НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
март–апрель 2025 Том 25 № 2 http://ntv.ifmo.ru/
SCIENTIFIC AND TECHNICAL JOURNAL OF INFORMATION TECHNOLOGIES, MECHANICS AND OPTICS
March–April 2025 Vol. 25 No 2 http://ntv.ifmo.ru/en/
ISSN 2226-1494 (print) ISSN 2500-0373 (online)

КОМПЬЮТЕРНЫЕ СИСТЕМЫ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

COMPUTER SCIENCE

doi: 10.17586/2226-1494-2025-25-2-253-260
УДК 004.032.26

Разработка файловой системы для хранения данных интеллектуальной системы видеонаблюдения

Алексей Николаевич Субботин[1]✉, Наталия Александровна Жукова[2]

[1] Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина), Санкт-Петербург, 197022, Российская Федерация
[1] ГУП «Петербургский метрополитен», Санкт-Петербург, 190013, Российская Федерация
[2] Санкт-Петербургский федеральный исследовательский центр Российской академии наук (СПб ФИЦ РАН), Санкт-Петербург, 199178, Российская Федерация

[1] alesu1543@gmail.com✉, https://orcid.org/0000-0002-4823-6288
[2] nazhukova@mail.ru, https://orcid.org/0000-0001-5877-4461

Аннотация

Введение. Рассмотрена проблема создания файловой системы с характеристиками, отличными от универсальных, для хранения данных интеллектуальных систем видеонаблюдения. Доступ к файловой системе является определяющим фактором, от которого зависит быстродействие всей системы. Скорость выполнения операций с данными определяется не только быстротой работы шины данных и наличием современного процессора, но и драйвером доступа к жесткому диску, который может ограничивать возможности системы выполнять основные функции: наблюдение, анализ изображений, определение образов и событий. Существует потребность в использовании более производительного сервера, что требует дополнительных материальных затрат, в разработке специализированной файловой системы для повышения скорости записи и чтения с жесткого диска. Использование специализированной файловой системы, ориентированной на решение одной или ограниченного числа задач, может значительно повысить скорость работы системы при использовании серверов с одинаковыми техническими характеристиками. В интеллектуальных системах видеонаблюдения применение специализированной файловой системы может обеспечить повышение скорости обработки изображений и точности определения объектов в видеопотоке благодаря повышению скорости записи и чтения данных с диска. Анализ существующих файловых систем показал, что имеющиеся решения не позволяют обеспечить требуемую скорость работы с данными в интеллектуальных системах видеонаблюдения при наличии ограничений на количество и мощность используемых технических средств. Предложена специализированная файловая система для хранения данных систем интеллектуального видеонаблюдения. **Метод.** Разработана файловая система с описанием, ориентированная на решение задачи хранения данных в системах интеллектуального наблюдения. Используемая база данных обладает функциями чтения, поиска, записи и обновления структурированных данных, размещенных в ее таблицах. База данных оптимизирована для работы с данными интеллектуальных систем видеонаблюдения, имеет ограниченную размерность столбцов, задаваемую в соответствии с размещаемыми в ней сущностями. Особенность база данных состоит в том, что она постоянно находится в оперативной памяти, синхронизация данных с жестким диском выполняется через заданный интервал времени. Примененная база данных, подобно Redis, работает значительно быстрее традиционных. Разработанный драйвер действует напрямую с жестким диском, не использует функции операционной системы, что повышает скорость работы с данными. **Основные результаты.** Сравнение скорости записи и чтения данных при использовании разработанного и существующих универсальных драйверов показало, что применение нового драйвера позволяет повысить скорость записи и чтения на 43,4 % относительно New Technology File System (NTFS). **Обсуждение.** В рамках проведенного исследования выполнена разработка файловой системы для интеллектуальных систем видеонаблюдения. Отмечено, что подобные специализированные файловые системы могут разрабатываться для применения в других областях, где требуется повысить скорость (снизить время) записи и чтения данных с диска.

Ключевые слова

файловая система, интеллектуальная система видеонаблюдения, хранение данных, взаимодействие с объектной базой данных, снижение времени доступа к данным, база данных

Благодарности
Исследование выполнено при финансовой поддержке государственного бюджета, номер проекта № FFZF-2025-0019.

Ссылка для цитирования: Субботин А.Н., Жукова Н.А. Разработка файловой системы для хранения данных интеллектуальной системы видеонаблюдения // Научно-технический вестник информационных технологий, механики и оптики. 2025. Т. 25, № 2. С. 253–260. doi: 10.17586/2226-1494-2025-25-2-253-260

Development of a file system for storing data of an intelligent video surveillance system

Alexey N. Subbotin[1]✉, Nataly A. Zhukova[2]

[1] Saint Petersburg Electrotechnical University "LETI", Saint Petersburg, 197022, Russian Federation
[1] State Unitary Enterprise "Petersburg Metropolitan", Saint Petersburg, 190013, Russian Federation
[2] St. Petersburg Federal Research Center of the Russian Academy of Sciences, Saint Petersburg, 199178, Russian Federation

[1] alesu1543@gmail.com✉, https://orcid.org/0000-0002-4823-6288
[2] nazhukova@mail.ru, https://orcid.org/0000-0001-5877-4461

Abstract
The article considers the problem of creating a file system with characteristics different from universal ones for storing data of intelligent video surveillance systems. Access to the file system is a determining factor on which the performance of the entire system depends. A fast data bus and a modern processor do not always determine the speed of data operations, but also the hard disk access driver which, accordingly, can limit the system ability to perform basic functions: surveillance, image analysis, detection of images and events. It is necessary to select a more productive server which is expensive, or use a specialized driver to increase the speed of writing and reading on the hard disk. The use of a specialized file system focused on solving one or a limited number of problems can significantly increase the speed of systems in cases where the server is used with the same technical characteristics. In intelligent video surveillance systems, the use of a specialized file system can provide an increase in the speed of image processing and the accuracy of object detection in the video stream, due to the increased speed of reading and writing from the disk. An analysis of existing file systems has shown that the existing solutions do not provide the required speed of working with data in intelligent video surveillance systems when using technical means with the same computing characteristics. In this article, the authors propose a specialized file system for storing data in intelligent video surveillance systems. A file system has been developed that is focused on solving one problem: storing data in intelligent surveillance systems. The developed driver increases the speed of accessing the data on the hard drive. The new file system for storing data in an intelligent video surveillance system works together with a database for one, separate task. A comparison of the speed of writing and reading data using the developed driver and using existing universal drivers made. As a result of the comparison, it has been established that the use of the new driver has increased the speed of writing and reading by 43.4 % relative to NTFS file system. As part of the study, a file system for intelligent video surveillance systems was developed, but similar specialized file systems can be developed for use in other areas where it is necessary to increase the speed (reduce the time) of writing and reading data from the file system.

Keywords
file system creation, intelligent video surveillance system, data storage, interaction with object database, reduction of data access time, database, file system

Acknowledgements
This work was supported by the state budget, project No. FFZF-2025-0019.

For citation: Subbotin A.N., Zhukova N.A. Development of a file system for storing data of an intelligent video surveillance system. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2025, vol. 25, no. 2, pp. 253–260 (in Russian). doi: 10.17586/2226-1494-2025-25-2-253-260

Введение

Хранение последовательности файлов, как единого потока данных, приводит к значительным временным затратам при перезаписи информации и добавлении новых данных в ранее созданные файлы. Исходя из этого, все современные файловые системы поддерживают кластеризацию, которая предусматривает условное разделение на фрагменты дискового пространства для быстрого добавления новых данных в файлы. Первые сектора жесткого диска заняты таблицей с информацией о файлах. Информация об одном файле занимает 1 КБ в таблице New Technology File System (NTFS), где содержится дата и время файла, занимаемые кластеры, длинное имя и другая служебная информация.

Изначально драйвер для работы с файловой системой был отдельной программой, поставляемой с устройством многократной записи. Но в последствии корпорацией IBM был создан универсальный драйвер FAT для всех жестких дисков и интегрирован в ядро операционной системы DOS (Disk Operating System). Такой подход применяется и в ядре Linux, где интегрированы файловые системы: Ext2, Ext3, Ext4, JFS, ReiserFS, XFS, Btrfs. Практически все современные операционные системы: Windows, FreeBSD, macOS, Linux, OpenSolaris, QNX и др. считают драйвер файловой системы частью ядра. Такой подход необходим для чтения служебных файлов операционной системы после передачи управления из сектора загрузчика в сектор операционной системы. После команды от загрузчика

система проверяет базовые порты записи/чтения (Input/Output, I/O) по спецификации процессора x86, загружает драйвера файловой системы, читает свои файлы, переопределяет порты I/O и проверяет работоспособность оборудования [1, 2]. Далее проверяется допустимость загрузки по техническим характеристикам: частота процессора, размер оперативной памяти и жесткого диска. Файловая система имеет большое значение на этапе загрузки операционной системы. Однако нестандартные файловые системы распространены и в наше время, поскольку современные файловые системы перегружены дополнительными функциями и сильно уступают простому бинарному чтению в стеке с XXX по YYY адреса. Не всегда необходимо журналирование и разбиение файлов на кластеры, а таблица разметки диска может быть реализована в самом драйвере (программе для доступа к данным в одном разделе). Раздел с любой файловой системой, созданной Linux, Windows, FreeBSD можно просто удалить, а данные читать своей программой с адреса сепарации до конечного адреса неразмеченного пространства. По такому принципу работают программы Norton Recovery Files[1], которые восстанавливают удаленные файлы из корзины Windows.

В представленной работе показаны пути повышения скорости чтения и записи данных в интеллектуальной системе видеонаблюдения (ИСВН) за счет использования нестандартной файловой системы. Это требует решения таких задач, как разработка файловой системы для хранения данных, драйвера доступа к файловой системе, который за счет непосредственной работы с жестким диском позволит повысить скорость работы. Проведена экспериментальная оценка скорости доступа к данным.

Рассмотрение аналогов

Исследования, направленные на создание файловой системы в Computer Science, ведутся с 60-х годов XX века, когда появилась первая операционная система GM-HAA и Multics-MIT. Так, многие трояны и шифровальщики по данным Лаборатории Касперского[2] используют неразмеченное дисковое пространство и создают свои скрытые разделы для хранения файлов. Но данный подход может быть применен и на пользу. Например, брендовые компьютеры (Acer, Asus, Dell, Apple и пр.) применяют скрытый файловый раздел в начале диска для восстановления оригинальной версии операционной системы со всеми драйверами, настройками производителя и фирменными программами [3–5]. Раздел не виден операционной системе, но он может быть прочитан специальной программой с адреса дискового пространства XXX до YYY. Однако вирусы научились читать этот восстановочный раздел и дописывать туда свой код и подменять файлы. После восстановления изначальная копия операционной системы уже содержит свежие вирусы.

Одним из примеров систем, в которых используются специализированные файловые системы, являются системы, разрабатываемые для технологических встраиваемых компьютеров, где сильно ограничены вычислительные ресурсы (Raspberry Pi, ASUS Tinker Board S, ClockworkPi, Arduino Mega 2560 и пр.). Отсутствие мощного процессора и современного чипсета не позволяет быстро передавать данные между жестким диском и регистрами процессора, что необходимо для выполнения таких функций, как просмотр видео. В камерах видеонаблюдения Axis[3] есть встроенное программное обеспечение P-Iris, где тоже используется специальная файловая система для повышения скорости работы с данными через процессоры: ARTPEC-3 и ARTPEC-B.

В настоящее время специализированные файловые системы используются в программных системах VMware, где применяется специализированная кластерная файловая система VMFS для хранения данных виртуальных машин. Широко распространена файловая система для хранения данных на оптических дисках CDfs, файловая система Cramfs, обеспечивающая возможность сжатия данных при записи в операционной системе GNU/Linux. Как сервисная функция используется менеджер логических томов и файловая система ZFS от Sun Microsystems без фрагментации с изменением томов как у NTFS.

Анализ широкого круга существующих ИСВН, показал, что в них не используются специализированные файловые системы. В процессе анализа рассматривались такие популярные системы видеонаблюдения как Видеоинтеллект, ISS, МТС Облачное видеонаблюдение, Ситроникс, Бевард, Ростелеком, Максима и многие др., предоставляющие широкие возможности по распознаванию образов с использованием различных профилей для определения номеров машин на автостоянке и других профильных объектов, например, детского садика, школы и т. д. Все средства предоставляют возможности для динамического анализа видеоряда и позволяют выполнять аналитику предиктивного характера. В имеющихся ИСВН (например, от производителя IVS) обеспечивается сохранение статистической информации на видеосервер, для доступа к данным используются драйвера по умолчанию, поддерживаемые операционной системой NTFS. Это приводит к замедлению обращений к диску почти в два раза. До настоящего времени специализированные файловые системы и драйвера для ИСВН не создавались, поскольку объемы обрабатываемых данных были меньше и возможностей имеющихся технических средств было достаточно для их обработки. Создание специализированной файловой системы позволит обеспечить повышение скорости (снижения времени) доступа к данным.

Определение проблемы

ИСВН характеризуются высокой интенсивностью чтения и записи с дискового хранилища, которое может быть представлено как озеро данных или как диско-

[1] [Электронный ресурс]. https://www.norton.com/ (дата обращения: 22.03.2025).

[2] [Электронный ресурс]. https://www.kaspersky.ru/ (дата обращения: 22.03.2025).

[3] [Электронный ресурс]. https://www.axis.com (дата обращения: 22.03.2025).

вое пространство на сервере в облачной платформе. Данные от ИСВН содержат как статистическую информацию, так и видеоряд, по которому можно выявлять объекты и события и их классифицировать. К статистической информации относится: дата и время работы ИСВН, наблюдаемые объекты (цеха, столовые, коридорные и другие помещения общего назначения), место расположения видеокамер, их характеристики, погодные (туман, освещенность, время дня и ночи и пр.) и температурные условия. Хранимая информация используется операторами ИСВН на устройствах визуализации событий и просмотра хроники для определения чрезвычайных ситуаций на объектах (метрополитен, стадион, аэропорт, магазин, детский садик, школа и др.) в зависимости от профиля деятельности учреждения и поставленных задач перед отделом безопасности (мелкое хулиганство, воровство, террористическая угроза и пр.). В таких системах доступ к данным требует много времени (скорость доступа к данным варьируется от 10 до 24 кадров/с с фрагментами видео и статистической информацией в зависимости от настроек системы).

Современные видеосерверы позволяют выполнять значительное количество операций в секунду, но для повышения их производительности требуется изменение технических характеристик и больших затрат. Разработка специализированной программы (драйвера) для работы с данными для определенной задачи, в частности, интеллектуального видеонаблюдения может позволить значительно повысить скорость доступа к данным на жестком диске (рис. 1). При разработке специализированной файловой системы необходимо точно определить размерностью файлов, которые будут храниться. Файловая система должна быть создана как логическое продолжение базы данных для ИСВН [6, 7].

Специализированный драйвер должен разрабатываться как часть системы, взаимодействующей со своей базой данных в виде объектов, которые считываются драйвером с жесткого диска. При этом дальнейшая работа с данными ИСВН осуществляется в оперативной памяти видеосервера.

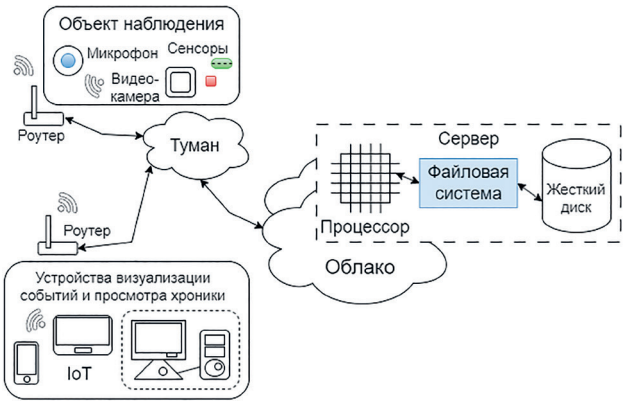


Рис. 1. Место файловой системы на видеосервере в интеллектуальной системе видеонаблюдения.

IoT — Internet of Things

Fig. 1. The place of the file system on the video server in the intelligent video surveillance system

Постановка задачи

Современные файловые системы, такие как FAT32, NTFS, EXT4 и многие другие наделены огромным количеством функций и большой универсальностью, позволяющей решать любые задачи. Однако если отказаться от универсальности, а разработать драйвер (программу) файловой системы под одну задачу, то скорость доступа к данным может возрасти значительно. Файловая система, созданная под одну задачу несравнимо быстрее, чем универсальный драйвер. Стандартный (универсальный) драйвер файловой системы проходит следующие шаги: WinAPI, контроллер, файловая таблица, кластеризация, адресация и пр. Данные шаги значительно увеличивают время выполнения главной функции I/O в ИСВН. Создав свой драйвер для ИСВН, можно значительно повысить скорость (снизить время) доступа к данным на жестком диске без изменения технических характеристик видеосервера и сэкономить технические ресурсы на обработку данных с видеокамер [8–10].

Таким образом, для обеспечения высокой скорости чтения и записи данных в ИСВН требуется решить следующие задачи:

— создать программу для работы с жестким диском (драйвер файловой системы);
— разработать базу данных для хранения информации ИСВН;
— выполнить интеграцию драйвера с ИСВН через объекты и методы базы данных;
— обеспечить взаимную совместимость трех компонентов: ИСВН, базы данных, файловых систем;
— разработать программы для оценки скорости под разные операционные системы.

Типовая задача разработки специализированного драйвера для ИСНВ может быть сформулирована следующим образом. Необходимо хранить статистическую информацию, серию кадров с камер видеонаблюдения (10–24 кадров/с с разрешением 4k и размерностью 3840 × 2160 пикселов и глубиной dpi: 450 с цветностью sRGB), видеофрагменты (25–35 МБ в формате MPEG/AVI и длиной 5–7 с), информацию о найденных объектах и событиях (одна строка с 31 ячейкой: дата, время, погодные условия, освещенность, количество объектов, точность определения в процентах, предполагаемое событие, характеристики видеокамеры, время года, время суток, день недели и информацию об объекте: цех, проходная, столовая, раздевалка и т. д.). Технические характеристики видеосервера: 2x Intel Xeon Gold gen3, 32 ГБ ОЗУ, 15x LAN 1 Гбит/с, 250GB SSD 2.5″ SATA, СХД — HDD 10TB Ent 7.2k SATA, IPMI 2.0, подключение 2 мониторов (1x HDMI, 1x DVI/HDMI), видеокарта nVidia GT 2Gb, операционная система Windows 11 IoT Enterprise High End[1] с поддержкой популярных программ: ITV/AXXON, Лини, AXIS, Macroscop, Milestone, TRASSIR. При использовании универсального драйвера ИСВН с рассмотренной конфигурацией позволяет выполнять сбор и обработку данных не более, чем с 12 видеокамер в непрерывном

[1] [Электронный ресурс]. https://www.videomax.ru/production/videomax_ip/ (дата обращения: 22.03.2025).

режиме. Разрабатываемый драйвер должен обеспечивать запись и параллельное чтение в постоянном режиме 100 Мбит/с при использовании одной видеокамеры. За счет создания нового специализированного драйвера должна быть обеспечена возможность одновременного подключения в режиме записи данных не менее, чем 20 видеокамер в непрерывном режиме.

Описание разработанной файловой системы

Для создания специализированной файловой системы предполагается использовать базу данных SubBase, которая разработана непосредственно для хранения информации ИСВН. База данных SubBase значительно быстрее аналогов и работает по принципу Redis[1], когда вся информация находится в оперативной памяти и периодически изменения сбрасываются на жесткий диск в MongoDB или любую другую документ-ориентированную системы управления базами данных (СУБД).

База данных ИСВН SubBase имеет ячеистую структуру и хранит информацию в виде таблиц, в которых размещаются данные определенного назначения в зависимости от их типа (дата, строка, текст, бинарное значение). Имеются таблицы в SubBase, которые хранят бинарные данные в формате BSON: серии кадров (JPEG/PNG), видеофрагменты (AVI/MPEG4), аудиопотоки в момент детекции объектов и событий, которые были на их основе определены. База данных (рис. 2) представлена в графическом виде в адресном пространстве жесткого диска, который имеет начальный и конечный адреса (условно: 00000001 и 00FFFFFF). Свободное дисковое пространство резервируется от адреса 00000XXX до адреса 00000YYY с емкостью SSS. Отличительной особенностью разработанной базы данных SubBase является жесткая и стабильная структура, состоящая из таблиц фиксированной длины, количеством N в двух копиях. Ячейки жестко ограничены в каждой таблице, имеют последовательность при записи данных, что позволяет достичь высокой скорости данных I/O в ИСВН.

ИСВН может работать с базой данных любого типа (SQL/NoSQL), но быстрее всего она работает с профильной базой SubBase с оптимизациями на уровне жесткого диска (MyISAM/InnoDB).

Обычно файловая система требует неразмеченное пространство на жестком диске [11–13], созданное утилитами для создания и управления разделами диска (Fdisk, GNU Parted, GParted, GNOME Disks, KDE Partition Manager и пр.). Однако предлагаемая файловая система SubSysVideo не требует дополнительных действий от разработчика. Удаление раздела и создание скрытого раздела не обязательно. Достаточно создать файл с условным названием «fileSubSysVideo.sys» в скрытом виде, только для чтения, и зафиксировать без возможности записи и чтения через WinAPI в NTFS. Стандартный драйвер Windows не будет получать доступа к файлу с размерностью SSS и не станет дефрагментироваться, что очень важно. Это необходимо,

Рис. 2. Графическое представление базы данных SubBase

Fig. 2. Graphical representation of the SubBase database

чтобы начальный и конечный адреса файла на жестком диске оставались неизменными. Далее запускается программа на языке программирования C++, скомпилированная на GPP (GNU) и читает побайтово дисковое пространство файла от X с размерностью SSS. Размер файла должен равняться двум копиям развернутой базы данных SubBase-v.254s в 4 ГБ. Доступ к данным ИСВН [14–16] получает через объект, которым является вся база данных SubBase. Объект управления SubBase — подобъект SubAction, а для работы с таблицами и строками — подобъект SubTable. Подобъект базы данных — новый объект, созданный с использованием метода «CreateObj();» главного объекта SubBase. У подобъекта SubAction имеются методы: Open, CopyBack, Create, Story и многие другие, которые позволяют копировать базу данных на жесткий диск из оперативной памяти. Методы: Insert, Update, Delete подобъекта SubTable (который создается через методы объекта SubAction) работают непосредственно с записями таблицы и обеспечивают: добавление новых записей, обновление существующих записей, удаление записей из одной таблицы. Используется размерность файловой системы 8 ГБ (4 ГБ + 4 ГБ) для основной базы данных, ко-

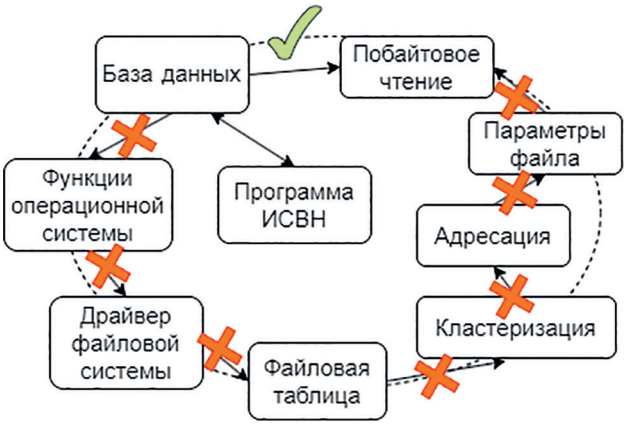


Рис. 3. Схема доступа к данным интеллектуальной системы видеонаблюдения (ИСВН) при использовании разработанной файловой системы

Fig. 3. Scheme of access to data of the intelligent video surveillance system using the developed file system

[1] [Электронный ресурс]. https://redis.io (дата обращения: 22.03.2025).

торая сохраняется на жесткий диск один раз в час, и резервной копии, которая копируется один раз в сутки. Вследствие создания резервных копий базы данных SubBase обеспечивается надежность работы, за счет разработки нового драйвера файловой системы под условным названием SubSysVideo-1.9.17fs достигается высокая скорость работы.

На рис. 3 схематично изображено предлагаемое решение, позволяющее решить проблему низкой скорости доступа к файловой системе применительно к одной задаче ИСВН. На схеме ИСВН изображена по центру и работает с базой данных.

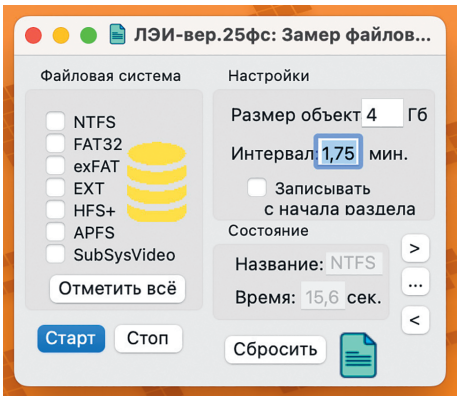
Экспериментальное исследование разработанной файловой системы

В рамках настоящей работы разработано приложение для MacOS Sequoia v.15, которое позволяет измерить скорость чтения и записи I/O данных в разных файловых системах. В ходе эксперимента проводилось копирование одного объекта (размером 4 ГБ) из оперативной памяти на жесткий диск, затем чтение с жесткого диска (обратная загрузка одного объекта в оперативную память). Для этого жесткий диск SSD размером 256 ГБ был разделен на два раздела: 154 ГБ под операционную систему и по 24 ГБ под каждую файловую систему. Чтение и запись объекта из RAM в раздел 24 ГБ выполнено последовательно через каждые 1,75 мин и с начала раздела, чтобы избежать несогласованности нитей данных.

Интерфейс приложения для измерения времени доступа (рис. 4) компилируется в кроссплатформенной среде Lazarus IDE под все популярные операционные системы (Windows 64k, Linux, QNX, MacOS, BSD). Использованы стандартные библиотеки для работы с файловой системой из индикативной среды разработки Lazarus. В инсталляторе IDE всегда присутствует модуль StrUtils, который предоставляет методы: Assign(subfs), Rewrite(subfs, pin), Write(subfs, str), Close(subfs) для работы с типом данных «file of integer». На рис. 4 представлен перечень файловых систем с возможностью отметить все настройки, которые позволяют указать размер объекта (4 ГБ), интервал между замерами (по умолчанию — 1,75 мин), отметить возможность записывать с начала раздела; состояние (название текущей файловой системы и время доступа). Правее от блока состояния показаны кнопки со стрелками для просмотра статистики по всем файловым системам, а по центру — возврат на первый пункт NTFS. Допустим сброс настроек в первоначальное значение (по кнопке «Сбросить»). Внизу размещены кнопки управления: «Старт» и «Стоп» для процесса проверки.

Проведены измерения скорости записи и чтения данных в пяти host операционных системах: Windows 11 (техническая версия сборки: 10.0.22631.4169), MacOS Sequia 15, FreeBSD 14.1, QNX Neutrino 7.1 и Debian 12 Linux 6.7.6. При сравнении рассматривались самые популярные файловые системы: NTFS, FAT32, exFAT, EXT, HFS+, APFS и созданная в настоящей работе — SubSysVideo-1.9.17fs. После проведения измерений (таблица) выяснилось, что предложенная файловая

a



b

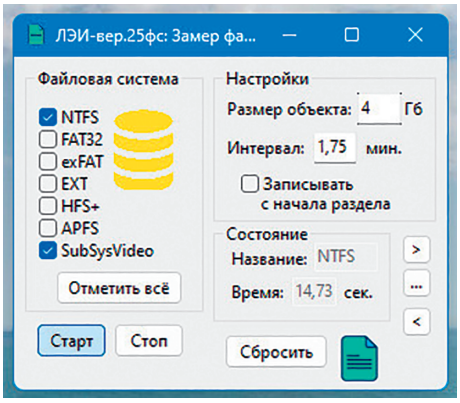


Рис. 4. Интерфейс приложения для измерения времени доступа к данным под MacOS (*a*) и Windows (*b*)

Fig. 4. Interface of the application for measuring data access time under MacOS (*a*) and Windows (*b*)

система работает быстрее на 43,4 % относительно NTFS.

Для воспроизведения эксперимента необходимо установить Python v.3.12.7pre под Mac OS X[1] и использовать приложение[2] для записи и чтения одного объекта из оперативной памяти в начало любого раздела. Также необходимо подготовить отдельные разделы по 24 ГБ, отформатированные в разных файловых системах. Для проведения измерений применялся компьютер Mac с техническими характеристиками: Apple MacBook Air (Z12700036), 13.3″, (WQXGA), 256 ГБ SSD (450–550 МБ/с), 7-core GPU M1, Mac OS Sequia, Silver. Результаты, полученные в таблице и в эксперименте, показали, что больше всего дополнительных функций находятся в файловой системе NTFS при работе из FreeBSD 14.1, что обусловлено второстепенностью драйвера NTFS под BSD систему. NTFS обладает огромным количеством дополнительных функций от главного разработчика Microsoft, относительно других файловых систем: FAT32, exFAT, EXT, HFS+, APFS и др. Быстрее всего работает файловая система EXT, поскольку обладает минимумом функций. Отметим, что в

[1] [Электронный ресурс]. https://www.python.org/downloads/macos/ (дата обращения: 22.03.2025).

[2] [Электронный ресурс]. https://github.com/alex1543/practPythonServ/tree/main/mac (дата обращения: 22.03.2025).

Таблица. Время доступа к данным на запись и чтение (один объект в озере данных — 4 ГБ), с
Table. Read and write access time (one object in the data lake — 4 GB), s

Название файловой системы	Операционная система					
	Windows 11	MacOS Sequia 15	FreeBSD 14.1	QNX Neutrino 7.1	Debian 12 Linux 6.7.6	Увеличение по сравнению с SubSysVideo-1.9.17fs, %
NTFS	14,7	15,6	15,8	15,7	15,5	**43,4**
FAT32	11,9	12,8	13,1	12,9	12,7	30,9
exFAT	12,6	13,7	13,9	13,8	13,3	35,1
EXT	9,7	10,3	11,2	10,7	10,1	15,7
HFS+	13,8	14,4	14,5	14,5	13,9	38,6
APFS	12,7	13,1	13,7	13,4	12,8	33,4
SubSysVideo-1.9.17fs	8,4	8,7	9,1	8,9	8,6	—

настоящей работе удалось ускорить функции записи и чтения в SubSysVideo-1.9.17fs на 15,7 % относительно самой быстрой файловой системы EXT и на 43,4 % относительно самой медленной файловой системы NTFS от Microsoft.

Заключение

Разработана файловая система, ориентированная на решение одной задачи — хранение данных интеллектуальной системы видеонаблюдения. Выполненный анализ позволил выявить, что время доступа к данным на жестком диске может быть значительно сокращено за счет исключения универсальных функций, реализуемых современными файловыми системами. Предложена файловая система для хранения данных интеллектуальной системы видеонаблюдения и база данных для хранения ее информации, разработан драйвер, который работает напрямую с жестким диском, не использует функции операционной системы, что повышает скорость работы с данными.

Проведенные экспериментальные исследования показали, что скорость доступа к данным повысилась на 43,4 % относительно файловой системы NTFS, которая оказалась самой медленной файловой системой из-за использования большого количества сервисных функций.

Разработка специализированных файловых систем, предназначенных для решения одной или нескольких задач, может быть востребована и в других областях деятельности. Например, такие файловые системы могут использоваться для хранения данных цифровых двойников, систем статистической обработки и визуализации информации, где необходимо обеспечить высокую скорость доступа к данным жесткого диска. Специализированные файловые хранилища могут быть разработаны не только для доступа к данным, размещенным на жестких дисках, но и на других носителях: flash-накопителях, оптических дисках, лентах и прочих устройствах хранения данных.

Литература

1. Avhad A.R., Gangad P.S., Kharote S.M., Muntode S.S., Sanap M.D. Blockchain based secure file transfer system with password protection // International Journal of Advanced Research in Science, Communication and Technology (IJARSCT). 2024. V. 4. N 2. P. 299–303. https://doi.org/10.48175/ijarsct-19651
2. Panjuta D., Jabbarzadeh J. C# File System // Learning C# Through Small Projects. Springer, 2024. P. 167–198. https://doi.org/10.1007/978-3-031-51914-7_6
3. Needhi J., Prasath R., Vikram K.K., Vishnu G. Performance optimization of voice-assisted file management systems // International Journal of Engineering and Computer Science. 2024. V. 13. N 7. P. 26250–26256. https://doi.org/10.18535/ijecs/v13i07.4854
4. Cho K., Bahn H. A lightweight file system design for unikernel // Applied Sciences. 2024. V. 14. N 8. P. 3342. https://doi.org/10.3390/app14083342
5. Gui J., Wang Y., Shuai W. Improving reading performance by file prefetching mechanism in distributed cache systems // Concurrency and Computation: Practice and Experience. 2024. V. 6. N 22. P. e8215 https://doi.org/10.1002/cpe.8215
6. Yuliana M., Hidayah N., Sudarsono A. Implementation of Web-Based file sharing security system // MOTIVECTION: Journal of Mechanical, Electrical and Industrial Engineering. 2024. V. 6. N 1. P. 41–52. https://doi.org/10.46574/motivection.v6i1.314

References

1. Avhad A.R., Gangad P.S., Kharote S.M., Muntode S.S., Sanap M.D. Blockchain based secure file transfer system with password protection. *International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)*, 2024, vol. 4, no. 2, pp. 299–303. https://doi.org/10.48175/ijarsct-19651
2. Panjuta D., Jabbarzadeh J. C# File System. *Learning C# Through Small Projects*. Springer, 2024, pp. 167–198. https://doi.org/10.1007/978-3-031-51914-7_6
3. Needhi J., Prasath R., Vikram K.K., Vishnu G. Performance optimization of voice-assisted file management systems. International *Journal of Engineering and Computer Science*, 2024, vol. 13, no. 7, pp. 26250–26256. https://doi.org/10.18535/ijecs/v13i07.4854
4. Cho K., Bahn H. A lightweight file system design for unikernel. *Applied Sciences*, 2024, vol. 14, no. 8, pp. 3342. https://doi.org/10.3390/app14083342
5. Gui J., Wang Y., Shuai W. Improving reading performance by file prefetching mechanism in distributed cache systems. *Concurrency and Computation: Practice and Experience*, 2024, vol. 36, no. 22, pp. e8215 https://doi.org/10.1002/cpe.8215
6. Yuliana M., Hidayah N., Sudarsono A. Implementation of Web-Based file sharing security system. *MOTIVECTION: Journal of Mechanical, Electrical and Industrial Engineering*, 2024, vol. 6, no. 1, pp. 41–52. https://doi.org/10.46574/motivection.v6i1.314

7. Man T., Osipov V.Yu., Zhukova N., Subbotin A., Ignatov D. Neural networks for intelligent multilevel control of artificial and natural objects based on data fusion: A survey // Information Fusion. 2024. V. 110. P. 102427. https://doi.org/10.1016/j.inffus.2024.102427
8. Man T., Vodyaho A., Zhukova N., Subbotin A., Shichkina Y. Urban intelligent assistant on the example of the escalator passenger safety management at the subway stations // Scientific Reports. 2023. V. 13. N 1. P. 15914. https://doi.org/10.1038/s41598-023-42535-x
9. Osipov V., Zhukova N., Subbotin A., Glebovskiy P., Evnevich E. Intelligent escalator passenger safety management // Scientific Reports. 2022. V. 12. N 1. P. 5506. https://doi.org/10.1038/s41598-022-09498-x
10. Cowan D.D., Stepien T.M., Ierusalimschy R., Lucena C.J.P. Application integration: Constructing composite applications from interactive components // Software: Practice and Experience. 1993. V. 23. N 3. P. 255–275. https://doi.org/10.1002/spe.4380230304
11. Sudharsan S., Sakthi Anand A., Shanmugaraj K., Palani Samy K.C. Deep learning-based intelligent video surveillance system for real-time motion detection // International Scientific Journal of Engineering and Management. 2024. V. 3. N 4. P. 1–10. https://doi.org/10.55041/ISJEM01492
12. Li J., Zheng Z., Li Y., Ma R., Xia S. Multitask deep learning for Edge Intelligence Video Surveillance system // Proc. of the IEEE 18th International Conference on Industrial Informatics (INDIN). 2020. P. 579–584. https://doi.org/10.1109/INDIN45582.2020.9442166
13. Жукова Н.А., Субботин А.Н. Алгоритм динамического распределения обработки изображений в облачных системах интеллектуального видеонаблюдения // Информационно-управляющие системы. 2024. № 6 (133). С. 15–26. https://doi.org/10.31799/1684-8853-2024-6-15-26
14. Bhatia J., Patel T., Trivedi H., Majmudar V. Htv dynamic load balancing algorithm for virtual machine instances in cloud // Proc. of the International Symposium on Cloud and Services Computing. 2012. P. 15–20. https://doi.org/10.1109/ISCOS.2012.25
15. Rahm E., Do H.H. Data cleaning: problems and current approaches // Bulletin of the IEEE Computer Society Technical Committee on Data Engineering. 2000. V. 23. N 4. P. 3–13.
16. Saecker M., Markl V. Big data analytics on modern hardware architectures: a technology survey // Lecture Notes in Business Information Processing. 2013. V. 138, P. 125–149. https://doi.org/10.1007/978-3-642-36318-4_6

7. Man T., Osipov V., Zhukova N., Subbotin A., Ignatov D. Neural networks for intelligent multilevel control of artificial and natural objects based on data fusion: A survey. *Information Fusion*, 2024, vol. 110, pp. 102427. https://doi.org/10.1016/j.inffus.2024.102427
8. Man T., Vodyaho A., Zhukova N., Subbotin A., Shichkina Y. Urban intelligent assistant on the example of the escalator passenger safety management at the subway stations. *Scientific Reports*. 2023. vol. 13, no. 1, pp. 15914. https://doi.org/10.1038/s41598-023-42535-x
9. Osipov V., Zhukova N., Subbotin A., Glebovskiy P., Evnevich E. Intelligent escalator passenger safety management. *Scientific Reports*, 2022, vol. 12, no. 1, pp. 5506. https://doi.org/10.1038/s41598-022-09498-x
10. Cowan D.D., Stepien T.M., Ierusalimschy R., Lucena C.J.P. Application integration: Constructing composite applications from interactive components. *Software: Practice and Experience*, 1993, vol. 23, no. 3, pp. 255–275. https://doi.org/10.1002/spe.4380230304
11. Sudharsan S., Sakthi Anand A., Shanmugaraj K., Palani Samy K.C. Deep learning-based intelligent video surveillance system for real-time motion detection. *International Scientific Journal of Engineering and Management*, 2024, vol. 3, no. 4, pp. 1–10. https://doi.org/10.55041/ISJEM01492
12. Li J., Zheng Z., Li Y., Ma R., Xia S. Multitask deep learning for Edge Intelligence Video Surveillance system. *Proc. of the IEEE 18th International Conference on Industrial Informatics (INDIN)*, 2020, pp. 579–584. https://doi.org/10.1109/INDIN45582.2020.9442166
13. Zhukova N.A., Subbotin A.N. Dynamic distribution algorithm for image processing in cloud-based intelligent video surveillance systems. *Information and Control Systems*, 2024, no. 6 (133). pp. 15–26. (in Russian). https://doi.org/10.31799/1684-8853-2024-6-15-26
14. Bhatia J., Patel T., Trivedi H., Majmudar V. Htv dynamic load balancing algorithm for virtual machine instances in cloud. *Proc. of the International Symposium on Cloud and Services Computing*, 2012, pp. 15–20. https://doi.org/10.1109/ISCOS.2012.25
15. Rahm E., Do H.H. Data cleaning: problems and current approaches. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 2000, vol. 23 no. 4, pp. 3–13.
16. Saecker M., Markl V. Big data analytics on modern hardware architectures: a technology survey. *Lecture Notes in Business Information Processing*, 2013, vol. 138, pp. 125–149. https://doi.org/10.1007/978-3-642-36318-4_6

Авторы

Субботин Алексей Николаевич — аспирант, Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина), Санкт-Петербург, 197022, Российская Федерация; ведущий инженер-программист, ГУП «Петербургский метрополитен», Санкт-Петербург, 190013, Российская Федерация, SC 57226847043, https://orcid.org/0000-0002-4823-6288, alesu1543@gmail.com

Жукова Наталия Александровна — доктор технических наук, профессор, старший научный сотрудник, Санкт-Петербургский федеральный исследовательский центр Российской академии наук (СПб ФИЦ РАН), Санкт-Петербург, 199178, Российская Федерация; SC 56406142300, https://orcid.org/0000-0001-5877-4461, nazhukova@mail.ru

Authors

Alexey N. Subbotin — PhD Student, Saint Petersburg Electrotechnical University "LETI", Saint Petersburg, 197022, Russian Federation; Leading Engineer, Software Developer, State Unitary Enterprise "Petersburg Metropolitan", Saint Petersburg, 190013, Russian Federation, SC 57226847043, https://orcid.org/0000-0002-4823-6288, alesu1543@gmail.com

Nataly A. Zhukova — D.Sc., Professor, Senior Researcher, St. Petersburg Federal Research Center of the Russian Academy of Sciences, Saint Petersburg, 199178, Russian Federation; SC 56406142300, https://orcid.org/0000-0001-5877-4461, nazhukova@mail.ru

Статья поступила в редакцию 06.10.2024
Одобрена после рецензирования 15.01.2025
Принята к печати 25.03.2025

Received 06.10.2024
Approved after reviewing 15.01.2025
Accepted 25.03.2025