АВТОМАТИЗАЦИЯ ПРОЦЕССОВ ОБРАБОТКИ БОЛЬШИХ ДАННЫХ С ИСПОЛЬЗОВАНИЕМ DEVOPS

К.А. Полянцева, *к.т.н., доцент, Московский технический университет связи и информатики, k.a.poliantseva@mtuci.ru*;
К.О. Егорова, *Московский технический университет связи и информатики, karina.ezhem@mail.ru.*

УДК 42+67 (004.9)

Аннотация. В статье проведен комплексный анализ интеграции *DevOps* (*Development and Operations*) и *Big Data*: рассмотрены теоретические основы, сравнительный анализ, мировой опыт, архитектурные решения, практические аспекты внедрения, риски и перспективы (*DataOps – Data Operations*, *MLOps – Machine Learning Operations*, *AIOps – Artificial Intelligence for IT Operations*). Сформулированы выводы и практическая значимость.

Ключевые слова: *Big Data; DevOps; DataOps; MLOps; AIOps; CI/CD;* контейнеризация; оркестрация; *IaC.*

AUTOMATING BIG DATA PROCESSING USING DEVOPS

K.A. Polyantseva, *Candidate of Technical Sciences, Associate Professor, Moscow Technical University of Communications and Informatics;*
K.O. Egorova, *Moscow Technical University of Communications and Informatics.*

Annotation. The article provides a comprehensive analysis of the integration of *DevOps* (*Development and Operations*) and Big Data: theoretical foundations, comparative analysis, world experience, architectural solutions, practical aspects of implementation, risks and prospects (*DataOps – Data Operations, MLOps – Machine Learning Operations, AIOps – Artificial Intelligence for IT Operations*) are considered. Conclusions and practical significance are formulated.

Keywords: *Big Data; DevOps; DataOps; MLOps; AIOps; CI/CD*; containerization; orchestration; *IaC.*

Введение

Современный этап развития информационных технологий характеризуется стремительным ростом объемов данных, которые необходимо обрабатывать, анализировать и использовать для принятия управленческих решений. По оценкам аналитических агентств, ежегодный объем создаваемых и передаваемых данных увеличивается экспоненциально (рис. 1), что требует принципиально новых подходов к организации вычислительных процессов. Традиционные методы разработки и эксплуатации программных систем становятся недостаточно гибкими и не способны обеспечить требуемую скорость и надежность при работе с большими данными (*Big Data*).

Одним из наиболее эффективных направлений цифровой трансформации является методология *DevOps* (от англ. *Development and Operations* – разработка и эксплуатация) [1], ориентированная на интеграцию процессов создания, тестирования и сопровождения программных продуктов. Основная идея *DevOps* заключается в объединении команд разработчиков и специалистов по эксплуатации, внедрении автоматизации и непрерывной интеграции *(CI, Continuous Integration)* и доставки *(CD, Continuous Delivery)* [2] программных решений. Применение данных

принципов в сфере обработки больших данных обеспечивает сокращение времени вывода систем в эксплуатацию, повышение их отказоустойчивости и адаптивности.

Рисунок 1

Интеграция *DevOps* и *Big Data* формирует новый уровень технологического взаимодействия, при котором процессы обработки информации становятся непрерывными, управляемыми и воспроизводимыми. Использование контейнеризации *(Docker, Podman)*, оркестрации *(Kubernetes, OpenShift)* и инфраструктуры как кода *(IaC, Infrastructure as Code)* позволяет создавать гибкие и масштабируемые платформы для анализа больших объемов данных в режиме реального времени.

Целью работы, результаты которой отражены в данной статье, является комплексное исследование возможностей применения *DevOps* для автоматизации процессов обработки больших данных, выявление преимуществ, рисков и перспектив дальнейшего развития этой интеграции.

Теоретические основы *DevOps* и *Big Data*

Концепция *Big Data* в современной науке и практике описывается через так называемую модель «5*V*»: объем *(Volume)*, скорость *(Velocity)*, разнообразие *(Variety)*, достоверность *(Veracity)* и ценность *(Value)*. Эти характеристики определяют специфику работы с большими данными. Объем данных требует масштабируемых систем хранения; скорость поступления информации – высокопроизводительных технологий обработки; разнообразие форматов (структурированные, полуструктурированные и неструктурированные данные) – гибких подходов к интеграции; достоверность – контроля качества данных; ценность – способности выделять полезную информацию для бизнеса и науки.

Среди технологий *Big Data* ключевыми являются [3]:

- *Hadoop* – экосистема для распределенного хранения и параллельной обработки больших объемов данных, рассчитанная на горизонтальное масштабирование кластера.
- *Apache Spark* – вычислительный движок с упором на обработку в оперативной памяти, позволяющий выполнять пакетную и потоковую аналитику с низкой задержкой, вплоть до *near real-time*.
- *Apache Kafka* – распределенная платформенная шина для потоковой передачи сообщений, широко применяемая в событийно-ориентированных (*event-driven*) архитектурах.

- *NoSQL* СУБД *(MongoDB, Cassandra)* [4] – нереляционные хранилища с высокой доступностью и гибкой схемой данных, удобные для работы с полуструктурированными и разнотипными наборами.

DevOps представляет собой методологический подход, объединяющий процессы разработки программного обеспечения *(Dev)* и его эксплуатации *(Ops)*. Главный упор делается на автоматизацию процессов и обеспечение непрерывной интеграции. Основные принципы *DevOps* включают:

- *CI/CD (Continuous Integration/Continuous Delivery*) – использование конвейеров автоматизации (таких как *Jenkins, GitLab CI/CD*) для упрощения процессов сборки, тестирования и выпуска программных продуктов.
- *IaC (Infrastructure as Code)* [5] – управление инфраструктурой через код *(Terraform, Ansible)*, обеспечивающее воспроизводимость окружений и сокращающее долю ручных операций.
- Контейнеризация – технологии *(Docker, Podman)*, которые позволяют упаковывать приложения вместе с зависимостями и запускать их в любом окружении [6];
- Оркестрация контейнеров осуществляется с помощью специализированных инструментов, таких как *Kubernetes* и *OpenShift*. Эти решения позволяют автоматизировать управление сервисами, включая масштабирование, обеспечение отказоустойчивости и восстановление после сбоев.
- Мониторинг и логирование реализуются с использованием таких систем, как *Prometheus*, *Grafana* и *ELK*-стек. Они помогают в реальном времени отслеживать состояние приложений и инфраструктуры, предоставляя ценную информацию для анализа и своевременного устранения проблем.

Анализ современных инструментов выявляет их ключевые преимущества: *Docker* лидирует благодаря развитой экосистеме и интуитивному интерфейсу, в то время как *Podman* фокусируется на безусловном приоритете безопасности. *Kubernetes* утвердился в роли отраслевого стандарта для оркестрации, а платформа *OpenShift* расширяет его корпоративными функциями, среди которых – встроенные механизмы защиты. *Jenkins* сохраняет позиции за счет гибкости и огромной библиотеки плагинов, тогда как *GitLab CI/CD* предлагает бесшовную интеграцию в рамках единой *DevOps*-платформы.

Эффект синергии между *DevOps* и *Big Data* заключается в объединении автоматизации и адаптивности первой с требованием к масштабируемости и скорости второй. Этот симбиоз позволяет создавать высокопроизводительные системы, способные надежно обрабатывать колоссальные массивы данных – от терабайтов до петабайтов. Яркой иллюстрацией служит тандем *Apache Spark* и *Kubernetes*: контейнеризация добавляет гибкости, а оркестрация гарантирует отказоустойчивость и динамическое масштабирование вычислительных кластеров.

Сравнительный анализ традиционного и *DevOps*-подхода

Традиционный подход к разработке программного обеспечения основан на каскадной модели *(Waterfall)*, которая предполагает строгое следование этапам: проектирование, разработка, тестирование, развертывание и эксплуатация. Каждый этап завершается перед переходом к следующему [7]. Такой подход обеспечивает предсказуемость, но делает процесс жестким и малоподвижным. При изменении требований приходится вносить корректировки уже после завершения отдельных этапов, что приводит к задержкам и росту затрат.

DevOps-подход принципиально отличается. Его основой является интеграция процессов разработки и эксплуатации, ориентация на непрерывные изменения и автоматизацию. Вместо крупных релизов несколько раз в год команды выпускают обновления каждую неделю или даже ежедневно. Ключевыми элементами являются *CI/CD*, тестирование на всех стадиях, контейнеризация и мониторинг. Сравнение ключевых характеристик традиционного и *DevOps*-подходов представлены в табл. 1.

Таблица 1.

Критерий	**Традиционный подход**	***DevOps*-подход**
Модель разработки	Каскадная *(Waterfall)*	Итеративная, *Agile* + автоматизация
Время релиза	Месяцы	Дни или недели
Выявление ошибок	На этапе тестирования	На всех стадиях (автотесты, *CI/CD*)
Масштабируемость	Ограниченная	Гибкая, автоматическая
Затраты	Растут при изменении требований	Снижаются за счет автоматизации
Роль эксплуатации	Подключается после завершения разработки	Интегрирована в процесс с самого начала
Инструменты	Ручное тестирование, серверные развертывания	*CI/CD (Jenkins, GitLab), Docker, Kubernetes, IaC*

Как видно из табл. 1, *DevOps* обеспечивает значительное ускорение релизов и более высокое качество продукта. Благодаря автоматизации тестирования и развертывания, вероятность ошибок снижается.

С точки зрения управления проектами, традиционный подход опирается на иерархическую структуру, тогда как *DevOps* строится на принципах коллаборации и ответственности команд. Это способствует формированию новой организационной культуры, где разработчики и специалисты по эксплуатации работают совместно.

Однако *DevOps* также имеет свои ограничения. Для его внедрения необходимы инвестиции в обучение персонала и автоматизацию процессов. Кроме того, не все компании могут отказаться от элементов традиционного подхода, особенно в проектах с жесткими регуляторными требованиями (например, в авиации или медицине).

Сравнительный анализ подтверждает, что *DevOps* представляет собой более современный и гибкий подход, лучше адаптированный к быстро меняющимся условиям цифровой экономики. Хотя традиционные методологии остаются актуальными для проектов с четко заданными требованиями, большинство компаний отдают предпочтение *DevOps*, стремясь повысить эффективность процессов и сократить сроки вывода продуктов на рынок.

Мировой опыт применения *DevOps* в *Big Data*

Интеграция *DevOps* и *Big Data* активно развивается в мировой практике. Наиболее показательные примеры демонстрируют компании *Netflix, Amazon, Google* и *Microsoft*, а также отрасли финансов, медицины и государственного управления [8-10].

Netflix является одним из лидеров в применении *DevOps*. Компания строит свою инфраструктуру на облачных решениях *AWS*, активно использует микросервисную архитектуру, *CI/CD* и контейнеризацию. Характерной особенностью *Netflix* стало применение хаос-инжиниринга: инструмент *Chaos Monkey* намеренно выводит из строя сервисы, позволяя проверить устойчивость системы. Для обработки потоковых данных *Netflix* применяет *Apache Kafka*, *Spark* и *Druid*, хранение обеспечивается через *Amazon S3*. Дополнительно компания внедрила концепцию *data mesh*, делегируя ответственность за доменные данные отдельным командам, что ускоряет внедрение инноваций.

Amazon не только внедряет *DevOps*, но и предлагает решения через *AWS*. Внутри компании действует принцип *API-first*: все сервисы разрабатываются с обязательным предоставлением интерфейсов. *DevOps*-практики обеспечивают автоматизацию развертывания, масштабирование и мониторинг. Сервисы *AWS (Kinesis, EMR, Redshift, Glue)* интегрируются в *Big Data*-проекты, обеспечивая гибкость и масштабируемость. Это позволяет *Amazon* ускорять выпуск новых продуктов и поддерживать устойчивость глобальной инфраструктуры.

Google сыграла ключевую роль в развитии *DevOps*-инструментов, создав *Kubernetes*, который стал стандартом для оркестрации контейнеров. Компания внедряет *DevOps*-практики для управления глобальными сервисами, такими как *Google Search* и *YouTube*. Использование распределенных систем обработки данных *(MapReduce, BigQuery)* [11] в сочетании с *DevOps* позволяет *Google* обеспечивать высокую производительность и надежность.

Microsoft активно развивает экосистему *Azure DevOps*. Инструменты *CI/CD*, интегрированные с облачными сервисами *Azure*, поддерживают проекты по обработке данных, обеспечивая автоматизированные пайплайны и управление инфраструктурой. *Microsoft* акцентирует внимание на гибридных архитектурах, объединяющих локальные и облачные ресурсы.

DevOps также активно применяется в разных отраслях. В финансовом секторе *DevOps* и *Big Data* используются для мониторинга транзакций и борьбы с мошенничеством. Банки применяют автоматизированные пайплайны для интеграции моделей машинного обучения. В медицине *DevOps* обеспечивает обработку геномных данных и медицинских изображений, поддерживает пайплайны машинного обучения для диагностики. Государственные системы применяют *DevOps* в проектах электронного правительства, обработке больших массивов данных о населении и прогнозировании социальных процессов.

Таким образом, мировой опыт подтверждает, что *DevOps* и *Big Data* становятся стандартом в цифровой экономике. Их совместное применение обеспечивает высокую скорость внедрения инноваций и устойчивость информационных систем.

Архитектурные решения *DevOps* и *Big Data*

Архитектурные решения, применяемые для интеграции *DevOps* и *Big Data* (рис. 2), можно разделить на три основных типа: *on-premise*, облачные и гибридные. Каждый из этих подходов имеет свои преимущества и ограничения, которые определяют выбор архитектуры в зависимости от целей и возможностей организации.

Рисунок 2

On-premise-архитектуры предполагают использование локальной инфраструктуры компании [12]. Такой подход обеспечивает полный контроль над данными и высокий уровень безопасности, что особенно важно для банковской сферы, государственных учреждений и предприятий, работающих с чувствительной информацией. Однако основным недостатком являются значительные затраты на оборудование и поддержание инфраструктуры, а также сложность масштабирования.

Облачные архитектуры *(Amazon Web Services, Microsoft Azure, Google Cloud)* обеспечивают гибкость, масштабируемость и возможность быстрого развертывания сервисов. В этих случаях *DevOps*-инструменты тесно интегрируются с облачными сервисами: *Kubernetes* управляет контейнерами, *Terraform* и *Ansible* обеспечивают инфраструктуру как код, *Jenkins* и *GitLab CI/CD* реализуют автоматизированные пайплайны. Облачные решения позволяют быстро адаптироваться к изменению нагрузки, но зависят от надежности и безопасности провайдера.

Гибридные архитектуры сочетают преимущества локальных и облачных решений. Они позволяют хранить критически важные данные на локальных серверах, а менее чувствительные задачи переносить в облако. Такой подход широко применяется в медицинских и финансовых проектах, где необходим баланс между безопасностью и масштабируемостью. Гибридные архитектуры становятся особенно популярными благодаря развитию технологий контейнеризации и оркестрации, которые позволяют унифицировать процессы развертывания и эксплуатации.

Таким образом, архитектурные решения *DevOps* и *Big Data* обеспечивают универсальность и гибкость в управлении информационными потоками. Выбор конкретной архитектуры зависит от требований к безопасности, бюджета компании и уровня зрелости *IT*-инфраструктуры.

Практические аспекты внедрения *DevOps* в *Big Data*

Внедрение *DevOps* в проекты, связанные с обработкой больших данных, представляет собой многоэтапный процесс, который требует как технологических, так и организационных изменений. Успешное применение возможно только при комплексном подходе, включающем выбор инструментов, адаптацию бизнес-процессов и обучение персонала.

Этапы внедрения *DevOps* в *Big Data* можно разделить на несколько ключевых шагов:

1. Анализ текущего состояния инфраструктуры. На этом этапе проводится аудит существующих систем обработки данных, оценивается готовность к автоматизации и выявляются узкие места.
2. Построение *CI/CD* пайплайнов (рис. 3). Используются инструменты *Jenkins, GitLab CI/CD* или *Azure DevOps* для автоматизации сборки, тестирования и развертывания сервисов обработки данных.
3. Контейнеризация и оркестрация. *Docker* и *Podman* применяются для упаковки приложений, *Kubernetes* и *OpenShift* – для их масштабируемого и отказоустойчивого управления.
4. Автоматизация инфраструктуры. *Terraform, Ansible* и *Chef* позволяют описывать инфраструктуру как код, создавая повторяемые окружения для разработки и эксплуатации.
5. Мониторинг и логирование. Инструменты *Prometheus, Grafana и ELK*-стек обеспечивают наблюдаемость за системами и своевременное реагирование на сбои.
6. Организационные изменения. Для эффективного внедрения *DevOps* требуется трансформация культуры компании: формирование кросс-функциональных команд, пересмотр ответственности и внедрение практики *«you build it – you run it»*.

Схема *CI/CD* пайплайна представлена на рис. 3.

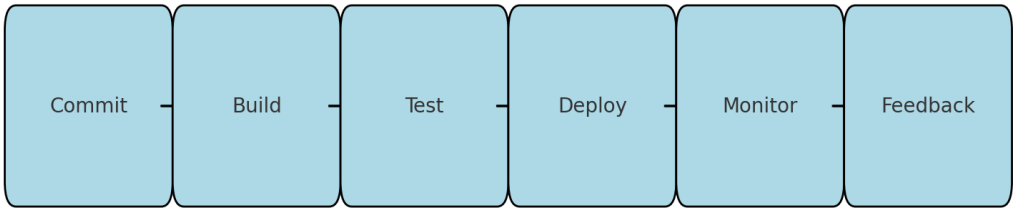

Рисунок 3

Практические кейсы показывают, что внедрение *DevOps* в *Big Data* может занимать от нескольких месяцев до двух лет в зависимости от масштаба организации и зрелости ее процессов. Например, крупные банки используют *DevOps* для обеспечения непрерывного анализа транзакций в реальном времени и выявления мошеннических операций. В медицинских центрах *DevOps* помогает поддерживать пайплайны анализа геномных данных, где критична скорость и качество обработки. В государственных проектах *DevOps* обеспечивает надежность цифровых платформ, обрабатывающих миллионы запросов ежедневно.

Одним из ключевых барьеров является недостаток квалифицированных специалистов. *DevOps*-инженеры должны обладать компетенциями в области администрирования, программирования, автоматизации и анализа данных. Для снижения рисков многие организации начинают внедрение с пилотных проектов, на которых отрабатываются процессы и формируются внутренние стандарты.

Таким образом, практические аспекты внедрения *DevOps* в *Big Data* включают сочетание технологических решений и организационных преобразований. Успех зависит от комплексного подхода, поддержки руководства и готовности персонала к изменениям.

Проблемы и риски внедрения *DevOps* в *Big Data*

Несмотря на значительные преимущества интеграции *DevOps* и *Big Data*, внедрение данных подходов сопряжено с рядом проблем и рисков, которые необходимо учитывать при проектировании и эксплуатации систем:

1. Технические риски. Основная сложность связана с интеграцией *DevOps*-инструментов в существующую инфраструктуру. Наличие устаревших *(legacy)* систем и несовместимость форматов данных создают барьеры для автоматизации. Кроме того, масштабируемость систем *Big Data* может быть ограничена аппаратными и сетевыми ресурсами.

2. Организационные проблемы. *DevOps* предполагает трансформацию культуры компании: отказ от иерархических моделей управления и переход к кросс-функциональным командам. Однако сопротивление изменениям со стороны персонала является серьезным препятствием. Отсутствие согласованности между подразделениями разработки и эксплуатации приводит к конфликтам и снижению эффективности.

3. Финансовые риски. Внедрение *DevOps* требует инвестиций в инструменты автоматизации, облачные сервисы и обучение сотрудников. Начальные затраты могут быть значительными, что создает барьер для компаний среднего и малого бизнеса. Кроме того, ошибки на этапе внедрения способны привести к дополнительным расходам.

4. Кадровые риски. На рынке труда наблюдается дефицит специалистов с компетенциями в области *DevOps* и *Big Data*. Высокая конкуренция за квалифицированных инженеров повышает стоимость найма и снижает скорость формирования команд. Для решения проблемы компании вынуждены инвестировать в обучение собственных сотрудников.

5. Риски информационной безопасности. Автоматизация и использование облачных решений увеличивают поверхность атаки. Уязвимости в *CI/CD* пайплайнах или контейнерах могут быть использованы злоумышленниками. Также существует риск утечки данных при хранении и обработке информации в облачных сервисах.

6. Юридические и регуляторные риски. В отдельных отраслях (например, медицина, финансы, государственные услуги) существуют строгие требования к хранению и обработке данных. Несоблюдение стандартов (*GDPR, HIPAA* и др.) может привести к штрафам и потере доверия пользователей.

Таким образом, внедрение *DevOps* в *Big Data* является сложным процессом, требующим комплексного управления рисками. Успешное применение возможно только при сочетании технологических мер, организационной поддержки и внимания к вопросам безопасности.

Перспективы развития *DevOps* в *Big Data*

Современные тенденции показывают, что классический *DevOps* эволюционирует в новые направления (рис. 4), ориентированные на работу с данными и интеллектуальными системами. К числу наиболее значимых относятся *DataOps, MLOps* и *AIOps* [13]. Эволюция направлений *DevOps* отражена на рис. 4.

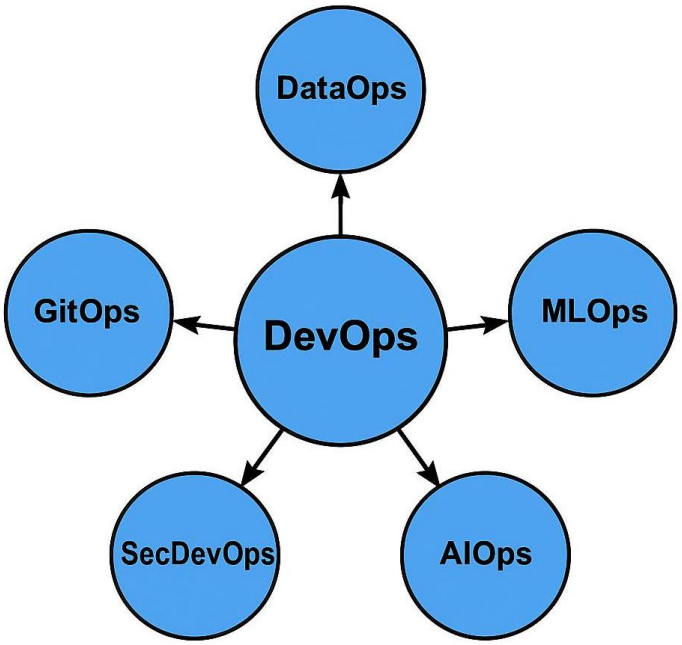

Рисунок 4

DataOps. Данный подход фокусируется на управлении данными и их качестве. Основная цель *DataOps* – автоматизация процессов интеграции, подготовки и доставки данных для аналитики и бизнес-приложений. По аналогии с *DevOps, DataOps* объединяет специалистов по данным, инженеров и аналитиков в единые команды. Это позволяет сократить время от поступления данных до получения бизнес-результатов. Ключевые инструменты: *Apache NiFi, Talend, dbt (data build tool)*. В будущем *DataOps* станет стандартом для компаний, работающих с большими объемами информации.

MLOps. Это направление направлено на автоматизацию жизненного цикла моделей машинного обучения. Оно включает процессы подготовки данных, обучения моделей, их тестирования, развертывания и мониторинга. *MLOps* позволяет значительно ускорить внедрение интеллектуальных сервисов и повысить их качество. Ключевые инструменты: *MLflow, Kubeflow, TensorFlow Extended*. Применение *MLOps* уже активно в таких областях, как финансы (скоринг клиентов), медицина (диагностика по изображениям), маркетинг (персонализация предложений) [14].

AIOps. *Artificial Intelligence for IT Operations* – это использование искусственного интеллекта для автоматизации управления *IT*-инфраструктурой. *AIOps* объединяет анализ больших данных, машинное обучение и *DevOps*-практики для предиктивной аналитики и самовосстановления систем. Основные сценарии применения: выявление аномалий, прогнозирование отказов, автоматическое реагирование на инциденты. Инструменты: *Splunk, Moogsoft, IBM Watson AIOps*. В перспективе *AIOps* станет основой для построения полностью автономных *IT*-систем.

Таким образом, перспективы развития *DevOps* в контексте *Big Data* связаны с дальнейшей автоматизацией и интеллектуализацией процессов. *DataOps* обеспечивает управление данными, *MLOps* – эффективную работу с моделями машинного обучения, а *AIOps* – интеллектуальное управление инфраструктурой [15].

Заключение

В ходе проведенного исследования были рассмотрены теоретические и практические аспекты интеграции *DevOps* и *Big Data*. Анализ литературы показал, что в последние годы данные подходы активно развиваются как в зарубежной, так и в отечественной научной и практической среде. *DevOps* зарекомендовал себя как методология, обеспечивающая ускорение процессов разработки и эксплуатации, а *Big Data* – как стратегический ресурс цифровой экономики.

В теоретической части были раскрыты основные принципы *DevOps* (*CI/CD, IaC*, контейнеризация, мониторинг) и ключевые технологии *Big Data (Hadoop, Spark, Kafka, NoSQL)*. Сравнительный анализ подтвердил преимущества *DevOps* по сравнению с традиционными подходами: сокращение времени релиза, повышение качества программного обеспечения, масштабируемость и снижение затрат.

Мировой опыт применения *DevOps* в *Big Data* на примере компаний *Netflix, Amazon, Google* и *Microsoft* продемонстрировал, что сочетание данных технологий обеспечивает устойчивость, гибкость и высокую скорость внедрения инноваций. Внедрение *DevOps* активно развивается также в банковском секторе, здравоохранении и государственных системах.

Рассмотренные архитектурные решения (*on-premise*, облачные и гибридные) позволяют адаптировать *DevOps* и *Big Data* под различные отраслевые и организационные условия. Практические аспекты внедрения показывают, что успех возможен при комплексном подходе, включающем технологические новации и организационные изменения.

В то же время были выявлены проблемы и риски, связанные с внедрением: технические ограничения, организационные барьеры, финансовые и кадровые риски, а также вопросы информационной безопасности. Эти аспекты требуют внимания и системного управления.

Перспективы развития *DevOps* в контексте *Big Data* связаны с эволюцией в направления *DataOps*, *MLOps* и *AIOps*, которые обеспечивают более высокий уровень автоматизации, управления данными и интеллектуализации инфраструктуры. Эти подходы будут определять дальнейшее развитие цифровой трансформации бизнеса и науки в ближайшие годы.

Таким образом, интеграция *DevOps* и *Big Data* является ключевым фактором успеха в условиях цифровой экономики. Она позволяет организациям адаптироваться к изменениям, повышать эффективность процессов и создавать новые конкурентные преимущества. Практическая значимость заключается в возможности применения разработанных подходов для оптимизации работы предприятий, повышения устойчивости информационных систем и сокращения времени вывода новых продуктов и услуг на рынок.

Будущие исследования могут быть направлены на разработку методик оценки эффективности внедрения *DevOps* в *Big Data*, а также на изучение вопросов обеспечения безопасности и этики при использовании интеллектуальных технологий.

Литература

1. Vangala V. Implementing Lean Principles in DevOps for Efficiency and Cost Savings // International Journal of Scientific Research in Computer Science, Engineering and Information Technology, 2025. – Vol. 11. – No. 2. – P. 3064-3074. DOI: 10.32628/CSEIT25112787.
2. Joshi S. A Review of Generative AI and DevOps Pipelines: CI/CD, Agentic Automation, MLOps Integration, and LLMs // International Journal of Innovative

Research in Computer Science and Technology (IJIRCST), 2025. – Vol. 13. – No. 4. – P. 1-14. DOI: 10.55524/ijircst.2025.13.4.1.
3. Muller J., Fischer L. Scalable Data Architectures for Real-Time Big Data Analytics: A Comparative Study of Hadoop, Spark, and Kafka // International Journal of Artificial Intelligence, Big Data, Cloud and Mobile Systems (IJAIBDCMS). 2020. Vol. 1. No. 4. P. 8-18. URL: https://ijaibdcms.org/index.php/ijaibdcms/article/view/24 (дата обращения: 07.10.2025).
4. Lu J. et al. Bridging the gap: Enabling natural language queries for nosql databases through text-to-nosql translation // arXiv preprint arXiv: 2502.11201. 2025. URL: https://arxiv.org/abs/2502.11201 (дата обращения: 01.09.2025).
5. Thota R.C. Enhancing Infrastructure as Code (IaC) with Automated Validation for Reliable and Error-Free Deployments // Journal of Artificial Intelligence Research and Applications, 2024. – Vol. 4. – No. 1. – P. 827-847. DOI: 10.5281/zenodo.15078796.
6. Полянцева К.А. Высоконагруженная платформа для агрегации и анализа неструктурированных данных о состоянии дорожного полотна // Автоматизация в промышленности, 2022. – № 5. – С. 32-37. DOI: 10.25728/avtprom.2022.05.09.
7. Polyantseva K., Gorodnichev M., Moseva M. Ensuring the Reliability of a Highly Loaded Vehicle Monitoring and Traffic Control Platform // Systems of Signals Generating and Processing in the Field of on Board Communications, 2023. – Vol. 6. – No. 1. – P. 365-372. DOI: 10.1109/IEEECONF56737.2023.10092031.
8. Cockcroft A. Netflix, Chaos Monkey Released Into The Wild // Netflix Tech Blog. 2011. URL: https://netflixtechblog.com/netflix-chaos-monkey-released-into-the-wild-6e8a5a6c7d5c (дата обращения: 01.09.2025).
9. AWS Big Data Solutions: AWS Whitepapers. 2020. URL: https://docs.aws.amazon.com/whitepapers/latest/big-data-analytics-options/aws-big-data-analytics-options.html (дата обращения: 01.09.2025).
10. Microsoft Azure DevOps. Documentation. Microsoft, 2021. URL: https://learn.microsoft.com/en-us/azure/devops/?view=azure-devops (дата обращения: 01.09.2025).
11. Lee K.-H., Lee Y.-J., Choi H., Chung Y. D., Moon B. Parallel data processing with MapReduce: a survey // ACM SIGMOD Record, 2011. – Vol. 40. – No. 4. – P. 11-20. DOI: 10.1145/2094114.2094118.
12. Thallam N.S. T. The Evolution of Big Data Workflows: From On-Premise Hadoop to Cloud-Based Architectures // International Journal For Multidisciplinary Research, 2020. – Vol. 2. – Is. 5. DOI: 10.36948/ijfmr.2020.v02i05.52070.
13. Kreuzberger D., Kühl N., Hirschl S. Machine Learning Operations (MLOps): Overview, Definition, and Architecture // IEEE Access, 2023. – Vol. 11. – P. 31866-31879. DOI: 10.1109/ACCESS.2023.3262138.
14. Faubel L., Schmid K. MLOps: A Multiple Case Study in Industry 4.0 // 2024 IEEE 29th International Conference on Emerging Technologies and Factory Automation (ETFA). IEEE, 2024. – P. 01-08.
15. Иевлев К.О., Городничев М.Г. Сравнительный анализ систем хранения данных HDFS и Apache Ozone // Computational Nanotechnology, 2025. – Т. 12. – № 1. – С. 26-33. DOI: 10.33693/2313-223X-2025-12-1-26-33.