МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра информатики и программирования

ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ КОНСОЛЬНОГО POSIX-ПРИЛОЖЕНИЯ ДЛЯ УПРАВЛЕНИЯ ЖУРНАЛАМИ ИЗМЕНЕНИЙ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 441 группы
направления 02.03.03 Математическое обеспечение и администрирование информационных систем
факультета компьютерных наук и информационных технологий
Шарова Кирилла Владимировича

Научный руководитель:
доцент _____ Кудрина Е. В.
подпись, дата

Зав. кафедрой:
к. ф.-м. н., доцент _____ Огнева М. В.
подпись, дата

Саратов 2025

ВВЕДЕНИЕ

Актуальность темы. Современная разработка программного обеспечения не обходится без использования систем контроля версий, средств автоматизации и непрерывной интеграции. В условиях стремительного роста числа программных проектов и команд, работающих в распределённой среде, особую значимость приобретает прозрачность процессов разработки и сопровождения программных продуктов. Одним из важнейших элементов такой прозрачности является журнал изменений (англ. changelog) — структурированный список модификаций, улучшений и исправлений, вносимых в проект с течением времени[1].

Процесс ведения changelog-файлов кажется тривиальным, однако на практике он сопряжён с рядом проблем: отсутствие единых стандартов, несогласованность форматов между проектами, ручной характер редактирования и высокая вероятность ошибок.

Для решения некоторых из этим проблем были предложены спецификации, которые предлагают единый формат ведения журналов изменений, основанный на понятной структуре и общепринятых терминах. Однако, несмотря на популярность этой инициативы, решения для работы с такими журналами либо недостаточно гибкие, либо плохо интегрируются в Unix-среду.

Цель бакалаврской работы – разработка программного инструмента, реализующего работу с одной из спецификаций журналов с интерфейсом командной строки POSIX.

Поставленная цель определила **следующие задачи** :

1. Изучить принципы и концепции Unix проектирования и разработки программного обеспечения.
2. Ознакомиться с популярными подходами к ведению журналов изменения программного обеспечения.
3. Проанализировать существующие инструменты для ведения changelog-файлов, включая ручные и автоматизированные решения, и

выявить их ограничения.

4. Определить требования к приложению для ведения журналов изменений, включая функциональные возможности и выбор формата журнала.
5. Спроектировать и реализовать архитектуру консольного POSIX-приложения в соответствии с парадигмой разработки Unix.
6. Продемонстрировать работу разработанного инструмента.

Методологические основы разработки консольных POSIX-приложений представлены в работах Эрика Стивена Реймонда, Ричарда Мэттью Столлмана.

Теоретическая и/или практическая значимость бакалаврской работы. В рамках данной бакалаврской работы было разработано консольное POSIX-приложение, охватывающее полный цикл работы с журналами изменений в соответствии со спецификацией Keep a Changelog (с англ. «Веди журнал изменений»). Приложение реализует набор команд и опций для управляемого и гибкого процесса обновления журнала. Инструмент может быть интегрирован в процессы разработки и автоматизации, повышая прозрачность и управляемость версионной истории проекта.

Структура и объём работы. Бакалаврская работа состоит из введения, 2 разделов, заключения, списка использованных источников и 12 приложений. Общий объем работы – 106 страниц, из них 58 страниц – основное содержание, включая 11 рисунков и 5 таблиц, список использованных источников информации – 36 наименований.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

Первый раздел «Теоретическая часть» посвящен анализу концептуальных и технологических основ, лежащих в основе проектирования консольного POSIX-приложения для управления журналами изменений.

На начальном этапе рассматривается философия Unix[2] как методологическая основа разрабатываемого инструмента. Описаны ключевые

принципы этой парадигмы: модульность, ясность, композиция, разделение механизма и политики, простота, бережность, представление знаний через структуры данных, наименьшая неожиданность, молчание, восстановление после сбоев, экономия времени разработчика, автоматизация, отложенная оптимизация, разнообразие решений и расширяемость[3]. Отдельно подчёркнута идея «делать одну вещь, но делать её хорошо», которая определяет подход к созданию специализированных, минималистичных CLI-инструментов, взаимодействующих друг с другом через текстовые потоки.

Подробно описаны стандарты интерфейсов командной строки в контексте POSIX[4]: различие между короткими (-x) и длинными (--example) опциями, их историческое происхождение и эргономика, правила передачи аргументов, обработка ошибок, соответствие соглашениям по выводу справки и версии. Уделено внимание обработке аргументов с помощью стандартных и расширенных библиотек.

Далее проанализированы современные подходы к ведению журналов изменений. Раскрывается эволюция от внутренних технических журналов к внешним пользовательским. Рассматриваются существующие практики: GNU-стиль[5], ориентированный на технические детали; спецификация семантического версионирования (semver)[6], обеспечивающая предсказуемое обозначение версий и строгую классификацию изменений; а также спецификация Keep a Changelog[7], предлагающая структурированный, человекочитаемый формат ведения изменений, пригодный как для документации, так и для автоматической обработки в проектах свободного программного обеспечения. Приведены основные категории изменений (Added, Changed, Fixed и др.), структура документа, понятие раздела Unreleased, ссылки на сравнение версий.

В завершение раздела приведён обзор существующих CLI-инструментов для управления журналами изменений: git-changelog, conventional-changelog, semantic-release, auto-changelog, git-chglog, towncrier, release-it,

phly/keep-a-changelog. Дано их сравнение по критериям автоматизации, соответствия POSIX, зависимости от внешней среды, гибкости и расширяемости. Отмечены недостатки текущих решений: либо чрезмерная зависимость от формализованных commit-сообщений и автоматизированной инфраструктуры, либо, напротив, полное отсутствие унификации и POSIX-интерфейса.

Философия Unix, и особенно её принципы модульности, минимализма и явного интерфейса, остаются актуальной и мощной методологической базой при разработке CLI-инструментов.

Формализация журналов изменений (semver и Keep a Changelog) позволяет обеспечить прозрачность, предсказуемость и автоматизируемость версионирования программного обеспечения.

Существующие инструменты либо избыточно автоматизированы, либо не соответствуют POSIX-стандартам, что затрудняет их использование в универсальных Unix-средах.

Требуется CLI-инструмент, соответствующий стандартам POSIX, с гибкостью ручного управления, реализующий одну из спецификаций журналов изменений. Это и стало основанием для разработки рассматриваемого в данной работе приложения changelogctl.

Раздел подводит теоретическую основу под последующую реализацию инструмента, обеспечивая обоснованный выбор как архитектурного подхода, так и формата взаимодействия с пользователем.

Второй раздел «Практическая часть» посвящен реализации программного инструмента changelogctl, представляющего собой консольное POSIX-приложение для ведения журналов изменений, соответствующих спецификации Keep a Changelog. Основное внимание уделено разработке архитектуры, выбору технологий, построению алгоритмов обработки Markdown-документов и реализации интерфейса командной строки, строго соответствующего требованиям POSIX и философии Unix.

На первом этапе была сформулирована и формализована задача, исходя

из требований к универсальности, переносимости и предсказуемости поведения CLI-инструмента. Было определено, что журнал изменений должен вестись в формате Markdown с чётко определённой структурой, а взаимодействие пользователя с программой должно происходить исключительно через командную строку. Приложение должно поддерживать полный жизненный цикл работы с changelog-файлом: от инициализации до фиксации изменений, формирования выпусков, чтения и восстановления предыдущих состояний.

В процессе реализации была разработана модульная архитектура, обеспечивающая разделение ответственности и расширяемость. Была реализована библиотека, включающая модули для работы с конфигурацией, программным представлением объектов changelog-документа, а также компоненты сериализации и десериализации. Особое внимание было уделено синтаксическому анализу Markdown-файлов: студентом предложено и реализовано формальное описание структуры changelog-документа в форме Бэкуса-Наура[8], что позволило создать собственный надёжный алгоритм парсинга. Вся система обработки ошибок была построена вручную, с многоуровневой иерархией типов, что обеспечило высокую степень устойчивости приложения к ошибкам во входных данных и удобство отладки.

Ключевой особенностью реализации является отказ от автоматической генерации changelog-файлов на основе истории коммитов, в пользу явного, управляемого пользователем подхода, предполагающего полуавтоматическое редактирование.

Все компоненты приложения, включая интерфейс командной строки, были реализованы в соответствии с POSIX-стандартом, а также расширением GNU. Была самостоятельно разработана структура команд и опций, обработка аргументов и сообщений об ошибках, взаимодействие с файловой системой и Git-репозиторием. При этом реализованная система поддерживает привычные пользователю соглашения о руководстве пользователя (-h, --help), версии (--version), категории изменений и генерацию ссылок на выпуски.

Для реализации проекта выбран язык программирования Rust, что обеспечило высокую производительность, безопасность и отсутствие зависимости от сторонних интерпретаторов[9]. В работе использованы современные и надёжные библиотеки экосистемы Rust, в том числе для обработки даты, версий, конфигурационных файлов и структуры Markdown-документа с её переводом в синтаксическое дерево. Все элементы программного продукта объединены в единый исполняемый файл, соответствующий требованиям переносимости и удобства использования в Unix-подобных операционных системах.

Таким образом, в результате проведённой работы был разработан и реализован с нуля полнофункциональный программный продукт, охватывающий цикл работы с журналами изменений: от инициализации до оформления версий.

ЗАКЛЮЧЕНИЕ

В ходе выполнения дипломной работы была достигнута поставленная цель — разработано консольное приложение *changelogctl*, предназначенное для управления журналами изменений с использованием интерфейса командной строки, соответствующего стандартам POSIX. Приложение реализует поддержку формата «Keep a Changelog» и обеспечивает предсказуемый и полуавтоматический способ ведения журналов изменений в проектах программного обеспечения.

В рамках исследования были изучены популярные подходы к оформлению журналов изменений, включая как неформальные практики, так и формализованные спецификации. Внимание также было уделено анализу существующих инструментов.

Основываясь на принципах проектирования ПО в духе Unix (в частности, минимализм, композиционность, простота интерфейса и явность поведения), были сформулированы требования к создаваемому инструменту и предложена архитектура, включающая модульную библиотеку и консольное приложение. В качестве языка реализации был выбран Rust, что позволило обеспечить нативность исполнения, высокую надёжность и безопасность кода.

Итоговая реализация приложения *changelogctl* охватывает весь цикл работы с журналом изменений: от инициализации до внесения изменений и оформления выпусков. Результаты демонстрации показали корректность работы приложения, его соответствие заданной спецификации и готовность к применению в сопровождении реальных проектов.

Основные источники информации:

1. Devineni, S. Version Control Systems (VCS) the Pillars of Modern Software Development: Analyzing the Past, Present, and Anticipating Future Trends / Siva Karthik Devineni // International Journal of Science and Research (IJSR). – 2020. – № 9. – С. 1816-1829. – URL:

https://www.researchgate.net/publication/378490782 (дата обращения: 21.10.2024).
2. Dantam, N. Unix Philosophy and the Real World: Control Software for Humanoid Robots / Neil Dantam, Kim Bondergaard, Mattias Johansson, Tobias Furuholm & Lydia Kavraki // Frontiers in Robotics and AI. – 2016. – № 3. – URL: https://www.researchgate.net/publication/297038367 (дата обращения: 19.02.2025).
3. Raymond, E. S. The Art of Unix Programming [Электронный ресурс] / Eric Steven Raymond. – URL: http://www.catb.org/~esr/writings/taoup/html/ (дата обращения: 18.02.2025).
4. The Open Group. The Open Group Base Specifications Issue 7. IEEE Std 1003.1-2017. 2018 Edition [Электронный ресурс]. URL: https://pubs.opengroup.org/onlinepubs/9699919799/ (дата обращения: 30.01.2025).
5. Николаев, А. А. Методология изменения версий программного обеспечения. Семантическое версионирование / А. А. Николаев, И. А. Истомина // Информационные технологии в УИС. – 2022. – № 4. – С. 15-24. – URL: https://www.elibrary.ru/download/elibrary_50136600_65300729.pdf (дата обращения: 11.01.2025).
6. Lacan O. Ведите Changelog: Не позволяйте своим друзьям сливать логи Git в changelog’и [Электронный ресурс]. – URL: https://keepachangelog.com/ru/1.1.0/ (дата обращения: 07.12.2024). – Яз. рус.
7. Stallman R. The GNU coding standards [Электронный ресурс]. – URL: https://www.gnu.org/prep/standards/standards.pdf (дата обращения: 08.12.2024). – Яз. англ.
8. Ваганова, С. Д. Нотация Бэкуса. РБНФ. Язык и метаязык. Формальное описание языков / С. Д. Ваганова, С. С. Титов // Безопасность информационного пространства : дополнение к сборнику научных

трудов XXI Всероссийской научно-практической конференции студентов, аспирантов и молодых ученых, Екатеринбург, 24–25 ноября 2022 года. Том Выпуск 4 (252) д. – Екатеринбург: Уральский государственный университет путей сообщения, 2023. – С. 22-26. – URL: https://elibrary.ru/download/elibrary_54333229_50934731.pdf (дата обращения: 23.03.2025).

9. Zhang, Y. Towards Understanding the Runtime Performance of Rust / Yuchen Zhang, Yunhang Zhang, Georgios Portokalidis, Jun Xu // ASE '22: 37th IEEE/ACM International Conference on Automated Software Engineering. – 2023. – № 140. – С. 1-6. – URL: https://www.researchgate.net/publication/366906441 (дата обращения: 20.01.2025).