МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«ТЮМЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ НАУК

Кафедра программного обеспечения

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

бакалавра

РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ РЕЗЕРВНОГО КОПИРОВАНИЯ И ВОССТАНОВЛЕНИЯ ДАННЫХ

02.03.03 Математическое обеспечение и администрирование

информационных систем

Профиль «Технологии программирования»

Выполнил работу

студент 4 курса Поляков И. А.

очной формы

обучения

Руководитель

Доцент

Ялдыгин В. Б.

**Оглавление**

[Введение 4](#_Toc106251111)

[Глава 1. ИССЛЕДОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 7](#_Toc106251112)

[1.1 Формальная постановка задачи резервного копирования 7](#_Toc106251113)

[1.2 Основные виды резервного копирования 7](#_Toc106251114)

[1.2.1 Полное резервное копирование 8](#_Toc106251115)

[1.2.2 Инкрементное резервное копирование 9](#_Toc106251116)

[1.2.3 Дифференциальное резервное копирование 10](#_Toc106251117)

[1.2.4 Сравнение видов резервного копирования 12](#_Toc106251118)

[1.3 Характеристики систем резервного копирования 13](#_Toc106251119)

[1.4 Технологии хранения резервных копий и данных 14](#_Toc106251120)

[1.4.1 Накопители на магнитных лентах 14](#_Toc106251121)

[1.4.2 Дисковые накопители 16](#_Toc106251122)

[1.4.3 Твердотельный накопитель 17](#_Toc106251123)

[1.4.4 Сетевые технологии 20](#_Toc106251124)

[1.5 Хранение резервных копий 23](#_Toc106251125)

[1.6 Восстановление данных из резервных копий 25](#_Toc106251126)

[1.6.1 Восстановление данных на чистом компьютере 25](#_Toc106251127)

[1.6.2 Проверка актуальности резервных копий 26](#_Toc106251128)

[1.7 Клиент-серверные модели резервного копирования 26](#_Toc106251129)

[1.8 Сравнение существующих программных решений 28](#_Toc106251130)

[Вывод по главе 28](#_Toc106251131)

[ГЛАВА 2. РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ РЕЗЕРВНОГО КОПИРОВАНИЯ И ВОССТАНОВЛЕНИЯ ДАННЫХ 29](#_Toc106251132)

[2.1 Разработка клиент-серверной модели приложения 29](#_Toc106251133)

[2.2 Архитектура системы 30](#_Toc106251134)

[2.3 Реализованные модули, классы и методы 32](#_Toc106251135)

[2.4 Установка программного обеспечения 38](#_Toc106251136)

[2.5 Описание программного обеспечения 38](#_Toc106251137)

[Вывод по главе 41](#_Toc106251138)

[ЗАКЛЮЧЕНИЕ 42](#_Toc106251139)

[СПИСОК ЛИТЕРАТУРЫ 43](#_Toc106251140)

# Введение

На сегодняшний день информация является одним из важнейших ресурсов и одной из движущих сил прогресса и цивилизации. Информационные технологии активно используются во многих сферах жизнедеятельности человека.

С момента появления разумного человека были придуманы различные способы хранения информации, например зарубки на деревьях, расположение предметов. Существенным развитием этих способов явилась письменность – графическое изображение символов на камне, глине, папирусе, бумаге. Огромное значение в развитии этого направления имело изобретение книгопечатания. За свою историю человечество накопило огромный объем информации в библиотеках, архивах, периодических изданиях и других письменных документах.

В настоящее время объем информации, используемой человеком, настолько велик и разнообразен, что бумажные носители не обеспечивают ее эффективное накопление, хранение и использование. С момента появления первых вычислительных машин получило широкое распространение хранение информации в двоичном коде. Для ее хранения используются разнообразные запоминающие устройства.

Большой объем важной и конфиденциальной информации хранится в цифровом виде. Для ее защиты разработаны различные инструменты шифрования, ограничения прав доступа и резервного копирования.

Резервное копирование в последнее время стало синонимом защиты данных. Защита данных от потери, различных повреждений и других проблем является одной из приоритетных задач для IT компаний.

Разработка приложений для резервного копирования направлена на обеспечение безопасности данных при непредвиденных ситуациях, сбоях аппаратных средств, программного обеспечения.

В настоящее время деятельность значительного большинства предприятий связана с обработкой и хранением данных, а потеря даже части данных может привести к серьезным убыткам и сбоям в работе (вплоть до приостановки деятельности). В связи с этим возникает задача резервирования данных (и их восстановления в случае необходимости). Сегодня существует множество различных видов услуг резервного копирования данных, которые помогают предприятиям, организациям и частным лицам обеспечить безопасность данных и не терять важную информацию в случае стихийных бедствий, кражи или других чрезвычайных ситуаций. Основные причины потерь информации представлены на рисунке 2 [6].



Рис. 2. Основные причины потерь информации

Как правило, в организациях есть несколько источников данных для резервирования, например:

* базы данных;
* файлы и папки с файловых серверов;
* файлы и папки данные с серверов автоматизированных систем;
* данные с компьютеров пользователей.

В связи с резервным копированием, помимо самой задачи создания резервных копий, возникают задачи хранения, управления жизненным циклом и восстановления резервных копий данных. Важным аспектом резервного копирования является вопрос управления и оптимального использования дискового пространства.

При наличии большого количества разнородных источников, для которых требуется резервирование, повышается сложность работы системного администратора по организации резервного копирования и по управлению дисковым пространством.

Для повышения эффективности работы системного администратора по организации и контроля выполнения резервного копирования в случае наличия большого количества и контроля корректности его выполнения целесообразно разработать программное обеспечение, осуществляющее централизованное управление процессом резервирования.

Цель: разработка программного обеспечения для управления процессами создания, хранения, управления жизненным циклом и восстановления резервных копии данных. Целью создания системы является предотвращение потери критически важной информации в случае сбоев или выхода из строя аппаратуры, ошибок программных средств или пользователей, а также злонамеренного уничтожения информации, чем обеспечивается непрерывность технологических процессов и бизнес-процессов предприятия.

Задачи:

* изучить существующие аналоги;
* сформулировать требования к разрабатываемой системе;
* спроектировать разрабатываемую систему;
* определить форматы и структуру данных системы;
* разработать программное обеспечение, реализующее резервирования и восстановление для наиболее востребованных типов источников (файлы, каталоги, базы данных SQL Server) и позволяющее добавлять новые типы источников.

# Глава 1. ИССЛЕДОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

# 1.1 Формальная постановка задачи резервного копирования

# 1.2 Основные виды резервного копирования

Рассмотрим определение термина резервного копирования по версии SNIA (Storage Networking Industry Association) Резервная копия (англ. backup copy) – данные, хранимые на энергонезависимых носителях, обычно удаленно предназначенные для восстановления, в случае если оригинал копии данных утерян или недоступен. Резервное копирование (англ. backup) – процесс создания резервных копий.

Существует несколько распространенных операций копирования, в зависимости от важности хранимой на компьютере информации и частоты её использования, используют несколько видов резервного копирования данных:

* Полное резервное копирование (Full backup).
* Дифференциальное резервное копирование (Differential backup).
* Инкрементное резервное копирование (Incremental backup).

Первым этапом решения задачи резервного копирования является структурирование и анализ информаци, также ее эффективная обработка и определение объекта хранения информации, приоритет и частота использования, оценка количественных характеристик: общий объем данных.

Жизненный цикл информации включает в себя создание, копирование, использование, хранение (краткосрочное/долгосрочное).

Резервое копирование обеспечивает защиту данных вне зависимости от времени и места обработки.

В настоящее время большая часть программных продуктв использует хранение резервных копий в облаке, сетевых хранилищах инфорации, что позволяет восстановить информацию полностью при потере данных на локальных машинах.

# 1.2.1 Полное резервное копирование

Является главным и основополагающим методом создания резервных копий, резервная копия содержит все данные о файлах и папках, выбранных для резервного копирования, при таком способе их проще восстановить. Это наиболее полный и надежный вид резервного копирования, хотя и самый затратный. В случае необходимости сохранить несколько копий данных общий хранимый объем будет увеличиваться пропорционально их количеству. Для предотвращения большого объёма использованных ресурсов используют алгоритмы сжатия, а также сочетание этого метода с другими видами резервного копирования: инкрементным или дифференциальным. И, конечно, полное резервное копирование незаменимо в случае, когда нужно подготовить резервную копию для быстрого восстановления системы с нуля.

Достоинства метода:

1. Легкий поиск файлов - Поскольку выполняется резервное копирование всех данных, содержащихся на устройстве, для поиска нужного файла не требуется просматривать несколько носителей.
2. Текущая резервная копия всей системы всегда расположена на одном носителе или наборе носителей - Если потребуется восстановить всю систему, то всю необходимую информацию можно найти в последней полной резервной копии.

Недостатки метода:

1. Избыточная защита данных - поскольку большинство файлов системы изменяются достаточно редко, то каждая последующая полная резервная копия представляет собой копию данных, сохраненных в ходе первого полного резервного копирования. Для полного резервного копирования требуется большой объём носителя.
2. Полное резервное копирование занимает больше времени - Для создания полных резервных копий может потребоваться длительное время, в особенности, если для хранения выбраны устройства в сети.

Полное резервное копирование чаще ограничивается ежедневным или еженедельным графиком, но при достаточно большой скорости и емкости носителя повляется возможность значительно снизить временное окно копирования.

Также есть проблема безопасности, полные резервные копии содержат копию всех данных, потому нужно убедиться в использовани шифрования для защиты резервных копий.

# 1.2.2 Инкрементное резервное копирование

В отличие от полного резервного копирования в этом случае копируются не все данные а только те, что были изменены со времени последнего копирования. Для определения времени копирования могут применяться различные методы, например, в системах Windows используется соответствующий атрибут файла (архивный бит), который устанавливается, когда файл был изменен, и сбрасывается программой резервного копирования. В других системах используется дата изменения файла. Схема с применением данного вида резервного копирования будет неполноценной, если время от времени не делать полное резервное копирование. При восстановлении данных нужно провести восстановление из последней копии, созданной Full backup, а потом поочередно восстановить данные из инкрементных копий в порядке их создания. Данный вид используется для того, чтобы в случае создания архивных копий сократить расходуемые объемы на устройствах хранения информации (например, сократить число используемых ленточных носителей). Также это позволит минимизировать время выполнения заданий резервного копирования, что может быть важно в условиях, когда машина работает постоянно. У инкрементного копирования есть один нюанс: поэтапное восстановление возвращает и нужные удаленные файлы за период восстановления. Например: допустим, по выходным дням выполняется полное копирование, а по будням инкрементное. Пользователь в понедельник создал файл, во вторник его изменил, в среду переименовал, в четверг удалил. Так вот при последовательном поэтапном восстановлении данных за недельный период мы получим два файла: со старым именем за вторник до переименования, и с новым именем, созданным в среду. Это произошло потому, что в разных инкрементных копиях хранились разные версии одного и того же файла, и в итоге будут восстановлены все варианты. Поэтому при последовательном восстановлении данных из архива «как есть» имеет смысл резервировать больше дискового пространства, чтобы смогли поместиться в том числе и удаленные файлы, потому как при инкрементном копировании обрабатываются файлы, измененные со времени последнего резервного копирования, каждая резервнаяя копия является приращением предыдущей.

При восстановлении информации после инкрементного резервного копирования уйдет больше времени по сравнению с остальными методами резервирования, так как необходима полная резервная копия, а также каждая инкрементная, созданная после последнего полного резервного копирования.

Достоинства метода:

1. Эффективное использование носителей - поскольку сохраняются только файлы, измененные с момента последнего полного или инкрементного резервного копирования, резервные копии занимают меньше места.
2. Меньшее время резервного копирования - инкрементное резервное копирование занимает меньше времени, чем полное и дифференциальное резервное копирование из-за сохранения только измененных файлов.

Недостаток метода:

Восстановление происходит медленнее по сравнению с другими типами резервного копирования (необходима полная резервная копия и остальные инкрементные копии).

# 1.2.3 Дифференциальное резервное копирование

Отличается от инкрементного тем, что копируются данные с последнего выполнения полного резервного копирования . Данные при этом помещаются в архив «нарастающим итогом». В системах Windows этот эффект достигается тем, что архивный бит при дифференциальном копировании не сбрасывается, поэтому измененные данные попадают в архивную копию, пока полное копирование не обнулит архивные биты. В силу того, что каждая новая копия, созданная таким образом, содержит данные из предыдущей, это более удобно для полного восстановления данных. Для этого нужны только две копии: полная и последняя из дифференциальных, поэтому вернуть к жизни данные можно гораздо быстрее, чем поэтапно накатывать все как при инкрементном копировании. К тому же этот вид копирования избавлен от вышеперечисленных особенностей инкрементного, когда при полном восстановлении удаленные файлы также возвращаются. Возникает меньше путаницы. Но дифференциальное копирование проигрывает инкрементному в экономии требуемого пространства. Так как в каждой новой копии хранятся данные из предыдущих, суммарный объем зарезервированных данных может быть сопоставим с полным копированием. И, конечно, при планировании расписания (и расчетах, поместится ли процесс бэкапа во временное окно) нужно учитывать время на создание последней, самой большой, дифференциальной копии.

Достоинства метода:

1. Легкий поиск файлов - для восстановления системы, защищенной с помощью стратегии дифференциального резервного копирования требуются две резервные копии - последняя полная резервная копия и последняя дифференциальная резервная копия. Время восстановления меньше по сравнению со стратегиями резервного копирования, для которых требуются последняя полная резервная копия и все инкрементальные резервные копии, созданные с момента последнего полного резервного копирования.
2. Меньшее время резервного копирования и восстановления - Дифференциальное резервное копирование занимает меньше времени, чем полное резервное копирование. Восстановление после выполняется быстрее, поскольку для полного восстановления устройства необходимы только последняя полная резервная копия и дифференциальная резервная копия.

Недостаток метода:

1. Избыточная защита данных - сохраняются все файлы, измененные с момента последнего полного резервного копирования. Таким образом, создаются избыточные резервные копии файлов, созданных при предыдущих дифференциальных копированиях.

# 1.2.4 Сравнение видов резервного копирования

В Таблице 1 представлен принцип работы основных видов резервного копирования. Происходят 4 процесса резервного копирования. Первый процесс может быть только полным резервным копированием, так как без него инкрементное и дифференциальное резервное копирование невозможно.

Таблица 1 – Сравнение видов резервного копирования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер резервного | Полное | Инкрементное | Дифференциальное | |
| копирования |  |  |  |  |
|  |  |  |  |  |
| Backup 1 | Все данные | – | – |  |
|  |  |  |  |  |
| Backup 2 | Все данные | Изменения после | Изменения | после |
|  |  | backup1 | backup1 |  |
|  |  |  |  |  |
| Backup 3 | Все данные | Изменения после | Изменения | после |
|  |  | backup2 | backup1 |  |
|  |  |  |  |  |
| Backup 4 | Все данные | Изменения после | Изменения | после |
|  |  | backup3 | backup1 |  |
|  |  |  |  |  |

# 1.3 Характеристики систем резервного копирования

Важную роль при выборе систем резервного копирования играют характеристики процессов хранения и резервирования. Их необходимо оговаривать в контракте с поставщиком программного обеспечения. Требования к данным характеристикам описываются в отдельном соглашении об уровне услуг SLA (англ. Service Level Agreement).

При выдвижении требований к характеристикам обычно оперируют такими терминами, как RPO, RTO, Backup Window.

Backup Window (окно резервного копирования) – количество времени, необходимое для выполнения операций резервного копирования на целевой системе.

RPO (англ. Recovery Point Objective) – момент времени для восстановления данных. RPO определяет насколько часто требуется производить операции резервного копирования и какое количество резервных копий нужно хранить.

RTO (англ. Recovery Time Objective) – время для восстановления системы в случае необходимости. RTO имеет низкое значение если существует копия необходимых данных. Использование технологии CDP также характеризуется низким RTO.

RTA (англ. Recovery Time Actual) – действительное время восстановления. Используется в сочетании со временем RTO. Определяется экспериментально, при проведении тестирования.

Data security (безопасность данных) – уровень защиты от неавторизованного доступа к хранимой информации. Это может подразумевать как защиту от физического доступа к системам хранения, так и шифрование данных.

# 1.4 Технологии хранения резервных копий и данных

В процессе выполнения резервного копирования данных появляется проблема выбора технологии хранения резервных копий и данных. В настоящее время особой популярностью пользуются следующие виды носителей:

* Накопители на магнитных лентах.
* Дисковые накопители.
* Сетевые технологии.

# 1.4.1 Накопители на магнитных лентах

Не только в крупных корпорациях, но и на предприятиях малого бизнеса хорошо понимают необходимость резервного копирования и восстановления информации. В системах масштаба предприятия и сетях крупных департаментов, в небольших компаниях и у индивидуальных пользователей одинаковым успехом пользуются потоковые накопители, или стримеры. В основе их конструкции лежит лентопротяжный механизм, работающий в инерционном режиме. Для обоснованного выбора системы резервного копирования надо ясно представлять себе достоинства и недостатки разных устройств, которые во многом определяются емкостью системы, ее быстродействием, надежностью и ценой. Основные стимулы к повышению производительности ленточных устройств среднего и старшего класса - это широкое использование Интернета и распространение корпоративных интрасетей, увеличение числа серверов (нужных, чтобы обеспечить рост этих сетей), а также ужесточение требований к хранению информации и ее восстановлению в случае непредвиденных ситуаций. Спрос на системы резервного копирования и хранения данных особенно подстегивается все более активным использованием таких приложений, как мультимедиа, видео по запросу, звуковое информационное наполнение, обработка изображений и т.п. Применяются два метода записи на магнитную ленту: наклонный и линейный серпантинный. В системах наклонной записи несколько считывающих/записывающих головок размещают на вращающемся барабане, установленном под углом к вертикальной оси (аналогичная схема применяется в бытовой видеоаппаратуре). Движение ленты при записи/чтении возможно только в одном направлении. В системах линейной серпантинной записи считывающая/записывающая головка при движении ленты неподвижна. Данные на ленте записываются в виде множества параллельных дорожек (серпантина). Головка размещается на специальной подставке; по достижении конца ленты она сдвигается на другую дорожку. Движение ленты при записи/чтении идет в обоих направлениях. На самом деле таких головок обычно устанавливается несколько, чтобы они обслуживали сразу несколько дорожек (они образуют несколько каналов записи/чтения).

Плюсы хранения данных на ленточном носителе:

* низкая стоимость.
* низкое энергопотребление накопителя.
* большие объемы данных.
* простой способ увеличения объема хранимых данных без значительных инвестиций.

Минусы хранения данных на ленточном носителе:

* низкая скорость доступа к данным.
* сложный процесс обработки параллельных запросов к данным.

# 1.4.2 Дисковые накопители

Существует два наиболее часто встречающихся вида дисковых накопителей: накопители на жёстких магнитных дисках и накопители на оптических дисках.

Накопители на жестких магнитных дисках (Hard Disk Drive, HDD) являются основными устройствами оперативного хранения информации. Для современных одиночных накопителей характерны объемы от сотен мегабайт до нескольких гигабайт при времени доступа 5-15 мс и скорости передачи данных 1-10 Мбайт/с. Относительно корпуса сервера различают внутренние и внешние накопители. Внутренние накопители существенно дешевле, но их максимальное количество ограничивается числом свободных отсеков корпуса, мощностью и количеством соответствующих разъемов блока питания сервера. Установка и замена обычных внутренних накопителей требует выключения сервера, что в некоторых случаях недопустимо. Внутренние накопители с возможностью "горячей" замены (Hot Swap) представляют собой обычные винчестеры, установленные в специальные кассеты с разъемами. Кассеты обычно вставляются в специальные отсеки со стороны лицевой панели корпуса, конструкция позволяет вынимать и вставлять дисководы при включенном питании сервера. Для стандартных корпусов существуют недорогие приспособления (Mobile Rack), обеспечивающие оперативную съемность стандартных винчестеров. Внешние накопители имеют собственные корпуса и блоки питания, их максимальное количество определяется возможностями интерфейса. Обслуживание внешних накопителей может производиться и при работающем сервере, хотя может требовать прекращения доступа к части дисков сервера.

Для больших объемов хранимых данных применяются блоки внешних накопителей - дисковые массивы и стойки, представляющие собой сложные устройства с собственными интеллектуальными контроллерами, обеспечивающими, кроме обычных режимов работы, диагностику и тестирование своих накопителей. Более сложными и надежными устройствами хранения являются RAID-массивы (Redundant Array of Inexpensive Disks - избыточный массив недорогих дисков). Для пользователя RAID представляет собой один (обычно SCSI) диск, в котором производится одновременная распределенная избыточная запись (считывание) данных на несколько физических накопителей (типично 4-5) по правилам, определяемым уровнем реализации (0-10). Например, RAID Level 5 позволяет при считывании исправлять ошибки и осуществлять замену любого диска без остановки обращения к данным.

Устройства считывания компакт-дисков CD-ROM расширяют возможности системы хранения данных NetWare. Существующие накопители обеспечивают скорость считывания от 150 кбайт/с до 300/600/900/1500 Кбайт/c для 2-,4-,6- и 10-скоростных моделей при времени доступа 200-500 мс. NetWare позволяет монтировать компакт-диск как сетевой том, доступный пользователям для чтения. Объем тома может достигать 682 Мбайт (780 Мбайт для Mode 2). Устройства CD-ROM выпускаются с различными интерфейсами, как специфическими (Sony, Panasonic, Mitsumi), так и общего применения: IDE и SCSI. Сервер NetWare обслуживает только CD-ROM с интерфейсами SCSI, новые драйверы существуют и для IDE; устройства со специфическими интерфейсами могут использоваться только в DOS для инсталляции системы. С точки зрения повышения производительности предпочтительнее использование CD-ROM SCSI, однако они существенно дороже аналогичных IDE-устройств. В сервере с дисками SCSI применение CD-ROM с интерфейсом IDE может оказаться невозможным из-за конфликтов адаптеров.

Достоинствами таких накопителей является:

* быстрый доступ к данным.
* возможность параллельного доступа к данным без значительной потери скорости.

Недостатки дисковых накопителей:

* более высокая стоимость чем ленты.
* более высокое энергопотребление.
* более дорогое расширение системы хранения даннх.
* невозможность обеспечения высокой безопасности копий.

# 1.4.3 Твердотельный накопитель

**Твердотельный накопитель** (англ. Solid-State Drive, **SSD**) — компьютерное энергонезависимое немеханическое запоминающее устройство на основе микросхем памяти, альтернатива жёстким дискам (HDD). Наиболее распространённый вид твердотельных накопителей использует для хранения информации флеш-память типа NAND, однако существуют варианты, в которых накопитель создаётся на базе DRAM-памяти, снабжённой дополнительным источником питания — аккумулятором. Помимо собственно микросхем памяти, подобный накопитель содержит управляющий контроллер.

В настоящее время твердотельные накопители используются как в носимых (ноутбуках, нетбуках, планшетах), так и в стационарных компьютерах для повышения производительности. На 2016 год наиболее производительными выступали SSD формата M.2 с интерфейсом NVMe, у которых при подходящем подключении скорость записи/чтения данных могла достигать 3800 мегабайт в секунду.

По сравнению с традиционными жёсткими дисками твердотельные накопители имеют меньший размер и вес, являются беззвучными, а также многократно более устойчивы к повреждениям (например, при падении) и имеют гораздо бóльшую скорость производимых операций. В то же время, они имеют в несколько раз бóльшую стоимость в пересчёте на гигабайт и меньшую износостойкость (ресурс записи)

SSD представляют собой устройства, хранящие данные в микросхемах вместо вращающихся металлических дисков или магнитных лент. Причина их появления отражает тот факт, что скорость обработки данных в процессоре намного превышает скорость записи данных в HDD. Магнитные диски на протяжении десятилетий доминировали в корпоративном сегменте хранения данных, за это время (с 1950-х) ёмкость носителей выросла в двести тысяч раз, скорость работы процессоров тоже сильно возросла, но скорость доступа к данным изменилась значительно меньше и диски стали «узким местом». Проблему решают твердотельные накопители — они обеспечивают намного большие скорости работы с данными по сравнению с жёсткими дисками[[3]](https://ru.wikipedia.org/wiki/%D0%A2%D0%B2%D0%B5%D1%80%D0%B4%D0%BE%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%BD%D0%B0%D0%BA%D0%BE%D0%BF%D0%B8%D1%82%D0%B5%D0%BB%D1%8C#cite_note-_d2e28494c107a990-3). SSD за счёт использования микросхем флеш-памяти по своим характеристикам существенно отличаются от жёстких дисков с магнитными пластинами.

Основные характеристики твердотельных накопителей:

наименьшее время доступа к данным: от 100 до 1000 раз быстрее, чем у механических дисков;

высокая скорость, вплоть до нескольких гигабайт в секунду для произвольно расположенных данных;

высокие значения IOPS благодаря высокой скорости и низкому времени доступа;

низкая цена производительности, лучшее соотношение цены к производительности среди всех устройств хранения;

высокая надёжность; SSD дают уровень сохранности данных такой же, как другие полупроводниковые устройства.

В отличие от жёстких дисков, цена SSD очень сильно зависит от доступной ёмкости, что связано с ограниченной плотностью размещения ячеек памяти и ограничением размера кристалла в микросхеме

Достоинства:

* Доступность.
* Компактность.
* Эстетичность.
* Долговечность в случае рациональной эксплуатации.

Недостатки:

* Риск потери вследствие компактности.
* Риск физического повреждения внешнего или внутреннего USB-порта.
* Выход из строя по строку службы.

# 1.4.4 Сетевые технологии

Услуги удаленного резервного копирования приобретают все большую популярность. Резервное копирование через Интернет на удаленный компьютер может защитить от некоторых наихудших сценариев, таких как пожары, наводнения или землетрясения, которые разрушили бы любые резервные копии в непосредственной близости от места катастрофы.

Есть, однако, ряд недостатков для способа удаленного резервного копирования. Во-первых, Интернет-соединение, как правило, медленнее, чем локальные устройства хранения данных. Во-вторых, пользователи должны доверять стороннему поставщику услуг, чтобы быть уверенным в сохранении конфиденциальности и целостности своих данных, хотя конфиденциальность может быть обеспечена путем шифрования данных перед передачей в службу резервного копирования с помощью ключа шифрования, известным только пользователю.

Достоинства:

* Экономия места на локальной машине.
* Расположение хранилища данных в отдалении от локальной машины.
* Возможность доступа к архивным данным с любой точки, имеющей доступ к Интернету.
* Большой объем пространства для хранения данных.
* Разграничение прав доступа.
* Большинство удаленных служб использует шифрование при передаче данных.

Недостатки:

* Платный доступ к большим объемам удаленного пространства.
* Зависимость от наличия подключения к Интернету.
* Хранение конфиденциальной информации на посторонних ресурсах.

Сетевое хранение данных построено на трех фундаментальных компонентах: коммутации, хранении и файлах. Все продукты хранения можно представить в виде комбинации функций данных компонентов. Поначалу это может вызвать замешательство: поскольку продукты хранения разрабатывались по совершенно разным направлениям, функции часто перекрывают друг друга.

В сети работает множество приложений типа «клиент-сервер» и различных видов распределенных приложений, но в то же время хранение является уникальным и специализированным типом приложения, которое может функционировать в нескольких сетевых средах. Поскольку процессы хранения тесно интегрированы с сетями, будет уместно напомнить, что сетевые хранилища представляют собой системные приложения. Сервисами, которые предоставляются сетевыми приложениями хранения, могут пользоваться сложные корпоративные программы и пользовательские приложения. Как и в случае со многими технологиями, некоторые типы систем лучше отвечают требованиям сложных приложений высокого уровня.

Термин «коммутация» применяется ко всему программному и аппаратному обеспечению и к службам, которые обеспечивают транспортировку хранения и управление ею в сетевом хранилище. Сюда входят такие различные элементы, как разводка кабелей, сетевые контроллеры ввода-вывода, коммутаторы, концентраторы, аппаратура выборки адресов, контроль связи данных, транспортные протоколы, безопасность и резервы ресурсов. В сетевых хранилищах все еще широко используются технологии шин данных SCSI и ATA, и, скорее всего, они будут использоваться еще долго. Фактически продукты SCSI и ATA сегодня применяются гораздо чаще в технологии NAS. Существуют два важных различия между сетями хранения SAN и обычными локальными сетями LAN. Сети хранения SAN автоматически синхронизируют данные между отдельными системами и хранилищами. В сетевых хранилищах необходимы компоненты высокой степени точности для обеспечения надежной и предсказуемой среды. Несмотря на ограничения по расстоянию, параллельная SCSI - чрезвычайно надежная и предсказуемая технология. Если новые технологии коммутации, такие как Fibre Channel, Ethernet и InfiniBand, сменят SCSI, они должны будут продемонстрировать аналогичный или лучший уровень надежности и предсказуемости. Имеется и такая точка зрения, которая рассматривает коммутацию как канал хранилища. Сам термин «канал», берущий свое начало в среде больших вычислительных машин, предполагает высокую надежность и работоспособность.

Хранение в основном затрагивает блочные операции адресного пространства, включая создание виртуальной среды, когда адреса логического блока хранения отображаются из одного адресного пространства в другое. Вообще говоря, в сетевых хранилищах функция хранения почти не изменилась, если не считать двух заметных отличий. Первое - это возможность нахождения технологий виртуализации устройства, например управление устройством внутри оборудования сетевого хранения. Этот вид функции иногда называют контроллером домена хранения или виртуализацией LUN. Второе главное отличие хранения заключается в масштабируемости. Продукты хранения, такие как подсистемы хранения, имеют значительно больше контроллеров/интерфейсов, чем предыдущие поколения шинной технологии, а также намного больший объем хранения.

Функция организации файлов представляет абстрактный объект конечному пользователю и приложениям, а также организует разметку данных на реальных или виртуальных устройствах хранения. Основную часть функциональности файлов в сетевых хранилищах обеспечивают файловые системы и базы данных; их дополняют приложения управления хранением, например операции резервного копирования, также являющиеся файловыми приложениями. Сетевое хранение к настоящему времени почти не изменило файловые функции, за исключением разработки файловых систем NAS, в частности файловой системы WAFL компании Network Appliance. Кроме упомянутых технологий хранения данных NAS и SAN, ориентированных на крупные и глобальные сети, в небольших локальных сетях доминирующее положение занимает технология DAS, в соответствии с которой хранилище находится внутри сервера, обеспечивающего объем хранилища и необходимую вычислительную мощность.

Простейшим примером DAS может служить накопитель на жестком диске внутри персонального компьютера или ленточный накопитель, подключенный к единственному серверу. Запросы ввода- вывода (называемые также командами или протоколами передачи данных) непосредственно обращаются к этим устройствам. Однако такие системы плохо масштабируются, и компании с целью расширения объема хранилища вынуждены приобретать дополнительные серверы. Эта архитектура очень дорогая и может использоваться только для создания небольших по объему хранилищ данных.

# 1.5 Хранение резервных копий

Когда резервные копии сделаны, они должны быть сохранены. Однако, совсем не так очевидно, что именно следует хранить и где. Чтобы правильно определить место хранения копий, нужно сначала учесть обстоятельства, при которых будут использоваться резервные копии. Можно выделить три основные ситуации:

* Восстановление отдельных файлов по запросу пользователей.
* Глобальное восстановление при чрезвычайной ситуации.
* Архивное хранилище (скорее всего никогда не потребуется).

Между первой и второй ситуацией существуют несовместимые противоречия. Когда пользователь удаляет файл случайно, он хочет возвратить его немедленно. Следовательно, резервный носитель должен быть не дальше нескольких метров от компьютера, на котором должны быть восстановлены данные. В случае чрезвычайных ситуаций необходимо будет выполнить полное восстановление одного или нескольких компьютеров в вашем центре данных, а если произошедший сбой будет иметь физический характер, он разрушит не только компьютеры, но и все резервные копии, хранящиеся рядом. Вопрос архивного хранилища менее спорный - вероятность того, что администратор воспользуется им, довольно мала, поэтому если резервный носитель хранится далеко от центра данных, это не должно быть проблемой. Практика показывает, объем редко запрашиваемых данных в архиве составляет примерно 80% от объема всех данных хранящихся на оперативном носителе, при этом около 20% архивных данных не будут востребованы. Отправляя такие данные в архивный носитель, можно освободить до 80% объема на оперативном носителе, что уменьшает объем и окно резервного копирования. Для решения этих задач могут быть выбраны различные подходы, в зависимости от потребностей организации. Первый возможный подход заключается в хранении копий за несколько дней у себя на месте, а затем переносить эти копии в безопасный удалённый носитель, когда будут созданы новые ежедневные копии. Другой подход заключается в поддержке двух наборов носителей:

Набор носителей в центре данных, используемый исключительно для восстановления отдельных данных по запросу

Набор носителей для удалённого хранения и восстановления в случае чрезвычайных ситуаций

Наличие двух наборов подразумевает необходимость делать все резервные копии дважды или копировать их. Это можно сделать, но двойное резервное копирование может занять много времени, а для копирования резервных копий могут потребоваться несколько устройств для работы с резервными копиями (и возможно, выделить для копирования отдельный компьютер. Сложность для системного администратора заключается в выдерживании баланса между удовлетворением нужд пользователей и наличием резервных копий на случай наихудших ситуаций.

# 1.6 Восстановление данных из резервных копий

В большинстве случаев резервные копии выполняются ежедневно, а восстановление, как правило, происходит реже. Однако, восстановления неизбежно, в нём обязательно будет необходимость, поэтому к нему лучше подготовиться. Здесь важно проанализировать две важные ситуации, возникающие при восстановлении данных из резервных копий:

Восстановление данных на чистом компьютере.

Проверка актуальности резервных копий.

# 1.6.1 Восстановление данных на чистом компьютере

Восстановление данных на чистом компьютере - это процесс восстановления полной копии системы на компьютере, на котором нет никаких данных. Можно выделить два основных подхода к восстановлению на компьютере:

Переустановка, за которой следует восстановление, здесь базовая операционная система устанавливается таким же образом, как и на совершенно новый компьютер. Когда операционная система установлена и правильно настроена, оставшиеся диски можно подключить и отформатировать, и восстановить все копии с резервных носителей.

Диск для восстановления системы - это загрузочный носитель некоторого рода (обычно CD-ROM), который содержит минимальное системное окружение и позволяет выполнять самые основные административные задачи. Окружение восстановления содержит необходимые утилиты для разбиения на разделы и форматирования дисков, драйверы устройств, необходимые для обращения к устройству с резервными копиями, и программы, необходимые для восстановления данных с резервных носителей.

# 1.6.2 Проверка актуальности резервных копий

Все типы копий следует периодически проверять, чтобы убедиться в том, что эти копии можно прочитать и что они являются актуальными на настоящее время. Действительно, иногда копии, по той или иной причине, могут не читаться, чаще всего это обнаруживается только при потере данных, когда требуется резервная копия. Причины этого могут быть самыми разными, например: смещение головки стримера, неправильно настроенная программа резервного копирования и ошибка оператора. Но какова бы не была причина, не проводя периодических проверок, администратор не может быть уверен в том, что действительно есть резервные копии, с которых когда-нибудь позже можно будет восстановить данные.

# 1.7 Клиент-серверные модели резервного копирования

На сегодняшнее время выделяются две клиент-серверные модели для резервирования данных:

* Централизованная модель.
* Децентрализованная модель.

При централизованной модели используется четкая иерархическая модель[http://samag.ru/archive/article/1115], на каждый компьютер устанавливается клиентское приложение-агент, а на сервер приложение сервер, для управления заданиями клиентов. По такому принципу работает большая часть популярных систем резервирования данных на сегодняшний день.

Принцип работы: в серверном приложении настраивается план резервирования, и дополнительные параметры, затем сервер передает агентам клиентов команды для выполнения резервного копирования, а агенты следуя командам выполняют задания резервирования и отправки на сервер, восстановления данных, передают серверу сообщения об ошибках.

Преимущества:

* Масштабируемость.
* Отказоустойчивость.
* Удобное управление.

Недостатки:

* Сложность реализации

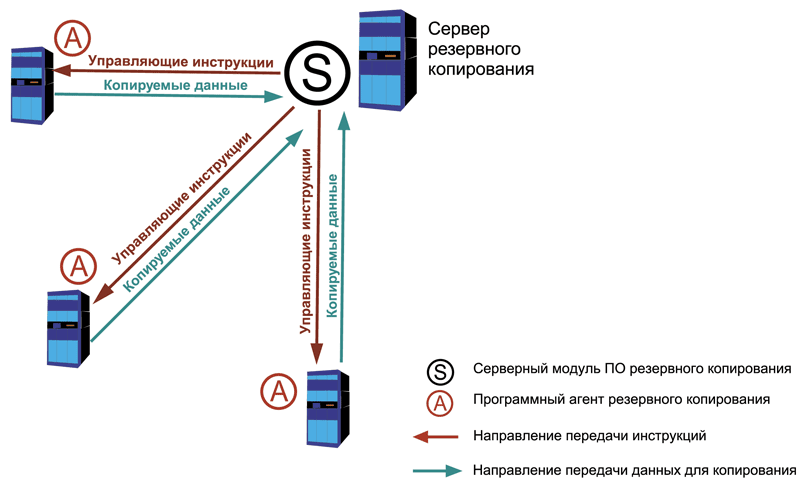


Рис. Централизованная модель резервирования данных.

Децентрализованная модель построена так, что сервером модели является сетевой общий ресурс, как пример сетевая папка или FTP-сервер. Клиентом является программа для резервного копирования, которая создает резервные копии данных и отправляет их на сервер. Используется в небольших сетях организаций, и при отсутствии возможности реализации централизованной модели резервного копирования имеющимися средствами.

Преимущества:

* Простота реализации.
* Низкая стоимость.

Недостатки:

* Нет масштабируемости
* Сложность администрирования

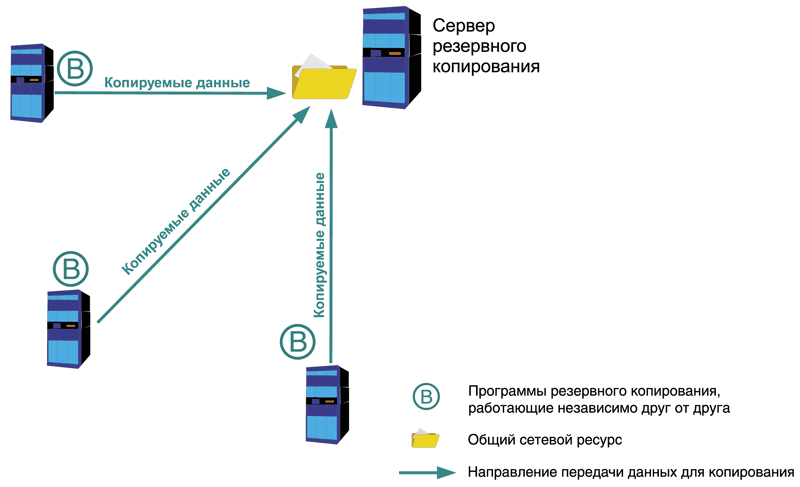


Рис. Децентрализованная модель резервирования данных.

При рассмотрении данных моделей можно сделать вывод, что следует применять централизованную модель если возможна необходимость дальнейшего расширения, в остальных случаях можно ограничиться децентрализованной.

# 1.8 Сравнение существующих программных решений

## Вывод по главе

В данной главе были рассмотрены технологии резервного копирования и хранения данных, приведены их преимущества и недостатки.

# ГЛАВА 2. РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ РЕЗЕРВНОГО КОПИРОВАНИЯ И ВОССТАНОВЛЕНИЯ ДАННЫХ

## 2.1 Разработка клиент-серверной модели приложения

На основе обзора решений и технологий в области резервирования данных в Главе 1, мною разработана централизованная модель приложения, осуществляется полное резервное копирование.

В качестве среды разработки используется Visual Studio 2022, в качестве языка программирования используется объектно-ориентированный язык программирования C#.

Для создания интерфейса приложения администратора используется технология Windows Forms (WinForms).

Целевая платформа – NET 6.0.

При разработке исходный код проекта сохранялся на GitHub (адрес репозитория: <https://github.com/polyakovGit/BackupSystem>).

Решение включает 4 проекта:

* ServerService – управляющая служба, реализована в виде службы Windows;
* ClientService – подчиненная служба, также реализована в виде службы Windows;
* DesktopClient – приложение с графическим интерфейсом. С его помощью пользователь может добавлять, редактировать, удалять, а также мониторить список задач;
* SharedData – библиотека классов, компилируемая в DLL и используемая остальными компонентами системы.
* ClientConfig – класс для хранения конфигурации подчиненной службы, используется для автономного подключения к серверу.

Взаимодействие между приложениями осуществляется по сети по протоколу TCP/IP. Для этих целей используется библиотека Network [1]. Библиотека позволяет посредством сети получать и отправлять пакеты с различными данными. Для передачи списка задач используется пакет ‘tasks’. Передача файлов осуществляется с помощью пакета ‘backup’.

Управляющая служба сохраняет список задач в файле Tasks.json. Для сохранения списка в формате JSON применяется библиотека Newtonsoft.Json [2].

## 2.2 Архитектура системы

Архитектура разрабатываемой системы приведена на рисунке 4.

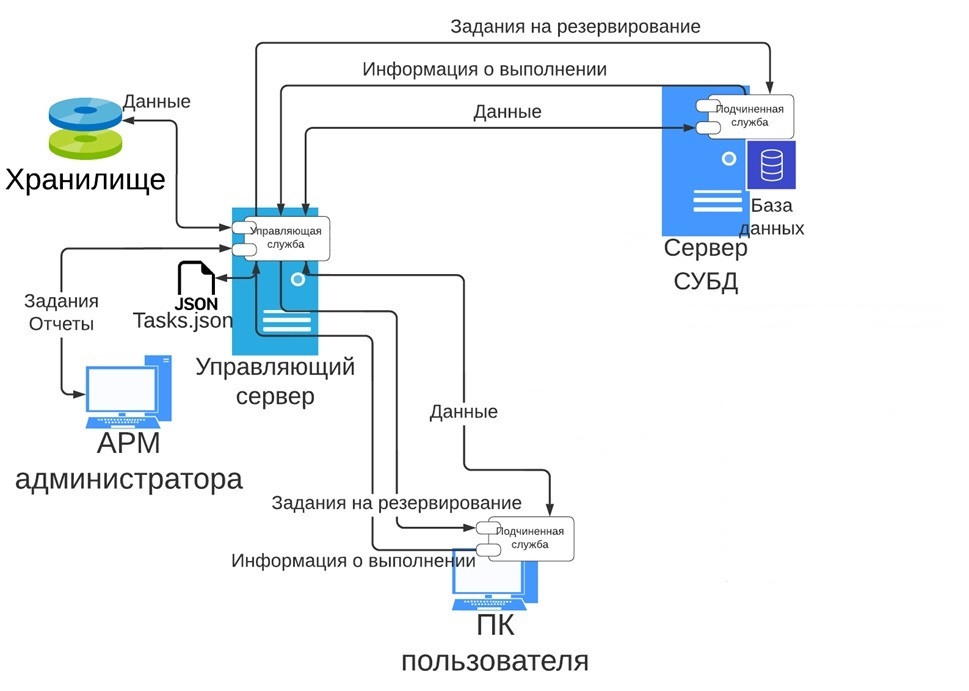


Рис. 4. Архитектура системы

Центральным звеном системы является управляющий сервер резервного копирования, на котором запущена управляющая служба. Управляющая служба прослушивает порт, который задается при установке службы (по умолчанию порт 1708). В качестве управляющего сервера можно использовать выделенный сервер или использовать сервер, уже задействованный под другие задачи (если вычислительные мощности позволяют).

К управляющему серверу подключено хранилище (например, RAID-массив или система хранения данных). При выполнении резервного копирования производится запись в хранилище, при восстановлении и проверке наличия резервных копий производится чтение из хранилища. Кроме того, периодически производится удаление устаревших резервных копий из хранилища.

На серверах и персональных компьютерах, данные которых следует резервировать, устанавливаются подчиненные службы (порт также задается при установке, по умолчанию также используется порт 1708). Подчиненные службы взаимодействуют только с управляющей службой, при этом от управляющей службы к подчиненным службам передаются *задания на резервирование* (информация о заданиях включает идентификатор, номер версии задания, информацию о резервируемом объекте и расписании). Управляющая служба передает подчиненным службам подмножества заданий, относящиеся к соответствующим подчиненным службам. От подчиненных служб к управляющей службе передается *информация о выполнении заданий (и произошедших при этом ошибках)*. Сами резервные копии передаются от подчиненной службы к управляющей при резервировании и в обратном направлении (от управляющей службы к подчиненным) при восстановлении.

Управление заданиями на управляющем сервере (и, следовательно, заданиями на подчиненных службах) выполняется из приложения, запускаемого на ПК системного администратора. Для удобства данная программа может быть установлена на нескольких ПК, также целесообразно для удобства установить ее на управляющий сервер. При подключении к управляющей службе производится идентификация и аутентификация пользователя (данные для идентификации и аутентификации хранятся на управляющем сервере). Из АРМ администратора можно создавать задания, просматривать и редактировать параметры созданные задания, просматривать информацию о результатах выполнения заданий, а также удалять задания. После изменения заданий информация об изменениях сразу отправляется на подчиненные службы. Информация о пользователе, выполнившем создание, изменение или удаление задания, сохраняется на сервере (действия пользователей записываются в лог, информация о пользователе, последним изменившим задачу, дополнительно сохраняется в самой задаче). Информация о задания хранится в файле Tasks.json.

## 2.3 Реализованные модули, классы и методы

При разработке приложения были реализованы следующие модули и классы:

Модуль ClientConfig. Библиотека для работы с параметрами подключения клиентских приложений.

Класс Config. Класс для работы с параметрами подключения клиентов.

Методы:

public Config() – конструктор по умолчанию.

public void SaveToFile(string filename) – Сохраняет параметры подключения в указанный файл в формате JSON.

public async Task SaveToFileAsync(string filename) – Асинхронно сохраняет параметры подключения в указанный файл в формате JSON.

public static Config LoadFromFile(string filename) – Загружает параметры подключения из указанного файла.

public async static Task<Config> LoadFromFileAsync(string filename) – Асинхронно загружает параметры подключения из указанного файла.

Модуль ClientService. Служба, работает на клиентском компьютере и выполняет задания по резервированию.

Класс WinService.

Методы:

public WinService() – конструктор по умолчанию.

protected override async void OnStart(string[] args) – асинхронный метод, выполняется при запуске службы.

protected override void OnStop() – выполняется при остановке службы.

private async Task Connect() – выполняет подключение к серверу.

private async void RecvHandler(SharedRequest packet, Connection connection) – обработчик пакетов полученных от сервера по сети.

private async Task Handler() – основной рабочий цикл службы. Проверяет задания и выполняет их.

Модуль DesktopClient. Графическое приложение для просмотра и управления заданиями.

Класс Globals. Содержит глобальные данные приложения и выполняет действия, не связанные с графическим интерфейсом.

Методы:

public static bool Init() – выполняет инициализацию приложения (запуск службы и подключение к серверу)

public static bool Connect() – подключение к серверу.

private static async void RecvHandler(SharedRequest packet, Connection connection) – обработчик входящих от сервера пакетов.

public static async void SendLogin(string username, string password) – отправляет пакет Login на сервер по сети.

public static async void SendTasks() – отправляет пакет tasks на сервер по сети.

public static async void SendRestore(int id) – отправляет пакет restore на сервер по сети.

public static void LoadConfig() – загружает параметры поключения из файла конфигурации.

public static void SaveConfig() – сохраняет параметры подключения в файл конфигурации.

Класс Login. Форма с параметрами подключения.

Методы:

public Login() – конструктор по умолчанию.

private void buttonConnection\_Click(object sender, EventArgs e) – обработчик нажатия на кнопку ‘Подключиться’.

public void Clear() – выполняет очистку полей с логином и паролем в форме.

private void Login\_Load(object sender, EventArgs e) – выполняется при загрузке формы

private void Login\_FormClosing(object sender, FormClosingEventArgs e) – выполняется при закрытии формы.

Класс Main. Основное окно приложения.

Методы:

public Main() – конструктор по умолчанию.

public void UpdateTable(TasksInfo tasks) – обновляет список заданий в окне согласно переданому параметру.

private void buttonAddFile\_Click(object sender, EventArgs e) – Открывает диалог добавления резервирования файла. Создает задание по резервированию.

private void buttonEdit\_Click(object sender, EventArgs e) – Открывает диалог для редактирования задания.

private void buttonDelete\_Click(object sender, EventArgs e) – Удаляет выбранное задание.

private void listView1\_SelectedIndexChanged(object sender, EventArgs e) – При изменении выбранного создания в списке активирует или деативирует часть кнопок.

private void Main\_Load(object sender, EventArgs e) – Выполняется при загрузке окна. Запускает диалог с формой Login.

private void buttonAddDb\_Click(object sender, EventArgs e) – Открывает диалог для добавления задания по резервированию БД.

private void buttonRestore\_Click(object sender, EventArgs e) – Отправляет на сервер команду для восстановления файла или БД.

private void buttonDisable\_Click(object sender, EventArgs e) – Включает или отключает выбранное задание.

private void buttonQuota\_Click(object sender, EventArgs e) – Открывает диалог настройки квоты.

private void buttonHistory\_Click(object sender, EventArgs e) – Открывает диалог отображения истории выбранного задания.

Класс Quota. Форма для настройки параметров квоты.

Методы:

public Quota() – конструктор по умолчанию. Заполняет поля формы необходимыми значениями.

Класс TaskDatabaseEdit. Форма для добавления и редактирования задания по резервированию БД.

Методы:

public TaskDatabaseEdit() – конструктор по умолчанию.

private void buttonConnect\_Click(object sender, EventArgs e) – обработчик нажатия на кнопку ‘Подключиться’. Выполняет соединение с БД и заполняет выпадающий список именами баз данных.

public DbBackupTask GetTask() – Создает класс задания из параметров полей в форме.

public void SetTask(DbBackupTask task) – Заполняет поля на форме параметрами из класса DbBackupTask.

Класс TaskFileEdit. Форма для добавления и редактирования задания по резервированию файлов.

Методы:

public TaskFileEdit() – конструктор по умолчанию.

private void buttonSelect\_Click(object sender, EventArgs e) – открывает стандартный диалог выбора файла.

public FileBackupTask GetTask() – Создает экземпляр класса FileBackupTask из полей формы.

public void SetTask(FileBackupTask task) - – Заполняет поля на форме параметрами из класса FileBackupTask.

Класс TaskHistory. Форма для отображения истории задания.

Методы:

public TaskHistory() – конструктор по умолчанию.

public void UpdateHistoty(BackupTask task) – заполняет параметры таблицы данными из истории задания.

Модуль ServerService. Служба сервера.

Класс Server.

Методы:

public async Task Listen() – принимает входящие подключения от клиентов.

async void LoadUsers() – загружает данные по пользователям из файла.

private async void HandlerCommand(SharedRequest packet, Connection connection) – обработчик входящий по сети пакетов.

void CheckBackups() – Проверяет папку с бэкапами. Удаляет лишние бэкапы.

long GetFolderSize(DirectoryInfo di) – вычисляет размер папки.

void SendLoginState(bool logged, Connection connection) – отправляет состояние авторизации по сети клиенту.

protected override async Task ExecuteAsync(CancellationToken stoppingToken) – выполняется при запуске службы.

protected internal void Disconnect() – выполняется при остановке службы.

Модуль SharedData. Содержит классы данных, которые пересылаются по сети между модулями.

Класс FilesInfo. Служит для передачи файлов по сети. Содержит список с информацией по файлам.

Методы:

public FilesInfo() – конструктор по умолчанию.

private FilesInfo(List<FileStruct> binFiles) – консруктор, создает экземпляр класса из списка файлов.

public void Add(int id, DateTime date, string nameFile, byte[] bin) – добавляет файл и информаци в список.

public byte[] ToArray() – преобразовывает список в массив байт для передачи по сети.

public static FilesInfo FromBin(byte[] bin) – создает экземпляр класса из массива байт.

Класс SharedRequest – класс для отправки запроса по сети.

Класс SharedResponse – класс для отправки ответа по сети.

Класс BackupTask – абстрактный класс задания по резервированию.

Методы:

public void UpdateNextBackupTime() – вычисляет и обновляет дату и время следующего бэкапа.

public string GetStatusString() – возвращает текстовую строку на основе статуса задания.

public string GetScheduleString() – возвращает текстовую строку на основе расписания задания.

public void AddAction(TaskAction action) – добавляет действие в историю задания.

Класс TaskInfo – содержит список всех заданий по резервированию.

Методы:

public byte[] ToArray() – преобразовывает содержимое класса в массив байт для передачи по сети.

public static TasksInfo FromArray(byte[] array) – создает экземпляр класса из массива байт.

public void SaveToFile(string filename) – сохраняет содержимое класса в файл в формате JSON.

public async Task SaveToFileAsync(string filename) – асинхронно сохраняет содержимое класса в файл в формате JSON.

public static TasksInfo LoadFromFile(string filename) – создает экземпляр класса на основе содержимого файла в формате JSON.

public int GetNextId() – возвращает Id для новой задачи.

В качестве базового класса для заданий определен абстрактный класс BackupTask. Данный класс содержит общие свойства для всех задач:

* числовой идентификатор задания (поле Id);
* адрес сервера, на котором находится подчиненная служба (поле Address);
* порт, на котором находится подчиненная служба (поле Port);
* дата и время последнего резервирования (поле LastBackupTime);
* дата и время следующего резервирования (поле NextBackupTime);
* тип бэкапа (поле TypeTimeBackup);
* статус бэкапа (поле Status, тип – перечисление TaskStatus);

Существуют следующие типы статуса (перечисление TaskStatus):

* TaskStatus.New – новое задание;
* TaskStatus.Working – задание выполняется;
* TaskStatus.Error\_NoFile – указанного файла нет.
* TaskStatus.Error\_DbConnect – ошибка соединения с сервером SQL.
* TaskStatus.Restored – восстановлено.

Для конкретных типов резервирования определены потомки базового класса. В настоящее время реализовано 2 типа резервирования: резервирование файлов (класс FileBackupTask) и резервирование базы данных SQL Server (класс DbBackupTask).

В случае добавления нового типа резервирования необходимо создать еще одного потомка для класса BackupTask, при этом общий функционал будет работать сразу.

Класс FileBackupTask (прямой потомок класса BackupTask) определяет задание резервирования файловой системы. В данном классе дополнительно определено поле Path – путь к резервируемому файлу или папке (если путь заканчивается на слеш, то это папка, иначе файл).

Класс DbBackupTask (также прямой потомок класса BackupTask) определяется задание резервирования базы данных SQL Server.В данном классе дополнительно определены следующие поля:

* Server – имя или ip-адрес сервера;
* DbName – название базы данных;
* Login – логин для доступа к базе данных;
* Password – пароль для доступа к базе данных.

Диаграмма классов заданий резервирования приведена на рисунке 5:

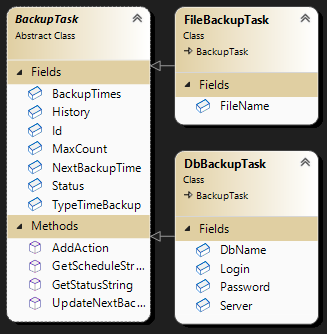


Рис. 5. Диаграмма классов заданий резервирования

В приложении 1 приведен основной код, отвечающий за передачу данных на сервер.

В приложении 2 приведен основной код, реализующий управляющую службу.

В приложении 3 приведен основной код, реализующий АРМ администратора.

В приложении 4 приведен основной код, реализующий подчиненную службу.

## 2.4 Установка программного обеспечения

Для установки системы необходимо выполнить установку следующих компонентов.

* Управляющая служба. Необходимо скопировать на управляющий сервер каталог ServerService и выполнить установку службы запуском командного файла MainService.bat из данного каталога.
* Подчиненная служба. Устанавливается на серверах и персональных компьютерах, данные которых необходимо резервировать. Необходимо скопировать на каждый из таких объектов каталог ClientService и выполнить установку службы запуском командного файла MainService.bat из данного каталога.
* АРМ администратора. Устанавливается на ПК системных администраторов и на управляющем сервере. Установка не требуется, необходимо скопировать каталог DesktopClient.

Удаление компонентов производится следующим образом:

* Управляющая служба. Для удаления необходимо от имени администратора в режиме командной строки выполнить команду sc delete ServerService

Далее следует удалить каталог ServerService.

* Подчиненная служба. Для удаления необходимо от имени администратора в режиме командной строки выполнить команду

sc delete ClientService

Далее следует удалить каталог ClientService.

* АРМ администратора. Для удаления необходимо удалить каталог DesktopClient.

## 2.5 Описание программного обеспечения

Управление заданиями выполняется из АРМ администратора. Для работы АРМ администратора необходимо, чтобы была запущена управляющая служба. Кроме того, на сетевом оборудовании и на брандмауэрах должно быть разрешено взаимодействие (отправитель – АРМ администратора, динамический порт, диапазон 49152 – 65535, согласно [4]; получатель – управляющий сервер, порт, настроенный при установке, по умолчанию порт 1708).

При запуске приложения откроется окно, изображенное на рисунке 6.



Рис. 6. Начальное окно программы

В поле «Сервер» необходимо указать имя или ip-адрес управляющего сервера, порт (по умолчанию 1708), логин и пароль. Если сервер или номер порта указаны неверно, будет выдано сообщение об ошибке (рисунок 7):

Изображение выглядит как текст

Автоматически созданное описание

Рис. 7. Сообщение об ошибке при подключении к серверу

Если возникнет ошибка при аутентификации, будет выдано следующее окно (рисунок 8):



Рис. 8. Сообщение об ошибке при вводе логина / пароля.

Если подключение выполнено успешно, значения, указанные в полях «Сервер», «Порт» и «Логин» сохраняются (при следующем запуске потребуется ввести только пароль).

В случае успешного соединения с сервером пользователь видит интерфейс со списком задач, полученный от управляющей службы. Интерфейс представлен на рисунке 9.

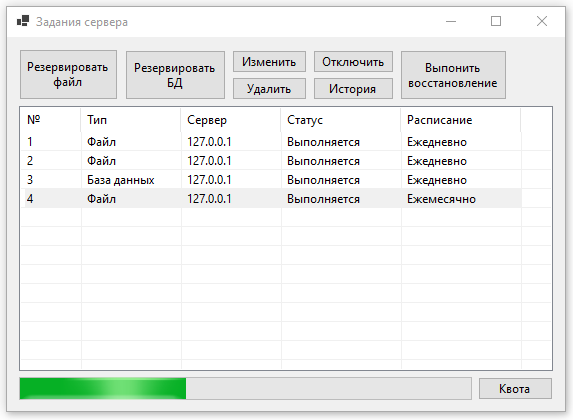


Рис. 9. Окно управления заданиями на резервирование

В данном окне можно создать, редактировать или удалить задание. В списке отображается тип задачи, адрес сервера, статус выполнения и расписание, где указан период выполнения.

Для создания задания необходимо нажать на кнопку создания соответствующего задания.

При создании задания резервирования файла нужно указать путь к файлу или каталогу, который будет сохраняться, также следует выбрать один из трех вариантов периодичности выполнения задания (ежедневно, еженедельно или ежемесячно) и указать время выполнения. Интерфейс создания и редактирования задачи изображен на рисунке 10.

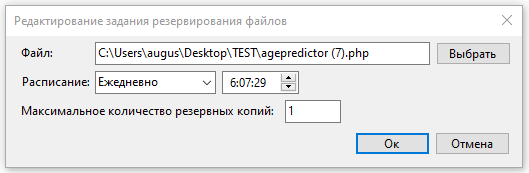


Рис. 10. Интерфейс создания и редактирования задачи файла.

В случае создания задания резервирования базы данных нужно ввести имя сервера, где находится необходимая для резервирования база данных, и указать имя пользователя, пароль. Если подключение успешно, то надпись о статусе соединения сменится и появится список баз данных. Интерфейс создания и редактирования задачи изображен на рисунке 11. Также есть настройка количества хранимых копий.

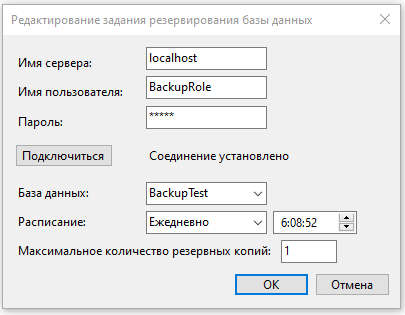


Рис. 11. Интерфейс создания и редактирования задачи базы данных.

Для восстановления базы данных или файла нужно выбрать из списка задачу и нажать кнопку восстановления.

Доступен просмотр истории действий с заданием по резервированию, рисунок 12 и отключение задания.

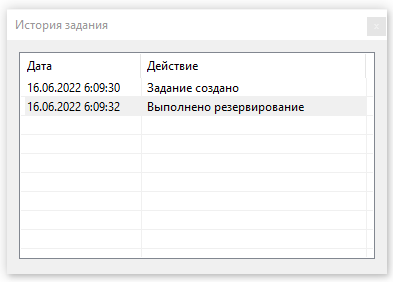


Рис. 12. Просмотр истории действий с заданием.

Также есть возможность управления квотой резервируемых данных на сервере, рисунок 13.

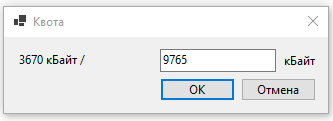


Рис. 13. Редактирование допустимого объема для резервирования.

Вывод по главе

На основе информации, рассмотренной в теоретической части данной работы была разработана модель клиент-серверного приложения. На основе модели представлен программный комплекс, разработанный при выполнении дипломной работы.

# ЗАКЛЮЧЕНИЕ

В ходе дипломной работы были решеные все задачи, необходимые для создания клиент-серверного приложения резервирования и восстановления данных:

* Изучены технологии резервирования данных;
* Сформулированы требования к разрабатываемой системе;
* Спроектирована разрабатывается система;
* Разработано программное обеспечение, реализующее резервирование и восстановление наиболее востребованных типов источников (файлы, базы данных SQL server) и позволяющее добавлять новые типы источников;

Такие образом, цель дипломной работы достигнута.

# СПИСОК ЛИТЕРАТУРЫ

1. Работа с библиотекой Network // [Электронный ресурс]. URL: https://github.com/Toemsel/Network (дата обращения: 14.05.2022).
2. Работа с библиотекой Newtonsoft.Json на реальном примере // [Электронный ресурс]. URL: <https://habr.com/ru/post/481514/> (дата обращения: 14.05.2022).
3. Виды резервирования // [Электронный ресурс]. URL: https://www.sim-networks.com/ru/blog/backup-full-increment-differential (дата обращения: 21.05.2022).
4. The default dynamic port range for TCP/IP has changed since Windows Vista and in Windows Server 2008 // [Электронный ресурс]. URL: <https://docs.microsoft.com/en-us/troubleshoot/windows-server/networking/default-dynamic-port-range-tcpip-chang> (дата обращения: 01.06.2022).
5. Процесс в C# (класс Process) // Интернет портал dir.by [Электронный ресурс]. URL: <https://dir.by/developer/csharp/process/> (дата обращения: 18.05.2022).
6. Абидарова А. А. Резервное копирование и хранение данных [Электронный ресурс]. — URL: https://cyberleninka.ru/article/n/rezervnoe-kopirovanie-i-hranenie-dannyh (дата обращения: 12.06.2022).