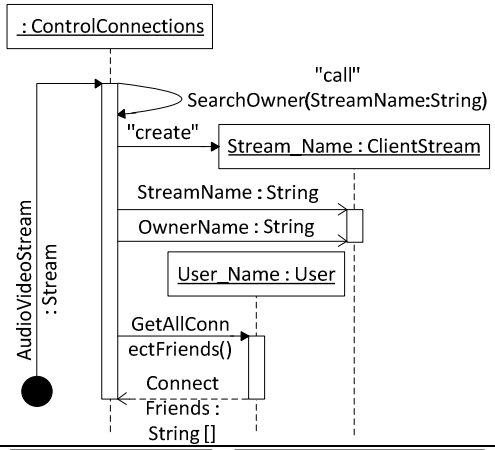
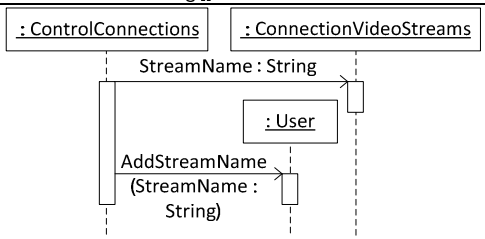
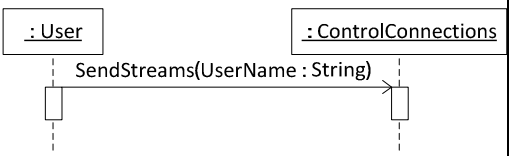
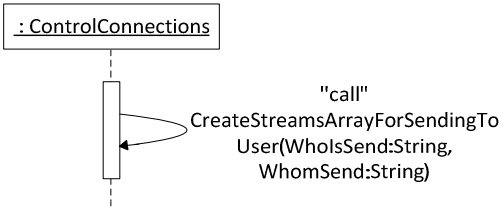
В клиентской части приложения всего три класса используются для взаимодействия с сервером: «Registration», «ConnectionVideoStreams», «CameraStreams». Класс «Registration» служит для проверки логина и пароля пользователя, а также для создания новой учетной

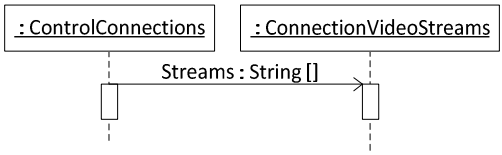
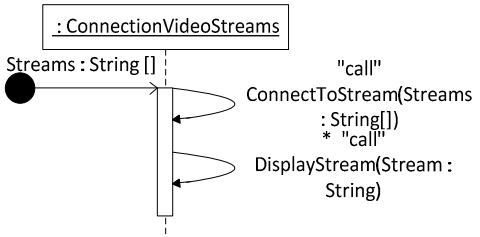
записи пользователя в базе данных сервера. Класс «ConnectionVideoStreams», отвечает за прием новых потоков аудио- и видеоданных и их вывод на дисплей устройства. Последний класс взаимодействия с серверной частью приложения «CameraStreams» формирует и отправляет аудио- и видеопотоки данных с камер и микрофонов устройства на сервер.

В таблице 3 представлены основные этапы взаимодействия серверной и клиентской частей приложения видеоконференцсвязи с использованием функций вышеперечисленных классов. Нужно отметить, что каждый этап может повторяться при условии, что приложение выполнило предыдущий этап успешно и связь между сервером и клиентом установлена.

Таблица 3. **Основные этапы взаимодействия клиентской и серверной частей приложения**

№ этапа	UML диаграмма взаимодействия	Описание
1		Происходит подключение клиентской части приложения к серверной части, создание нового экземпляра класса «User» и заполнение данными экземпляра класса «User».
2		Клиентская часть приложения отправляет аудио- и видеопотоки данных с камер и микрофонов на сервер.

№ этапа	UML диаграмма взаимодействия	Описание
3		Обработка потока данных в серверной части: создание нового экземпляра класса «Client-Sttream», занесение данных о потоке – имя потока, имя владельца потока, поиск других пользователей, подключенных к владельцу потока.
4		Рассылка и добавление имени потока всем пользователям, подключенным к владельцу потока.
5		Подключенные пользователи (их серверная часть) вызывают функцию, которая отправляет, подключенные к ним и их собственные потоки другому пользователю.
6		В серверной части происходит формирование массива имен потоков, на основе фильтрации этих имен, выявляются потоки, еще не содержащиеся у пользователя.

№ этапа	UML диаграмма взаимодействия	Описание
7		Отправка сформированного массива потоков пользователю.
8		Обработка и отображение пришедшего массива потоков.

На основе описанных в таблице 3 этапов работы клиентской и серверной частей приложения можно выделить два основных алгоритма взаимодействия: подключение потока данных от клиента и пересылка потока данных другим пользователям. Алгоритм обработки потоков при первом взаимодействии устройств, изображенный на рисунке 1, включает в себя 2 и 3 этапы таблицы 3. Действия, происходящие в ходе алгоритма, необходимы для учета всех данных о потоке в серверной части приложения, упрощения дальнейшего взаимодействия между пользователями и увеличения скорости обмена информацией между ними. Второй алгоритм взаимодействия, основанный на этапе 4 таблицы 3 и представленный на рисунке 2, отображает систему распространения потока. Все действия второго алгоритма происходят в серверной части приложения, тем самым освобождая клиентскую часть от нагрузки при нахождении еще не добавленных потоков. Другая положительная сторона этого алгоритма – это отсутствие обмена данными между клиентами для определения еще не добавленных потоков, все необходимые данные уже существуют на сервере и время выполнения алгоритма очень мало, благодаря отсутствию обмена данными с клиентами.

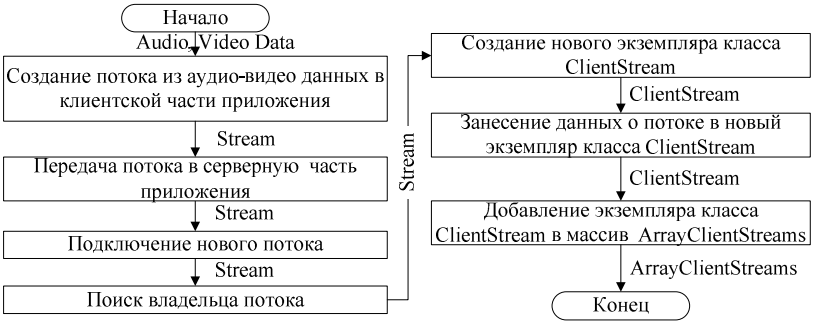


Рис. 1. Алгоритм создания нового потока

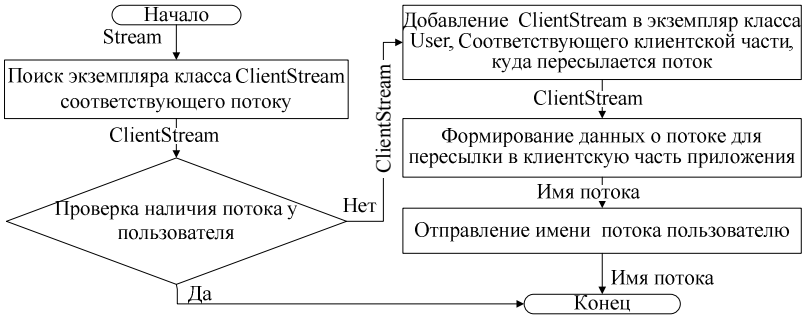


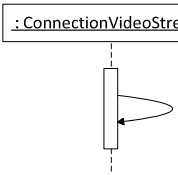
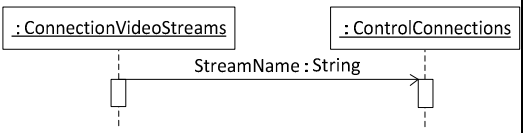
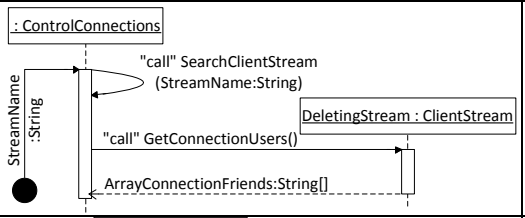
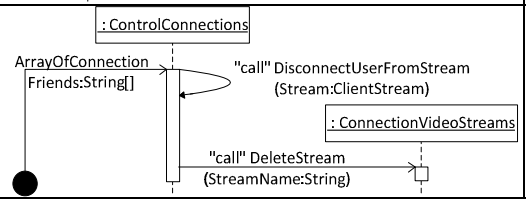
Рис. 2. Алгоритм распространения потока

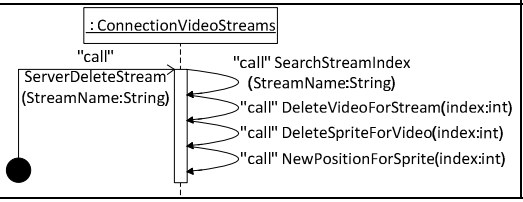
Алгоритмы формирования и распространения потоков создают полноценную видеоконференцсвязь для двух и более участников. Эти алгоритмы связаны тем, что каждый новый поток, который приходит от другого клиента, инициирует запуск функций для создания и пересылки уже имеющихся у клиента потоков, клиенту, который прислал новый поток. На основе двух этих алгоритмов осуществляется оптимизация работы клиент-серверного приложения, используя ограниченное количество действий, требующихся для формирования сложной структуры – цепочек взаимодействия клиентов и сервера. Преимущество построения таких цепочек взаимодействия заключается в том, что все пользователи, включающиеся в них, получают данные о других пользователях, уже находящихся внутри цепочки. Данные о других пользователях позволяют быстро сформировать запросы для обмена потоками, тем самым уменьшая время на обработку запроса и количе-

ство задействованных ресурсов. Так как весь функционал и объекты манипуляций этих цепочек содержатся на сервере, то это снижает время для обмена данными между клиентскими частями и значительно уменьшает нагрузку на них.

Цепочки взаимодействия клиентов и сервера позволяют так же быстро удалять потоки и пользователей, как и формировать их. Это способствует ускорению освобождения ресурсов устройств, необходимых для обработки других данных. Основные этапы по удалению потоков представлены в таблице 4.

Таблица 4. **Основные этапы удаления потоков от пользователей**

№ этапа	UML диаграмма взаимодействия	Описание
1		Клиентская часть приложения закрывает собственный поток аудио- и видеоданных, формируемый камерой и микрофоном.
2		Происходит пересылка имени закрываемого потока из клиентской части в серверную часть.
3		Серверная часть осуществляет поиск пользователей, подключенных к данному потоку.
4		Вызов функции удаления имени потока у подключенных пользователей.

№ этапа	UML диаграмма взаимодействия	Описание
5		Удаление потока и его отображения на клиентской части приложения.

Удаление пользователя происходит в два этапа: закрытие клиентской части приложения и удаление пользователя и всех его данных в серверной части при отключении соединения клиентской части от сервера. Во время удаления всех данных о пользователе в серверной части осуществляются этапы 3, 4 и 5 из таблицы 4 над всеми собственными потоками клиентской части приложения. Это позволяет отключить всех остальных участников конференции от уже несуществующих потоков и освободить ресурсы устройств еще до окончания сеанса видеоконференцсвязи.

Такая система взаимодействия позволяет оптимизировать работу как клиентской части, минимизировав ее функции по обработке данных, так и серверной, позволяя быстро и точно совершать манипуляции по распределению потоков данных среди клиентских частей, используя ограниченное количество действий.

4. Анализ потребляемых ресурсов устройствами в ходе видеоконференцсвязи. После оптимизации было проведено тестирование приложения и измерены объемы потребляемой оперативной памяти, загруженность центрального процессора устройства, на котором установлена клиентская часть приложения. Основная задача по оптимизации данного приложения – это повышение скорости обмена данными между клиентскими частями и повышение производительности их работы. График, представленный на рисунке 3, отображает объем потребляемой оперативной памяти на различных этапах работы приложения в зависимости от количества входящих и исходящих потоков данных. В работе клиентской части приложения можно выделить несколько ключевых этапов с потреблением разных объемов памяти: 1 этап – установление связи с сервером; 2 этап – вход в основное меню приложения; 3 этап – создание потока аудио- и видеоданных; 4 этап – публикация потока на сервер; 5 этап – удаление потока; 6 этап – прием потока; 7 этап – отключение принятого потока. Проанализировав график, изображенный на рисунке 3, можно отметить, что при удалении и

отключении потока – 5 и 7 этапы, происходит уменьшение потребляемой оперативной памяти. Надо также отметить, что уменьшение занимаемой оперативной памяти происходит в достаточно короткий момент времени – порядка несколько долей секунд. Благодаря быстрому освобождению оперативной памяти, увеличивается скорость обработки информации клиентским приложением, а также снижается нагрузка на устройство пользователя.

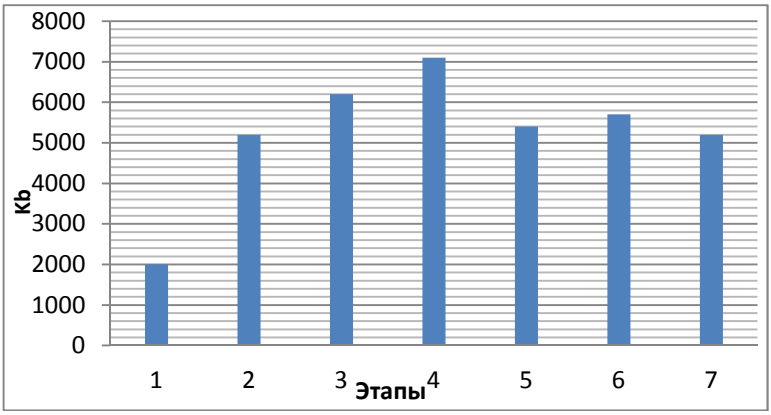


Рис. 3. Объем потребляемой оперативной памяти на различных этапах работы приложения.

Также была проанализирована зависимость занимаемой оперативной памяти от числа участников в видеоконференцсвязи. На основе стандартных методов Action Script 3 возможно определить объем занимаемой оперативной памяти в любой момент времени выполнения программы. В таблице 5 представлены зависимости объема занимаемой оперативной памяти на клиентском устройстве при разном количестве одновременно входящих и исходящих потоков. Данные параметры использовались для сравнения разработанного приложения видеоконференцсвязи с клиентской частью программы “Skype”. Полученные экспериментальные результаты приведены в таблице 6. Все этапы тестирования были проведены на одном и том же устройстве, работающем под управлением операционной системы “Windows 7”.

Таблица 5. **Нагрузка на оперативную память в зависимости от количества потоков**

Максимальный объем занимаемой оперативной памяти, кБ	Количество входящих потоков	Количество исходящих потоков
7 548	1	0
7 823	1	1
8 905	4	2
10 556	8	4
11 848	10	8

Для подтверждения целесообразности оптимизации было проведено сравнение с ведущим приложением видеоконференцсвязи – «Skype». Экспериментальные данные, приведенные в таблице 6, указывают на то, что оптимизация приложения значительно повлияла на количество занимаемых ресурсов.

Таблица 6. **Сравнительные характеристики клиентских частей приложения видеоконференцсвязи и программы “Skype”**

Режим работы клиентской части приложения	Объем потребляемой оперативной памяти клиентской частью приложения		Загруженность процессора клиентской части приложения	
	Видеоконференцсвязь	“Skype”	Видеоконференцсвязь	“Skype”
Режим проверки логина и пароля	~3 500 Kb	~10 000 Kb	1%	4-5%
Режим ожидания	~7 300 Kb	~30 000 Kb	0-1%	2-5%
Режим приема и отправки одного аудио- и видеопотока	~7 800 Kb	~60 000 Kb	5-6%	10-15%
Режим приема четырех и отправки одного аудио- и видеопотоков данных	~8 600 Kb	~80 000 Kb	7-9%	12-16%

Как видно из таблицы 6 в режиме приема-передачи данных, разработанное приложение после проведенной оптимизации по сравнению с программой “Skype” потребляет почти в 10 раз меньше оперативной памяти и в 2 раза меньше процессорного времени.

5. Заключение. Оптимизация разработанного приложения видеоконференцсвязи позволила значительно сократить ресурсы, прежде

всего, оперативной памяти, потребляемой клиентской части и упростить структуру данных, обрабатываемых серверной частью, что повысило скорость обмена аудио- и видеопотоками между клиентскими частями приложения и уменьшило количество обращений между клиентской и серверной частями приложения. Данная оптимизация увеличивает конкурентоспособность разработанного приложения видеоконференцсвязи за счет возможности его применения на мобильных устройствах с ограниченными ресурсами без значительных потерь функционала и скорости обмена данными. Дальнейшее развитие данного приложения видеоконференцсвязи будет направленно на улучшение качества передаваемых аудио- и видеопотоков и эргономики пользовательского интерфейса.

Литература

1. *Мещеряков Р.В., Бондаренко В.П.* Диалог как основа построения речевых систем // Кибернетика и системный анализ. 2008. № 2. С. 30-41.
2. *Потапова Р. К., Собакина А. Н., Маслов А. В.* Возможность идентификации говорящего по голосу в системе интернет-телефонии Skype // Вестник Московского Государственного Лингвистического Университета, № 13, 2013. С. 177-188.
3. *Ронжин А.Л.* Топологические особенности морфофонемного способа представления словаря для распознавания русской речи // Вестник компьютерных и информационных технологий, № 9, 2008, С. 12-19.
4. *Ронжин А.Л., Будков В.Ю.* Технологии поддержки гибридных е-совещаний на основе методов аудиовизуальной обработки // Вестник компьютерных и информационных технологий, № 4, 2011, С. 31-35.
5. *Ронжин А.Л., Карпов А.А.* Сравнение методов локализации пользователя многомодальной системы по его речи // Известия вузов. Приборостроение. Т. 51, № 11. 2008. С. 41-47.
6. *Ронжин Ал.Л., Будков В.Ю., Ронжин Ан.Л.* Формирование профиля пользователя на основе аудиовизуального анализа ситуации в интеллектуальном зале совещаний // Труды СПИИРАН. Вып. 23. 2012. С. 482-494.
7. *Савельев А.И.* Разработка приложения телеконференцсвязи для мобильной операционной системы Android // Завалишинские чтения: Сборник докладов / ГУАП, СПб, 2012. С. 176-184.
8. Adobe [Электронный ресурс] // http://help.adobe.com/ (Дата обращения 10.03.12).
9. *Bulut H., Fox G., Pallickara S., Uyar A., Wu W.* Integration of NaradaBrokering and Audio/Video Conferencing as a Web Service // In Proceedings of IASTED International Conference on Communications, Internet, and Information Technology, 2002. pp. 401-406.
10. *Erol B., Li Y.* An overview of technologies for e-meeting and e-lecture // In Proceedings of IEEE International Conference on Multimedia and Expo (ICME'2005), 2005. pp. 6-12.
11. *Fox D., Gannon S.-H., Ko S., Lee S., Pallickara M., Qiu X., Rao X., Uyar A., Wang M., Wu W.* Peer-to-Peer Grids // Chapter 18 of Grid Computing: Making the Global Infrastructure a Reality edited by Fran Berman, England, ISBN 0-470-85319-0, 2003, pp. 471-490.

12. *Fox G., Pallickara S.* The Narada Event Brokering System: Overview and Extensions // In Proceedings of International Conference on Parallel and Distributed Processing Techniques and Applications, 2002. pp. 353-359.
13. Java [Электронный ресурс] // http://www.java.com/ru/ (Дата обращения 27.03.13).
14. Red5 [Электронный ресурс] // http://www.red5.org/ (Дата обращения 14.11.12).
15. *Ronzhin A.L., Saveliev A.I., Budkov V.Yu.* Context-Aware Mobile Applications for Communication in Intelligent Environment // Springer-Verlag Berlin Heidelberg, S. Andreev et al. (Eds.): NEW2AN/ruSMART 2012, LNCS 7469, 2012. pp. 307-315.
16. *Uyar A., Pallickara S., Fox G.* Towards an Architecture for Audio/Video Conferencing in Distributed Brokering Systems // In Proceedings of International Conference on Communications in Computing, 2013, pp. 17-23.

Савельев Антон Игоревич — младший научный сотрудник лаборатории речевых и многомодальных интерфейсов, СПИИРАН. Область научных интересов: кроссплатформенные системы, видеоконференцсвязь, потоковая трансляция аудио- и видеоданных. Число научных публикаций — 6. saveliev@iias.spb.su; СПИИРАН, 14-я линия В.О., д. 39, г. Санкт-Петербург, 199178, РФ; р.т. +7(812)328-7081, факс +7(812)328-7081.

Saveliev Anton Igorevich — junior researcher, Laboratory of Speech and Multimodal Interfaces, SPIIRAS. Research interests: cross-platform systems, videoconferencing, audio and video streaming broadcast. The number of publications — 6. saveliev@iias.spb.su; SPIIRAS, 39, 14-th Line V.O., St. Petersburg, 199178, Russia; office phone +7(812)328-7081, fax +7(812)328-7081.

Поддержка исследований. Работа выполнена при финансовой поддержке РФФИ (проекты № 13-08-0741-а, № 13-07-00279-а), проекта Программы ОНИТ РАН «Научные основы создания гетерогенных телекоммуникационных и локационных систем и их элементной базы»

Рекомендовано лабораторией речевых и многомодальных интерфейсов, заведующий лабораторией Ронжин А.Л, д.т.н., доц.
Статья поступила в редакцию 09.10.2013.

РЕФЕРАТ

Савельев А.И. **Оптимизация алгоритмов распределения потоков мультимедийных данных между сервером и клиентом в приложениях видеоконференцсвязи.**

Оптимизация работы с мультимедийными ресурсами с целью сокращения передаваемого объема данных между пользователями является одной из проблем приложений видеоконференцсвязи. В статье описаны алгоритмы и программные средства, позволившие провести оптимизацию разработанного кроссплатформенного приложения видеоконференцсвязи. Основными этапами работы средств видеоконференцсвязи являются: создание и удаление аудио- и видеопотоков данных, их передача от сервера к клиенту и обратно, создание цепочек потоков и их поиск на сервере. Вышеперечисленные этапы присутствуют в любом приложении видеоконференцсвязи и требуют оптимизации в связи с присутствием в них главных процессов и сложностью архитектуры самого приложения. Поэтому в ходе работы было выполнено упрощение клиентской части приложения и реорганизация структуры серверной части приложения. В программной реализации это хорошо видно в упрощении всей структуры классов клиентской части, в серверной части приложения были созданы дополнительные классы, позволяющие производить различные действия над входящими и исходящими потоками мультимедийных данных и организовывать их взаимодействие для соединения клиентов. Такое изменение в программной структуре приложения позволило значительно снизить ресурсы, потребляемые клиентской частью приложения, сократить количество используемого трафика и уменьшить время на создание связи между двумя пользователями. В режиме приема-передачи данных разработанное приложение после проведенной оптимизации по сравнению с программой “Skype” стало потреблять почти в 10 раз меньше оперативной памяти и в 2 раза меньше процессорного времени.

SUMMARY

Saveliev A.I. **Optimization algorithms distribution streams of multimedia data between server and client in videoconferencing application.**

Optimization of working with multimedia resources on purpose reduce transmitted amount of data between users is one of the problems videoconferencing applications. The paper describes algorithms and software, that allows to made optimization of cross-platform videoconferencing application. The main stages of videoconferencing application are: creation and deletion of audio and video streams of data, their transmission from the server to the client and back, creation chains of streams and their search on the server. The above stages are presents in any videoconferencing application and they have to be optimized due to the presence that they contain key processes and the complexity of the architecture of the application. Therefore, in the course of work has been done simplification of the client part of the application and the reorganization of the structure of the server-side application. In a software implementation, this can be clearly seen in the simplification of the entire structure of classes client part, on the server part of the application have been developed additional classes to perform various actions on the incoming and outgoing streams of multimedia data and the organization their interaction for connections of clients. Such a change in the structure of the application significantly reduced the consumption of resources by the client part of the application, to reduce the amount of the traffic and reduce the time to create a connection between two users. In transmit-receive mode of data developed application after optimization, compared with the program "Skype" was consuming almost 10 times less RAM and 2 times less CPU.