# Проектирование и реализация системы удаленного управления

У

Изм.

Лист

№ докум.

Подп.

Дата

Лист

1

ДП 03.00.ПЗ

Разраб.

Веялкин И. А.

Пров.

Пацей Н. В.

Консульт.

Пацей Н. В.

Н. Контр.

Шиман Д. В.

Утв.

Урбанович П. П.

Проектирование и реализация системы удаленного управления

Лит.

Листов

15

БГТУ 64419902, 2013

Целью данного раздела является проектирование и описание схем построения архитектуры представленного в дипломном проекте мобильного приложения для удаленного управления компьютером. А так же обоснования выбранных подходов к организации данной распределенной вычислительной системы и описание процесса реализации представленного программного продукта по выбранному и спроектированному архитектурному решению и методу разработки.

В данном разделе так же приводится описание процесса разработки и особенностей различных алгоритмов, примененных в ходе создания описываемого программного продукта.

## Общая схема организации удаленного доступа к ПК

В рамках решения поставленной в данном дипломном проекте задачи для организации удаленного управления компьютером была выбрана архитектура построения приложения «Клиент-Сервер».

Представленное мобильное приложение для платформы Android представляет из себя клиентское программное обеспечение VNC системы и взаимодействует с серверной частью (VNC сервером). Клиент в данной архитектуре представляет из себя программу «тонкий-клиент», которое предоставляет пользователю интерфейс удаленного управления.

Для организации удаленного управления компьютером требуется VNC сервер, который обеспечивает связь клиента и операционной системы, установленной на удаленном ПК, поддерживая возложенные на него функции.

Клиент, установленный на мобильном устройстве под управлением ОС Android, и сервер, установленный на удаленном ПК под управлением ОС семейства MS Windows, либо MacOS X, либо Linux, взаимодействуют посредствам стека сетевых протоколов TCP/IP и используют для этого общую компьютерную сеть для передачи данных. Сетевое взаимодействие поддерживает адресацию, как на уровне доменных имен, так и сетевых ip адресов.

Для установления сессии удаленного управления на ПК запускается приложение-сервер VNC системы, задаются его сетевые настройки, а далее приложение переводится в состояние приема подключений от VNC клиентов. После перевода сервера в режим приема подключений пользователь запускает мобильный клиент распределенной VNC системы, установленный на устройство под управлением ОС Android, и производит настройку клиента для подключения к серверу VNC. Для этого пользователь вводит сетевой ip адрес или доменное имя компьютера, на котором запущен сервер, а также сетевой TCP порт, связанный с процессом сервера

на удаленной операционной системе.

Кроме настроек сетевого соединения на этапе подготовки мобильного клиента к установлению соединения с сервером пользователь имеет возможность задать некоторые настройки передачи и кодирования передаваемой информации о состоянии удаленного ПК, а именно формат кодирования передаваемого кадрового буфера.

Заключительной информацией, которую необходимо предоставить пользователем мобильному приложению для инициирования сессии удаленного управления является предоставления пароля для аутентификации клиента на сервере. Пароль – это любая комбинация символов, которую можно указать при задании настроек перед запуском сервера в режиме приема подключений от клиентов. Аналогичный пароль, с точностью до регистра вводимых символов, должен быть передан пользователем мобильному клиенту при указании начальных настроек для создании сессии удаленного управления.

Процесс аутентификации пользователя построен так, что пароль в явном виде не передается по каналу связи, а используется некоторая генерируемая сервером псевдо случайная последовательность символов для процесса рукопожатия.

На основе всех необходимых данных для подключения к серверу VNC, расположенному на удаленном компьютере, мобильное приложение-клиент производит установление сессии удаленного управления. На этом этапе оба сетевых компонента (клиент и сервер) должны быть подключены к сети и быть доступными друг для друга.

Процесс установления сетевого соединения клиента с сервером заключается в передаче запроса на указанный сетевой TCP порт на соединение по протоколу TCP, после установления соединения происходит проверка версии VNCсервера и обмен информации о совместимости клиента и сервера, определяющей набор функций доступных клиенту при взаимодействии с конкретной реализацией VNC сервера. Далее производится процесс аутентификации клиента на сервере и выполняется рукопожатие сторон сетевого взаимодействия в рамках устанавливаемой сессии сетевого взаимодействия. По завершению этапа аутентификации клиент переходит в режим обработки удаленного кадрового буфера, получаемого от сервера VNC, установленного в ОС на удаленно управляемом компьютере.

В начале новой сессии удаленного управления клиент посылает запрос на получение полного содержимого кадрового буфера и сброса счетчика инкрементных обновлений.

В ходе работы системы, после установления сессии, клиент и сервер обмениваются набором стандартизированных сообщений в режиме запрос-ответ. Сервер передает различные данные о состояние контролируемых им параметров операционной системы. Клиент передает различные управляющие команды для изменения состояния удаленно управляемого компьютера, а так же запрашивает обновленные сведения о состоянии контролируемой сервером операционной системы, инициирует переходы в режимы дополнительной функциональности (для передачи файлов), посылает запросы на изменение настроек сессии удаленного управления, связанных с методами кодирования передаваемого кадрового буфера.

В результате организованная схема удаленного управления компьютером, изображенная на рисунке 3.1 (более подробно с ней можно ознакомится в приложении А), предоставляет возможности по управления, схожие с непосредственным локальным взаимодействием пользователя с ПК.

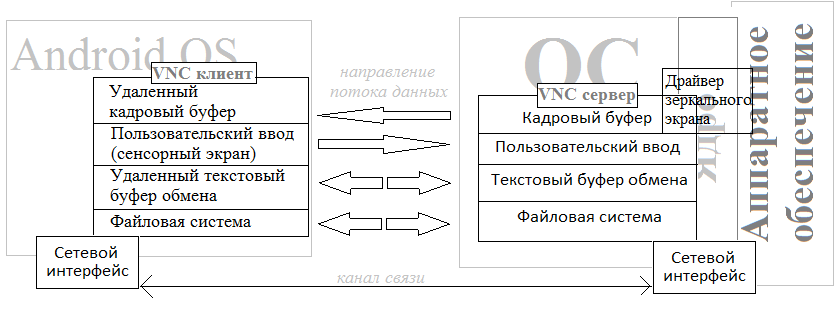


Рисунок 3.1 – Схема работы VNC системы

## Приложение–сервер

Сервер – программа, предоставляющая доступ к экрану компьютера, на котором она запущена, а так же обрабатывающая ввод пользователя, принятый от клиента. Кроме описанных функций, VNC сервер так же предоставляет доступ к текстовому буферу обмена данными, а ряд существующих реализаций поддерживает функции работы с файловой системой, однако данные возможности не описаны в стандарте и не могу требоваться от совместимого сервера.

В рамках данного дипломного проекта разработка собственной реализации VNC сервера не проводилась. Клиент создавался для работы в режиме совместимости с большинством существующих решений на базе VNC сервера и поэтому для работы системы с использованием представленного в данном дипломном проекте мобильного приложения подходит практически любой совместимый VNC сервер. Дополнительной возможностью, которую можно получить от используемого сервера является наличие функций для работы с файлами, их передача между мобильным устройством и удаленным ПК, а так же удаление файлов и создание каталогов в рамках локальной (на используемом мобильном устройстве под управлением Android) и удаленной (на удаленном компьютере работающим как под управлением ОС семейства MS Windows, так и под управлением Unix подобных систем) файловых систем.

VNC сервер работает по протоколу RFB, специально разработанному для решения задач, связанных с организацией графического доступа к удаленному рабочему столу компьютера. Большинство серверов поддерживают работу как в режиме приложения, так и в режиме сервиса в контексте ОС Windows, либо демона в контексте Unix подобных ОС. В стандарте, описывающем работу протокола RFB определен ряд функций и алгоритмов преобразования кадрового буфера которые должны поддерживаться в рамках стандартной VNC системы, причем указан способ преобразования, поддержка использования которого настоятельно рекомендуется к реализации всем без исключения компонентам VNC систем для организации максимально совместимости – «сырое» кодирования, метод без преобразования кадрового буфера.

VNC системы руководствуются идеей «тонких клиентов» и потому в данной архитектуре сервер имеет ведущую роль. В основные задачи сервера входит хранение состояния сессии, отслеживание изменений в кадровом буфере операционной системы и учет этих изменений при отправке данных клиенту. Сервер должен запоминать последние состояние, отправленное клиенту и уметь рассчитывать разницу между последним отправленным клиенту состоянием и состоянием, в котором находится кадровый буфер ОС на текущий момент.

В задачи сервера входя так же функции работы с текстовым буфером обмена данными, среди которых операции чтения буфера с передачи его на клиентское программное обеспечение, а так же записи в текстовый буфера информации, переданной с клиентской части системы.

В общем случае сервер VNC выполняется на стандартном для пользовательских приложений уровне работы в конкретно ОС компьютера, и использует доступные другим приложения средства работы с операционной системой через предоставляемый ею API для доступа к экрану ПК, а также для эмуляции действий пользователя, работы с буфером обмена, передачи файлов.

Кроме отмеченного ранее способа взаимодействия VNC сервера с операционной системой, в контексте которой он выполняется, многие существующие сервера распространяются с дополнительным программным обеспечением, называемым драйвером зеркального экрана (mirror display driver), который предоставляет доступ напрямую к устройству кадрового буфера экрана для выполнения необходимых операций, так как выполняется на уровне системных драйверов. Такой подход существенно ускоряет процесс получения данных об актуальном состоянии кадрового буфера, позволяя повысить время отклика сервера, а так же увеличить степень интеграции сервера с кадровым буфером.

## Мобильное приложение–клиент

Клиент (VNC viewer) – программа, получающая изображение экрана с сервера и взаимодействующая с ним по протоколу RFB, передавая на сервер пользовательский ввод, позволяя обрабатывать содержимое буфера обмена удаленного компьютера, в некоторых конкретных реализациях так же позволяя передавать пользовательские файлы.

Представленное в дипломном проекте программное средство является мобильным VNC клиентом и потому должно реализовывать все необходимые для этого функции. Алгоритм работы мобильного приложения для организации удаленного управления компьютером в качестве клиентского приложения VNC системы изображен в виде блок-схемы в приложении Б.

Клиент поддерживает работу с содержимым удаленного кадрового буфера по протоколу RFB в двух режимах:

– полного обновления содержимого кадрового буфера, когда клиент каждый раз получает полное содержимое удаленного кадрового буфера от сервера, характерна высока избыточность при передаче;

– инкрементного обновление кадрового буфера до актуального состояния, в котором находится удаленный кадровый буфер, когда клиент получает от сервера только закодированные данные об изменившихся участках кадрового буфера.

В режиме инкрементного обновления, как должно быть ясно, объем передаваемой информации наиболее оптимален, что напрямую сказывается на загрузке канала связи между клиентом и сервером, а так же на скорости передачи данных и реакции клиента на произошедшие обновления.

Стоит отметить, что клиент не ведет учета состояния, так как придерживается идеи «тонкого клиента», а потому вся работа по вычислению разницы, на которую отличается состояние отображения содержимого удаленного кадрового буфера на клиенте и его актуальным состоянием на удаленном ПК выполняется на сервере. Этот процесс постоянно поддерживается, для работы в инкрементальном режиме обновления удаленного кадрового буфера и на основе проведенных вычислений сервер передает необходимые данные клиенту. Важной особенностью инкрементного обновления является так же то, что сервер рассчитывает необходимые данные для обновления состояния удаленного кадрового буфера на клиенте только в момент запроса клиентом обновления. Общая схема алгоритма обработки RFB сообщений представлена в виде блок-схемы в приложении В.

Одной из ключевых особенностей в реализации мобильного клиента является метод ввода пользователя, а так же существенные различия в размерах и пропорциях экранов мобильных устройств.

Основным способам ввода пользователя на мобильном устройстве под управлением ОС Android является сенсорный экран, в то время, как ОС на администрируемом удаленном ПК в общем случае рассчитана на клавиатуру и манипулятор типа многокнопочная «мышь». Так же современные персональные компьютеры и другие вычислительные системы находятся под управлением ОС рассчитанных на разрешения от 800 на 600 пикселей и более, а в большинстве своем современные ПК обрабатывает графический пользовательский интерфейс с разрешением 1280 на 1024 и выше, ведь уже достаточно давно широкоформатные экраны мониторов с диагональю 19 и более дюймов стали преобладать при комплектации персональных компьютеров. Разрешение и размеры экрана мобильного устройства под управлением ОС Android существенно меньше, и находится в большинстве своем в пределах диапазона от 320 на 470 до 480 на 640 пикселей при диагонали экрана от 3 до 4 дюймов. Все это накладывает некоторые ограничения и определяет особенности построения клиента VNC для мобильного устройства.

Разработанное приложение сочетает в себе метод ввода, стандартный для мобильного устройства на базе ОС Android (сенсорный экран) и расширяет его возможностью использования датчика приложенного ускорения, которым на текущий момент комплектуется большинство мобильных устройств. Результат взаимодействия пользователя с клиентом VNC интерпретируется в формат ввода, используемый VNC системой и рассчитанный на персональный компьютер – команды ввода с клавиатуры и манипулятора типа «мышь».

Реакция системы на пользовательский ввод с мобильного устройства напрямую зависит от режима, в котором запущен клиент. Представленное в дипломном проекте программное средство поддерживает два режима удаленного управления компьютером: режим удаленного терминала, режим управления презентацией. Режимы различны между собой в подходе к организации ввода, а так же предоставляют различные функции по удаленному управлению и наблюдению за работой управляемого компьютера. Так в режиме удаленного терминала передается содержимое удаленного кадрового буфера управляемого ПК, а с точки зрения ввода одиночные короткие касания сенсорного экрана интерпретируются как клик левой кнопкой мыши, долгое касание как клик правой кнопкой мыши. Присутствует возможность выделения содержимого на экране используя специальный жест.

В режиме управления презентацией содержимое удаленного кадрового буфера управляемого компьютера не передается, а пользовательский ввод интерпретируется иначе. Сенсорный экран выступает в виде манипулятора типа «сенсорная панель», одиночные касания по которой ничего не дают, но присутствует возможность перемещать курсор мыши соответствующими жестами. Для данной функции требуется поддержка обработки множественных одновременных касаний от сенсорного экрана.

Как отмечалось ранее, присутствует возможность перемещать курсор мыши при помощи датчика приложенного ускорения на мобильном устройстве. Эта возможность так же доступна, только в режиме управления презентацией и потребует больше пропускной способности от канала связи с сервером. Суть метода управления в позиционировании устройства в пространстве, при его наклонах и перемещении происходит соответствующее смещение курсора на удаленном администрируемом ПК.

Клавиатурный ввод эмулируется при помощи виртуальной экранной «QWERTY» клавиатуры. Этот способ ввода доступен в обоих режимах выполнения приложения. Так же стоит отметить, что переключения режимов работы приложения возможен без перезапуска самого приложения (однако потребует перезапуска сессии).

В режиме удаленного терминала происходит передача и отображение графического рабочего стала с удаленного управляемого компьютера. В этом режиме в разработанном клиенте поддерживается ряд стандартных методов кодирования передаваемой графической информации, которые в свою очередь могу работать с глубиной цвета в 8 и 24 бита. Ниже приведен список поддерживаемых алгоритмов работы:

– «сырое» кодирование, означает режим без осуществлений преобразований;

– кодирование копированием области экрана;

– кодирование расширяющимися прямоугольниками;

– кодирование сжатыми расширяющимися прямоугольниками;

– преобразование по алгоритму Hextile;

– сжатие содержимого удаленного кадрового буфера c использованием Zlib;

– преобразование по алгоритму Tight;

– преобразование и сжатие содержимого удаленного кадрового буфера по алгоритму ZRLE.

Реализован метод автоматического переключения на оптимальный по нескольким критериям метод преобразования содержимого удаленного кадрового буфера при передаче. Критериями выступают пропускной способности канала связи между клиентом и сервером, качество, обеспечиваемое алгоритмом при передачи, а так же вычислительной нагрузкой, необходимой для операций прямого и обратного преобразования данных удаленного кадрового буфера при передачи или приеме. Кроме того, поддерживается возможность установки пользователем желаемого метода преобразования графической информации при передаче. Стоит отметить, что переключение между режимами отображения кадрового буфера с разной глубина цвета не требует перезапуска сессии.

Кроме функций передачи графического интерфейса, разработанный клиент VNC также обладает базовой функциональностью на подобии FTP клиента и позволяет передавать и получать пользовательские файлы.

В рамках встроенного файлового менеджера, работающего на основе расширения VNC для работы с файлами, можно:

– просматривать содержимое файловых систем на удаленном и локальном устройстве;

– передавать и получать пользовательские файлы в направлениях, как от используемого мобильного устройства к удаленному компьютеру, так и в обратном направлении;

– создавать каталоги на удаленном или локальном устройстве;

– удалять уже существующие файлы.

Однако, как неоднократно отмечалось ранее, функции работы с файлами поддерживается не всеми VNC серверами, и достаточно ограничены. Возможность их использования напрямую зависит от сервера, а так же от реализации данной возможности сервером, в связи с тем, что функции работы с файлами отсутствуют в стандарте, описывающем VNC системы, и являются лишь его расширением, а потому могут быть реализованы с особенностями на усмотрение разработчика используемого сервера VNC.

## Процесс аутентификации

VNC системы проводят аутентификация в два этапа, где на первом шаге запрашивается тип используемой на данный момент сервером режим аутентификации, а второй шаг заключается непосредственно в проведении аутентификации согласно установленному типу. Второй шаг может включать в себя (в зависимости от типа используемого сервером процесса аутентификации) несколько итераций обмена данными в процессе рукопожатия.

Реализованный процесс аутентификации мобильного клиента на сервере VNC поддерживает два режима: без аутентификации и с использованием стандартной VNC аутентификации. Режим без аутентификации не запрашивает ни каких подтверждающих личность пользователя данных и позволяет сразу переходить к стандартному режиму обработки содержимого кадрового буфера, как следует из описанного ранее алгоритма работы реализованной VNC системы.

Стандартная аутентификация для VNC систем подразумевает одну итерацию передачи данных, после которой, в случае успеха выполнения данного этапа, клиент автоматически переходит в режим нормальной обработки данных от сервера. В процессе так называемого «рукопожатия» сервер генерирует псевдослучайную последовательность символов и передает ее клиенту и запоминает ее для себя. Клиент принимает от сервера переданную ему последовательность и проводит над ней криптографическое преобразование. Полученный результат преобразования клиент передает на сервер. Сервер, получив от клиента переданную ему преобразованную последовательность, проводит аналогичные шаги криптографического преобразования над исходной, сохраненной сгенерированной последовательностью для клиента и результат полученного криптографического преобразования сравнивает с полученными от клиента данными. Если полученная от клиента последовательность в точности совпадает с полученной в результате локальных криптографических преобразований последовательностью считается, что аутентификация прошла успешно и клиент является тем, за кого себя выдает.

Среди прочих особенностей реализации системы необходимо отметить, что пароль пользователя используется как ключ для шифрования и его длинна не должна превышать 8 символов, все превышающее игнорируется.

Для выполнения криптографического преобразования используется широко распространённый алгоритм симметричного шифрования DES, его использование оправдывается скоростью выполнения на многих реализациях аппаратных архитектур, однако в связи с не большой длинной ключа (используется ключ длинной 56 бит при длине блока 64 бита) по меркам развития вычислительных систем на момент написания дипломного проекта, нельзя говорит о достаточной крипто стойкости данного алгоритма на длительном периоде использования.

## Реализованные методы кодирования при передачи изображения рабочего стола

Одним из основных факторов, определяющих степень совместимости разрабатываемого VNC клиента и существующих VNC серверов, является полнота их реализации RFB протокола, набор поддерживаемых методов преобразования содержимого удаленного кадрового буфера. Разработанное приложение реализует все основные алгоритмы, описанные в стандарте протокола RFB, а так же поддерживает расширение «Tight», реализующее сжатие графических данных с потерями.

Как отмечалось ранее, клиент поддерживает работу с глубиной цвета в двух режимах: 24 бита и 8 бит. Для работы с 24 битной глубиной используется формат RGB888, а для передачи 8 битного изображения цвет кодируется в формате BGR233. С точки зрения реализации и исходя из особенностей устройства работы ОС Android при получении и хранении состояния удаленного кадрового буфера все данные о цвете каждого пикселя приводятся при необходимости к формату ARGB8888 и хранятся в 32 битных целочисленных массивах, хотя на практике канал прозрачности никогда не используется.

Процесс передачи удаленного кадрового буфера от сервера к клиенту разбит на кадры данных стандартизированной структуры, продемонстрированной на рисунке 3.2.

Каждый кадр всегда содержит в начале четырех байтовый идентификатор метода кодирования, который использовался для преобразования передаваемой в текущем кадре информации об области удаленного кадрового буфера. За идентификатором следуют координаты позиция и размеры закодированной области, выраженные в четырех двух байтовых величинах. Далее следует непосредственно данные в отмеченной ранее кодировке, причем длинна поля с закодированными данными может быть произвольной и для обработки кадра предварительно рассчитывается на основе уже известных параметров. Структура кадра изображена на рисунке 3.2.

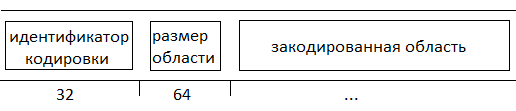


Рисунок 3.2 – Структура кадра для обновления области удаленного кадрового буфера по протокола RFB

Дополнительная информация об особенностях программной реализации (листинги программной реализации) конкретного алгоритма преобразования содержимого удаленного кадрового буфера содержится в приложении Г.

### Кодирование без преобразований

Метод кодирования без преобразований является наиболее универсальным и, согласно стандарту, должен реализовываться всеми без исключения компонентами VNC систем для организации минимальной совместимости решений от различных разработчиков и поддержки обратной совместимости в случае проблем, которые могут возникнуть во время передачи содержимого удаленного кадрового буфера.

Суть метода так же очень проста и заключается в простой передаче требуемой области экрана попиксельно с кодированием цвета каждого пикселя в формате RGB888, либо BGR233 в зависимости от определенной клиентом глубины используемой цветовой палитры. Дополнительных преобразований по оптимизации с целью сжатия передаваемых данных не происходит.

### Кодирование копированием области экрана

Самое оптимальное с позиции, как скорости передачи и обработки, так и объема передаваемых данных. Суть метода заключается в указаний на копирование уже существующей и раскодированной ранее области удаленного кадрового буфера на стороне клиента согласно новым координатам в пределах кадрового буфера. При это не требуется передавать каких-либо графических данных, а следовательно тратить время на их прямое и обратное преобразование. Никаких дополнительных расчетов не ведется, за исключением расчетов на стороне сервера по выявлению уже существующей области кадрового буфера, аналогичной требуемой к обновлению.

Метод используется совместно с другими методами кодирования и не выступает, как самостоятельный метод преобразования передаваемых графических данных из-за того, что им не возможно покрыть все возможные случаи, а для его работы требуется уже ранее раскодированная информация. Совместное применение данного метода позволяет в ряде случаев существенно увеличить скорость передачи и обработки данных, уменьшив объем передаваемых данных и уменьшив время отклика мобильного приложения-клиента.

### Кодирование расширяющимися прямоугольниками

Метод преобразования передаваемых данных на основе кодирования содержимого удаленного кадрового буфера прямоугольными областями заданного цвета. В представленном приложении предложено два схожих метода основанных на модификации известного алгоритма кодирования длин серий:

– кодирование расширяющимися прямоугольниками

– кодирование сжатыми расширяющимися прямоугольниками

Разница подходов к кодированию между упомянутыми разновидностями схожих алгоритмов в том, что в алгоритме сжатых расширяющихся прямоугольников вводится принудительное ограничение на максимально возможную величину позиции, ширины и высоты подобластей в 255 пикселей. Из-за этого сервер вынужден кодировать большую область группой меньших областей. Кодированные данные обновляемой области удаленного буфера в рамках RFB кадра, передаваемого сервером разработанному мобильному клиенту, содержат данные о размере и позиции подобластей, которыми кодируется обновляемый сектор, а так же цвет соответствующий конкретной области, передаваемый перед относящейся к нему области.

### Преобразование по алгоритму Hextile

Алгоритм Hextile работает с подобластями, входящими друг в друга. Работа алгоритма построена на рекурсивном проходе от одной большой области к набору содержащихся в ней областей меньшего размера. Так при движении от больших к меньшим составляющим постепенно увеличивается детализация фрагмента. Порядок укладки областей поверх друг друга производится в прямом порядке, то есть области большего размера, в общем случае, представляют фоновые фрагмента, поверх которых происходит отображение областей меньшего размера.

В процессе кодирования, и соответственно, для декодирования, для областей определяются цвет фона и цвет переднего плана, который явно передается в кадре RFB вместе с закодированными областями данными.

Каждая область может быть закодирована при помощи различных алгоритмов в рамках метода кодирования Hextile. Далее приведены используемые методы кодирования областей, при использовании Hextile:

– без кодирования данных области, когда подобласть передается без проведения над ней каких-либо преобразований и содержимое передаваемых данных представляет собой набор цветов последовательно идущих пикселей в обновляемой области удаленного кадрового буфера;

– сплошным цветом, в данном случае область может быть закрашена переданным ранее сплошным фоновым цветом, а поверх заливки может быть осуществлена закраска сплошным цветом переднего плана области такого же, либо меньшего размера;

– подобластями различных цветов, при этом область может быть закрашена сплошным цветом фона, переданным ранее, а поверх будут уложены дополнительно переданные подобласти с указанными к ним цветами сплошной закраски.

Алгоритм не использует дополнительного сжатия данных. Сам по себе метод достаточно ресурсоемок с точки зрения времени, затрачиваемого на прямое и обратное кодирования, однако обеспечивает разумное соотношение качество передаваемого изображения в отношении к объему передаваемых данных. Преимущество использования данного метода ощутимо при инкрементном методе обновления содержимого удаленного кадрового буфера на мобильном клиенте, однако, при использовании данного метода для обновления всего содержимого за кадр, может сказаться отрицательный эффект в связи высокой вычислительной нагрузкой оказываемой алгоритмом, что проявится как существенное увеличение времени отклика разработанного мобильного приложения.

### Сжатие Zlib

Данный метод достаточно прост в своей идее и заключается в том, что передаваемая область удаленного кадрового буфера без преобразований, осуществляемых над цветовой палитрой данных, сжимается по алгоритму DEFLATE (вариация алгоритма Лемпеля-Зива – LZ77). В остальном данный подход ничем не отличается от описанного ранее подхода без использования какого-либо преобразования.

Алгоритм DEFLATE не реализовывался в рамках данного дипломного проекта, а была взята стандартная реализация данного алгоритма из библиотеки предоставляемой Java SE (а точнее аналогом данной библиотеки, поставляемой с Android SDK).

### Преобразование по алгоритму Tight

Данный алгоритм, реализованный в рамках расширения Tight, которое было включено в набор функций предлагаемого в дипломном проекте программного средства, представляет собой один из наиболее сложных методов преобразования передаваемой области удаленного кадрового буфера. Метод сочетает в себе преобразование на основе сжатия графических данных с потерями согласно формата JPEG и сжатия по алгоритму DEFLATE. Отсюда должно быть ясно, что данный метод является одним из наиболее медленных, среди представленных с точки зрения затрат на прямое и обратное преобразования данных перед передачей и после приема.

Алгоритм оперирует в рамках одной итерации обновления содержимого удаленного кадрового буфера несколькими подходами к сжатию передаваемых данных, выбирая наиболее оптимальную комбинацию. Это решается при помощи такого понятия, как подобласть. Подобласть – это участок обновляемой области удаленного кадрового буфера, которая кодируется каким-то одним из определенных в рамках алгоритма Tight методом, а данные о выбранном методе кодирования и размере подобласти записываются в метаданные, передаваемые в рамках одного кадра. В зависимости от выбранного алгоритмом решения, область кадрового буфера может быть закодирована группой подобластей, в общем случае, имеющих разную кодировку в рамках алгоритма Tight.

В рамках расширения Tight подобласти могут быть:

– сжаты согласно формата jpeg;

– сжаты по алгоритму DEFLATE;

– без сжатия;

– скопированы с другой, переданной ранее области;

– сплошного цвета;

– градиентной заливкой;

– с определенной цветовой палитрой, переданной ранее.

Стандартная реализация алгоритма DEFLATE была взята из библиотеки предоставляемой Java SE (в рамках поставки с Android SDK).

### Преобразование по алгоритму ZRLE

Данный алгоритм так же основан на сжатии по алгоритму DEFLATE, использование которого было основано на библиотечной реализации из пакета поставки Android SDK.

Особенностью данного алгоритма является кодирование передаваемого содержимого области удаленного кадрового буфера на основе фиксированной палитры цветов, вычисляемой в ходе выполнения преобразования и передаваемой вместе с RFB кадром обновления состояния удаленного кадрового буфера на клиенте.

Сжатие по алгоритму DEFLATE проводится в поточном режиме над всем передаваемым кадром, то есть сжатие происходит по мере поступления необходимого блока данных. В начале, после заголовка кадра, в его тело записывается данные о цветовой палитре, затем последовательно после палитры записываются закодированные данные, использующие полученную палитру. На стороне клиента процесс обратного преобразования и разархивации происходит в том же порядке. Распаковка данных производится в поточном режиме по мере накопления достаточного объема данных, необходимого для выполнения процесса обратного сжатию. Согласно алгоритму ZRLE вначале декодируется палитра, потом с ее помощью область пиксельных данных кадрового буфера, закодированная этой палитрой.

Сжатие непосредственно данных о пикселях, содержащихся в кадровом буфере, проводится по известному алгоритму кодирования длин серий (RLE, откуда и название). Согласно которому последовательность одних и тех же символов (пикселей с одинаковым цветом) заменяется более короткой последовательностью, состоящей из одного повторяющегося символа (цвета пикселя, который заносится в палитру) и количества его повторов.

## Передача файлов

Передача файлов осуществляется на основе расширения реализованного в представленном мобильном приложении. Протокол работы расширения основывается на реализации функций передачи файлов, примененной в программном обеспечении UltraVNC, однако схожим образом работают и другие существующие реализации серверов VNC, поддерживающих функции передачи пользовательских файлов.

Реализованное API позволяет выполнять следующие операции:

– получать список логических дисков удаленной файловой системы, при их наличии (в случае файловых систем, используемых в ОС семейства MS Windows);

– получать список содержимого расположенного по указанному удаленному пути;

– запрашивать и получать файл расположенный по указанному удаленному пути;

– удалять файлы по указанному удаленному пути;

– создавать каталоги по указанному удаленному пути.

Процесс передачи и приема файлов реализован в несколько этапов. Первоначально на запрос приема файла сервер пересылает заголовок передаваемого пользовательского файла. Далее процесс приема переходит в режим приема тела запрашиваемого файла, который выполняется итеративно, так как весь процесс передачи построен на пересылке данных файла блоками. На каждой итерации приема файла сервер пересылает длину передаваемого блока, а затем непосредственно блок данных, это процесс повторяется пока весь файл, разбитый на блоки не будет передан. Аналогичным образом происходит отправка локального файла (расположенного в файловой системе используемого мобильного устройства, на котором установлен мобильный клиент VNC) на удаленный ПК по указанному удаленному пути.

Описанные функции по работе с удаленной файловой системой были реализованы в рамках созданного для представленного в дипломном проекте файлового менеджера, который позволяет производить перечисленные операции, как на уделенной, так и на локальной файловой системе. Файловый менеджер реализован в виде встроенной в созданное мобильное приложения подсистемы с распространенным двухоконным интерфейсом. Одно из окон связано с локальной файловой системе мобильного устройства, а второе окно реализует названные операции над удаленной файловой системой.

## Использование датчика приложенного ускорения

В настоящее время большинство устройств средней и дорогой ценовой категории на платформе Android оснащено датчиком приложенного ускорения. Данный датчик снимает показания ускорения, прикладываемого к устройству по трем осям, направление которых показано на рисунке 3.3.

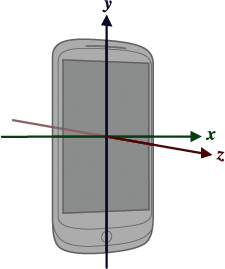


Рисунок 3.3 – Направление осей составляющих ускорения, измеряемые датчиком приложенного ускорения

Для достижения поставленной цели по манипулированию курсором мыши на удаленно управляемом ПК при помощи отмеченного датчика, требуется скорректировать его показания. Дело в том, что данное устройство выдает показания без каких-либо преобразований со стороны ОС, а значит в измерениях постоянно присутствуют составляющие ускорения свободного падения, которые, в целом, являются в данном контексте паразитным шумом.

Для решения описанной проблемы по исключению из получаемых данных значений, вносимых величиной ускорения свободного падения, был реализован высокочастотный фильтр (фильтр высоких частот) согласно формуле:

, (3.1)

где – текущая величина составляющей ускорения свободного падения по данной оси;

*α* – константа, величина частоты среза, ниже которой полученные значения не проходят через фильтр;

– предыдущая вычисленная величина составляющей ускорения свободного падения по данной оси;

*a* – текущее показание датчика приложенного ускорения по данной оси.

Величина константы *α* была подобрана эмпирически входе серии экспериментов и корректировок.

Кроме того, для комфортного использования данного метода управления (по средством показаний датчика приложенного ускорения) потребовалось установить достаточно высокую частоту опроса датчика, что может сказаться на потреблении энергии и в целом повысить нагрузку на систему при работе описываемого мобильного приложения.

## Вывод по разделу

Этап проектирования, как и описанный ранее этап определения инструментов и средств, на начальной стадии разработки всегда следует уделять большое внимание, так как грамотно спроектированная архитектура приложения и заложенные на данной стадии создания программного средства решения определяют возможности получаемого в итоге приложения, влияют на скорость разработки, надежность и стабильность получаемого продукта, а так же определяют потенциал дальнейшего развития программного средства. Это играет главную роль в пригодности программного средства к решению ставящихся задач и напрямую влияют на цикл жизни программного средства в определенном временном периоде. По результатам этапа проектирования были составлены диаграмма классов, представленная в приложении Д, и диаграмма вариантов использования, представленная в приложении Е.

Выбранные в процессе проектирования программного средства (далее ПС) решения по построению клиента VNC системы совместимого с большим количеством существующих серверов за счет реализованного подмножества стандартизированных алгоритмов преобразования кадрового буфера, а так же реализованная полноценная поддержка RFB протокола, с возможностью передачи пользовательского ввода, позволяет использовать представленное мобильное приложение в сочетании с различными VNC серверами для выполнения разнообразных задач так или иначе связанных с процессом удаленного управления компьютером, а так же задач связанных с обменом пользовательскими файлами и упрощенным взаимодействием с файловой системой ПК, либо задач связанных с передачей изображения с экрана удаленного ПК на ряд клиентских мобильных устройств, например при организации каких-либо демонстраций и презентаций.

Так же при проектировании и реализации представленного программного средства были заложены возможности для дальнейшего развития продукта с целью увеличения его полезности с точки зрения пользователя, а так же расширения сферы прикладных задач, решаемых с применением данного программного обеспечения. Основные направления по развитию, которые были заложены для дальнейшей поддержки и реализации можно сгруппировать в следующие группы:

– увеличение количества поддерживаемых мобильным клиентом методов и алгоритмов преобразования содержимого кадрового буфера операционной системы удаленного компьютера;

– реализация более стойких криптографических алгоритмов шифрования информации используемых при процессе аутентификации для защиты системы от несанкционированного доступа;

– добавление поддержки туннелирования потока данных, которым обменивается клиент и сервер, по зашифрованному туннелю для защиты от несанкционированного получения передаваемых данных третьей стороной;

– увеличение производительности работы приложения за счет вынесения критических ко времени выполнения участков кода в скомпилированный машинный код используя средства библиотек и технологию NDK;

– организация сетевого взаимодействия по сетевому протоколу UDP, оперирующему сообщениями и позволяющему увеличить скорость обмена данными между сервером и клиентом;

– расширение числа функций работы с удаленной файловой системой, использование заложенной возможности проверки контрольных сумм при приеме файлов от удаленного компьютера.

В целом все поставленные задачи были успешно выполнены в ходе проектирования и последующей реализации мобильного приложения на платформе Android для удаленного управления компьютером.

Разработанный программный продукт позволяет следующее:

– управлять удаленным ПК в реальном времени;

– отображать графическое состояние, в котором находится установленная на компьютере операционная система;

– выполнять операции с текстовым буфером обмена;

– выполнять операции по созданию каталогов и удалению файлов локально и удаленно, а так же выполнять операции обмена пользовательскими файлами между мобильным устройством и подключенным компьютером;

– использовать при управлении удаленной ОС сенсорный экран мобильного устройства в режимах абсолютного позиционирования курсора мыши на удаленном компьютере или в режиме относительного позиционирования

– использовать для ввода текстовой информации виртуальную клавиатуру;

– управлять курсором мыши удаленной ОС при помощи позиционирования мобильного устройства за счет показаний с датчика приложенного ускорения.