Министерство науки и высшего образования Российской Федерации

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

Кафедра информационных систем и программной инженерии

**Реферат**

**по дисциплине**

**«Качество программно-информационных систем»**

**Методика DevOps**

**Выполнил**:

ст. гр. ПРИ-120

Д. А. Грачев

**Принял**:

Хорошева Е. Р.

Владимир, 2022

Оглавление

[ВВЕДЕНИЕ 4](#_Toc152846912)

[1. ЧТО ТАКОЕ DEVOPS 5](#_Toc152846913)

[2. ОПРЕДЕЛЕНИЯ И ТЕРМИНЫ 6](#_Toc152846914)

[3. ПРИНЦИП РАБОТЫ DEVOPS 7](#_Toc152846915)

[4. ЖИЗНЕННЫЙ ЦИКЛ DEVOPS 8](#_Toc152846916)

[5. ИНСТРУМЕНТЫ DEVOPS 11](#_Toc152846917)

[5.1. ВЫБОР ИНСТРУМЕНТОВ 11](#_Toc152846918)

[5.2. ИССЛЕДОВАНИЕ 12](#_Toc152846919)

[5.3. ПЛАНИРОВАНИЕ 13](#_Toc152846920)

[5.4. СБОРКА 14](#_Toc152846921)

[5.5. ТЕСТИРОВАНИЕ 16](#_Toc152846922)

[5.6. МОНИТОРИНГ 17](#_Toc152846923)

[5.7. ЭКСПЛУАТАЦИЯ 17](#_Toc152846924)

[5.8. НЕПРЕРЫВНАЯ ОБРАТНАЯ СВЯЗЬ 18](#_Toc152846925)

[6. ПРЕИМУЩЕСТВА DEVOPS 20](#_Toc152846926)

[7. ТРУДНОСТИ ВНЕДРЕНИЯ DEVOPS 22](#_Toc152846927)

[8. МЕТОДИКИ DEVOPS 23](#_Toc152846928)

[9. ПРИМЕР ПРИМЕНЕНИЯ МЕТОДИКИ 25](#_Toc152846929)

[9.1. ТЕКУЩЕЕ СОСТОЯНИЕ 25](#_Toc152846930)

[9.2. ШАГИ ПО ВНЕДРЕНИЮ МЕТОДИКИ DEVOPS 26](#_Toc152846931)

[9.3. ИЗМЕНЕНИЯ ПОСЛЕ ВНЕДРЕНИЯ МЕТОДИКИ 27](#_Toc152846932)

[9.4. ПРЕИМУЩЕСТВА ПЕРЕХОДА 28](#_Toc152846933)

[9.5. НЕДОСТАТКИ ПЕРЕХОДА 29](#_Toc152846934)

[9.6. НОВЫЕ ПРОЦЕССЫ В КОМПАНИИ 30](#_Toc152846935)

[ЗАКЛЮЧЕНИЕ 32](#_Toc152846936)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 33](#_Toc152846937)

# ВВЕДЕНИЕ

В условиях постоянно меняющегося информационного поля и увеличивающихся требований к программному обеспечению, компании сталкиваются с необходимостью сокращения времени выхода новых продуктов на рынок, а также улучшения процессов разработки и эксплуатации. В данном контексте методика DevOps (Development and Operations) выделяется как подход, ориентированный на интеграцию процессов разработки и эксплуатации программного обеспечения.

Выбор методики DevOps для данного исследования обусловлен ее значимостью в современной ИТ-индустрии. DevOps не только предоставляет подход к автоматизации и оптимизации процессов, но и создает культуру совместной работы между разработчиками и операционными специалистами, что существенно влияет на конечное качество программного продукта и ускоряет его внедрение.

Разделение между разработкой и эксплуатацией часто приводит к тому, что циклы разработки сталкиваются с трудностями, а обновления системы достигают конечного пользователя с задержками. Методика DevOps стремится устранить эти барьеры, обеспечивая непрерывную поставку, улучшение коммуникации и снижение времени отклика на изменения.

# ЧТО ТАКОЕ DEVOPS

DevOps — это набор методик, инструментов и философия культуры, которые позволяют автоматизировать и интегрировать между собой процессы команд разработки ПО и ИТ‑команд. Особое внимание в DevOps уделяется расширению возможностей команд, их взаимодействию и сотрудничеству, а также автоматизации технологий.

Движение DevOps возникло примерно в 2007 году, когда сообщества разработчиков программного обеспечения и ИТ-специалистов заговорили о проблемах в традиционной модели разработки программного обеспечения. В соответствии с этой моделью разработчики, писавшие код, работали независимо от инженеров по эксплуатации, которые занимались развертыванием и поддержкой этого кода. Термин DevOps является комбинацией из двух слов — development (разработка) и operations (эксплуатация) — и отражает процесс объединения этих практик в один непрерывный процесс.

# ОПРЕДЕЛЕНИЯ И ТЕРМИНЫ

* DevOps (Development + Operations) – методология, философия и практики, направленные на объединение процессов разработки и операций для улучшения эффективности, скорости и качества поставки программного обеспечения.
* Цикл разработки – полный процесс создания программного продукта от начала до конца, включая этапы планирования, проектирования, разработки, тестирования, развертывания и поддержки.
* Непрерывная поставка – практика, направленная на автоматизацию процесса развертывания и поставки программного обеспечения, обеспечивая готовность к выпуску на каждом этапе разработки.
* Автоматизация – процесс использования инструментов и технологий для выполнения задач и операций без прямого вмешательства человека.
* Интеграция – объединение кода и изменений от разных разработчиков в единое целое, обеспечивая непрерывную работу приложения.
* Сборка – процесс компиляции и создания исполняемого кода из исходных файлов.
* Тестирование – процесс проверки программного обеспечения на соответствие заданным требованиям, выявление ошибок и обеспечение качества продукта.
* Инструменты CI/CD – совокупность программных средств, включая Jenkins, Bamboo и другие, обеспечивающих автоматизацию интеграции и поставки.
* Мониторинг – наблюдение и анализ производительности приложения и инфраструктуры для выявления проблем и оптимизации работы.
* Безопасность – защита программного обеспечения и инфраструктуры от угроз, включая меры по предотвращению, обнаружению и реагированию на атаки.

# ПРИНЦИП РАБОТЫ DEVOPS

В состав команды DevOps входят разработчики и ИТ‑специалисты, которые работают совместно на протяжении всего жизненного цикла продукта, чтобы повысить скорость и качество развертывания ПО. Это новый стиль работы и важное изменение культуры, имеющее значительные последствия для команд и организации в целом.

В модели DevOps команды по разработке и эксплуатации больше не являются разобщенными. Иногда эти две команды объединяются в одну, причем ее участники работают на протяжении всего жизненного цикла приложения — от разработки и тестирования до развертывания и эксплуатации — и обладают многосторонними профессиональными навыками.

Команды DevOps используют инструменты для автоматизации и ускорения процессов, благодаря чему повышается надежность. Пакет инструментов DevOps позволяет командам решить ключевые задачи DevOps: реализовать непрерывную интеграцию, непрерывную поставку, автоматизацию и совместную работу.

Ценности DevOps иногда используются не только в командах разработчиков. Когда подход DevOps применяют команды по обеспечению безопасности, безопасность становится активной и неотъемлемой частью процесса разработки. Такой подход называют DevSecOps.

# ЖИЗНЕННЫЙ ЦИКЛ DEVOPS

Непрерывный характер DevOps приводит к тому, что специалисты используют бесконечный цикл, чтобы показать взаимосвязь фаз жизненного цикла DevOps. Несмотря на внешнюю последовательность цикла, он символизирует необходимость постоянного сотрудничества и итеративного совершенствования на протяжении всего жизненного цикла.

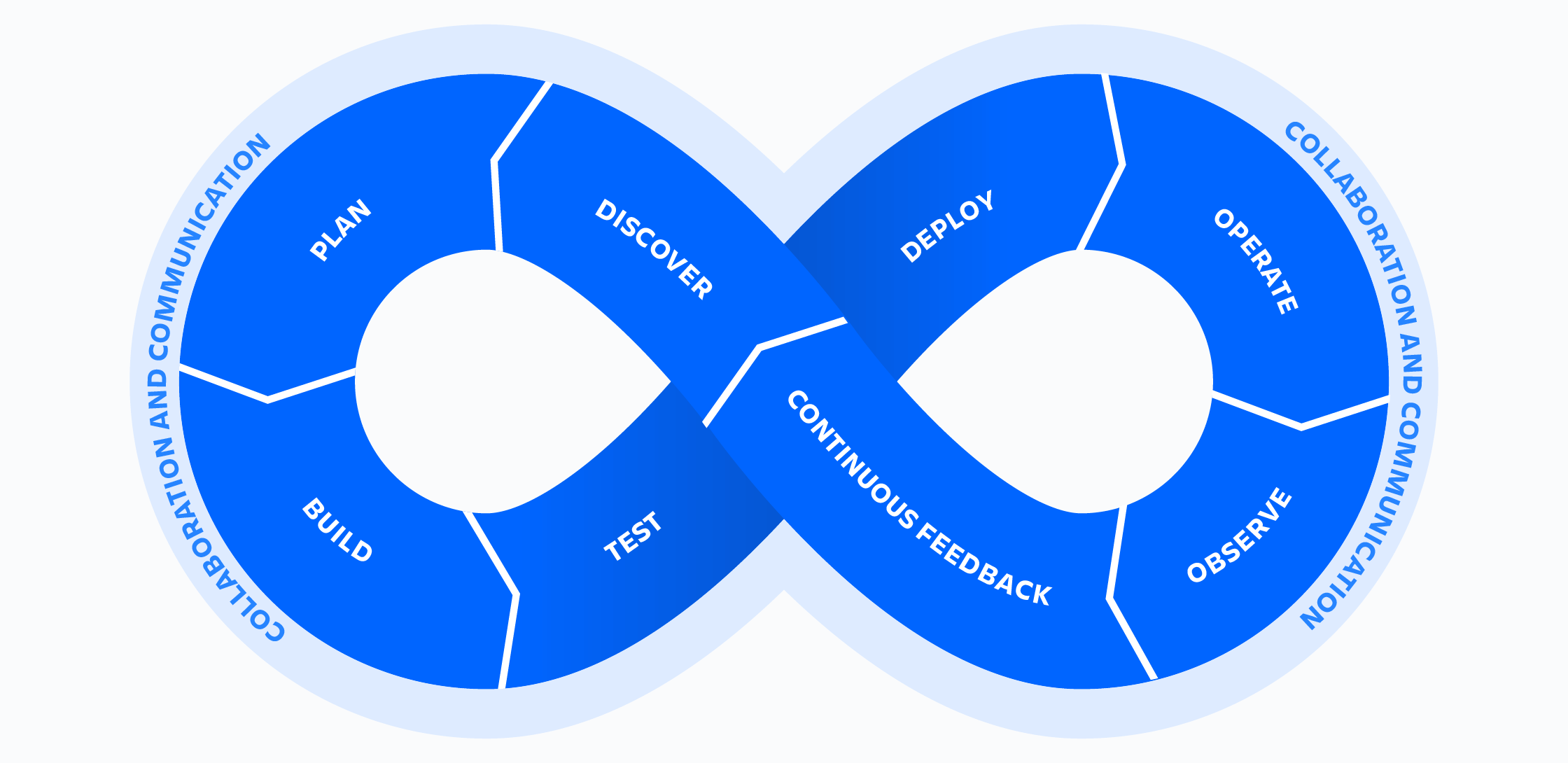


Рисунок 1. Жизненный цикл DevOps

Жизненный цикл DevOps состоит из восьми этапов, причем в левой части бесконечного цикла представлены процессы, возможности и инструменты, необходимые для разработки, а в правой — для эксплуатации. На каждом этапе команды ведут совместную работу и взаимодействуют для поддержания согласованности, скорости и качества.

Ключевые этапы жизненного цикла DevOps:

1. Исследование.

Разработка ПО — это командный спорт. При подготовке к предстоящему спринту команды должны провести семинар, чтобы изучить идеи, систематизировать их и расставить приоритеты. Идеи должны соответствовать стратегическим целям и удовлетворять потребности клиентов. Направлять работу команд DevOps помогает методика agile.

1. Планирование.

Чтобы повысить скорость и качество, команды DevOps должны применять методики Agile. Agile — это итеративный подход к управлению проектами и разработке ПО, который помогает командам разбивать работу на более мелкие части и поставлять результаты отдельными порциями.

1. Сборка.

Git — это бесплатная система управления версиями с открытым исходным кодом. В ней реализована отличная поддержка ветвлений, слияний и перезаписи истории репозитория. Благодаря этому появилось множество инновационных и эффективных рабочих процессов и инструментов, предназначенных для совершенствования процесса разработки.

1. Тест.

Непрерывная интеграция (CI) позволяет нескольким разработчикам помещать код в один общий репозиторий. При слиянии изменений выполняется автоматическое тестирование для проверки правильности кода перед интеграцией. Слияние и тестирование кода часто помогают командам разработчиков убедиться в качестве и предсказуемости развертываемого кода.

1. Развертывание

Непрерывное развертывание (CD) позволяет командам часто и в автоматизированном режиме выпускать в рабочую среду новые возможности. Кроме того, команды могут использовать при развертывании флажки возможностей, чтобы поставлять пользователям новый код постепенно и методично, а не весь сразу. Такой подход увеличивает скорость, производительность и стабильность работы команд разработчиков ПО.

1. Эксплуатация

Управляйте комплексным предоставлением ИТ-услуг клиентам. Это касается методик, связанных с проектированием, внедрением, настройкой, развертыванием и техническим обслуживанием всей ИТ‑инфраструктуры, которая обеспечивает поддержку услуг организации.

1. Наблюдение

Быстро выявляйте и решайте проблемы, которые влияют на время безотказной работы, скорость и функциональные характеристики продукта. Автоматически уведомляйте команду об изменениях, опасных действиях и сбоях, чтобы можно было продолжать предоставление услуг.

1. Непрерывная обратная связь

Команды DevOps должны оценивать каждый релиз и создавать отчеты для улучшения будущих релизов. Имея непрерывную обратную связь, команды могут совершенствовать свои процессы и учитывать отзывы клиентов для повышения качества последующих релизов.

# ИНСТРУМЕНТЫ DEVOPS

Инструменты DevOps применяются на всех основных этапах жизненного цикла DevOps. Они расширяют возможности методик DevOps, способствуя эффективной совместной работе, сокращению количества переключений контекста, внедрению автоматизации и созданию возможностей для наблюдения и мониторинга.

Все пакеты инструментов DevOps делятся на два типа: универсальные и открытые. Универсальный пакет инструментов является комплексным решением; обычно его нельзя интегрировать со сторонними инструментами, тогда как открытый пакет можно настраивать и дополнять различными инструментами. У каждого из этих подходов есть свои плюсы и минусы.

Примером открытого пакета инструментов DevOps является решение Atlassian Open DevOps, которое включает Jira в качестве базового продукта и интегрируется с приложениями ведущих поставщиков и приложениями Marketplace.

## ВЫБОР ИНСТРУМЕНТОВ

DevOps — это следующий этап развития методологии Agile и важное изменение в культуре, при котором команды по разработке и эксплуатации начинают работать более сплоченно. DevOps — это методика, требующая внесения изменений в культуру, внедрения новых принципов управления и передовых технологических инструментов.

Когда дело доходит до выбора пакета инструментов DevOps, организациям необходимо найти такие инструменты, которые способствуют эффективной совместной работе, позволяют уменьшить количество переключений контекста, автоматизировать процессы и использовать возможности мониторинга и наблюдения для ускоренной поставки качественного ПО.

Все пакеты инструментов DevOps делятся на два типа: универсальные и открытые. Универсальный пакет инструментов DevOps — это комплексное решение, которое часто нельзя интегрировать со сторонними инструментами. Открытый пакет можно видоизменять согласно потребностям команды, наполняя необходимыми инструментами. Мы в Atlassian уверены, что наилучшее решение — открытый пакет инструментов, который можно настроить в соответствии с уникальными потребностями организации, добавив в него лучшие в своем классе инструменты. Такой подход способствует эффективному использованию времени и ускорению выхода на рынок.

Какой бы тип пакета ни выбрала организация, процесс DevOps потребует подходящих инструментов на ключевых этапах цикла DevOps.

* Исследование
* Планирование
* Сборка
* Тестирование
* Мониторинг
* Эксплуатация
* Непрерывная обратная связь

Инструменты в составе открытого пакета DevOps предназначены для разных этапов цикла DevOps. В следующих разделах представлены наиболее популярные решения.

## ИССЛЕДОВАНИЕ

На этапе исследования команда DevOps изучает область и определяет объем работ по проекту. Сюда, в частности, входят такие действия, как исследование мнений пользователей, постановка целей и определение критериев успеха.

Команда разработчиков собирает идеи и проводит исследование, используя такие инструменты, как Mural и Miro. С помощью Jira Product Discovery они формируют практически полезные входные данные и определяют приоритеты действий для команд разработчиков. При расстановке приоритетов не забывайте и о бэклоге пользовательских отзывов.



Рисунок 2. Продукты для исследования

Исследование продукта — самое первое действие в процессе разработки продукта. По его результатам в дальнейшем происходит принятие решений. На этапе исследования продукта вы можете собрать всю важную информацию о любых проблемах пользователей и затем предложить их решение.

Мы рекомендуем выбирать инструменты, которые позволяют проводить асинхронные мозговые штурмы. Важно, чтобы каждый мог делиться информацией и обсуждать идеи, стратегии, цели, требования, дорожные карты и документацию.

## ПЛАНИРОВАНИЕ

Для эффективного улучшения производительности и качества внедрения программного обеспечения, команды, осуществляющие практики DevOps, находят выход в применении методологии Agile. Agile – это прогрессивный и итеративный подход к управлению проектами и разработке программного обеспечения, который не только способствует повышению эффективности, но и позволяет командам более эффективно структурировать рабочий процесс.

Суть Agile заключается в том, чтобы разбивать обширные задачи и проекты на более мелкие и управляемые части, что в свою очередь обеспечивает возможность регулярной поставки и оценки результатов в виде небольших инкрементов. Этот принцип позволяет командам не только демонстрировать промежуточные результаты, но и быстрее адаптироваться к изменяющимся требованиям и условиям проекта. Таким образом, интеграция Agile в методологию DevOps становится важным шагом в обеспечении гибкости, эффективности и высокого качества процессов разработки и эксплуатации программного обеспечения.



Рисунок 3. Инструменты для планирования

## СБОРКА



Рисунок 4. Инструменты для формирования среды разработки

Если среди популярных инструментов управления эксплуатацией выделяются решения Puppet и Chef, то для создания отдельных сред разработки используются инструменты с открытым исходным кодом, например Kubernetes и Docker. Программирование в виртуальных одноразовых копиях рабочей среды позволяет выполнять больше работы.

Если все участники команды работают в идентично сконфигурированных средах, проблема неработоспособности ПО на отдельных машинах теряет свою актуальность.



Рисунок 5. Инструменты для инфраструктуры

Разработчики создают модульные приложения, поскольку они более надежны и удобны в обслуживании. Почему бы не распространить этот принцип на ИТ-инфраструктуру? Применить его к системам может быть затруднительно, так как они непрерывно изменяются. Решить проблему можно с помощью кода распределения.

Подход «инфраструктура как код» предполагает, что на повторное распределение потребуется меньше времени, чем на ремонт, и дает более согласованные и воспроизводимые результаты. Этот подход также позволяет без труда создавать варианты среды разработки с конфигурацией, аналогичной конфигурации рабочей среды. Код распределения можно применять для того, чтобы сделать сервер частью известной основы. Его можно хранить в системе контроля версий, тестировать, встраивать в процесс непрерывной интеграции и подвергать оценке коллег.

Когда накопленные знания зафиксированы в коде, исчезает потребность в перечнях процедур и внутренней документации. Процессы становятся воспроизводимыми, а системы — надежными.



Рисунок 6. Инструменты для управления версиями

Иметь контроль над исходным кодом очень важно. Инструменты управления исходным кодом позволяют хранить код в разных цепочках. Так вы сможете видеть все изменения и без труда работать над ними всей командой. Вместо длительных собраний по подтверждению изменений перед развертыванием в рабочей среде проводите проверки в форме оценки коллег с помощью запросов pull. Это повысит скорость работы и качество кода.

Что такое запросы pull? С помощью запроса pull вы сообщаете команде об изменениях, внесенных вами в ветку разработки в репозитории. Прежде чем интегрировать предложенные изменения в главную ветку кода, команда сможет их просмотреть и обсудить. Применение запросов pull повышает качество программного обеспечения, что приводит к уменьшению количества багов и инцидентов. В итоге снижаются расходы на эксплуатацию и ускоряется разработка.

Системы управления версиями исходного кода должны интегрироваться с другими инструментами. Это позволяет объединить различные сегменты разработки и поставки кода и отслеживать поведение кода возможности в рабочей среде. Если произойдет инцидент, код поможет определить его причину.

## ТЕСТИРОВАНИЕ



Рисунок 7. Инструменты для тестирования

Инструменты тестирования охватывают множество потребностей и возможностей, в том числе глубокое тестирование, управление тестированием и оркестрацию. Однако важнейшая функция пакета инструментов DevOps — автоматизация. В долгосрочной перспективе автоматическое тестирование окупается благодаря ускорению циклов разработки и тестирования. В среде DevOps оно важно еще и потому, что повышает осведомленность.

Частое автоматическое тестирование на ранних этапах помогает повысить качество программного обеспечения и снизить риски. Команды разработчиков могут выполнять автоматические тесты регулярно, охватывая различные области: пользовательский интерфейс, безопасность, производительность при разной нагрузке. Определить зоны риска помогут сопутствующие отчеты и графики тенденций.

Риски неизбежны в сфере разработки ПО. Однако их можно свести к минимуму при своевременном выявлении. Особенно рекомендуются инструменты, которые позволяют привлекать специалистов по эксплуатации к блиц- и глубокому тестированию.

## МОНИТОРИНГ



Рисунок 8. Инструменты для мониторинга

Два типа мониторинга требуют автоматизации: мониторинг серверов и мониторинг производительности приложений.

Доработка вручную и тестирование API — отличный вариант для выборочной проверки. Однако для понимания тенденций и общей работоспособности приложений (и сред) необходимо программное обеспечение, которое отслеживает и сохраняет данные круглосуточно и без выходных. Возможность непрерывного наблюдения является ключом к успеху для команд DevOps.

## ЭКСПЛУАТАЦИЯ



Рисунок 9. Инструменты для эксплуатации

Для развития взаимодействия между командами DevOps необходимо, чтобы все сотрудники имели единое и целостное представление о текущей работе. Что происходит при возникновении инцидентов? Выполняется ли их привязка к проблемам программного обеспечения и отслеживаются ли они? Выполняется ли привязка новых изменений к релизам?

Наибольшие помехи взаимодействию между разработчиками и специалистами по эксплуатации возникают, когда отслеживание инцидентов и проекты по разработке ведутся не в единой, а в разных системах. Выбирайте инструменты, которые хранят инциденты, изменения, проблемы и программные проекты на одной платформе, чтобы выявление и устранение проблем было оперативным.

## НЕПРЕРЫВНАЯ ОБРАТНАЯ СВЯЗЬ



Рисунок 10. Инструменты для обратной связи

Если вы делаете правильные шаги, клиенты уже сообщают вам об этом — нужно только прислушаться. Непрерывная обратная связь — это культура и процессы регулярного получения обратной связи, а также инструменты извлечения из нее аналитических данных. Методики непрерывной обратной связи включают сбор и анализ данных NPS, опросов о причинах оттока клиентов, отчетов о багах, заявок в службу поддержки и даже твитов. Согласно принципам культуры DevOps каждый участник команды по продукту должен иметь доступ к комментариям пользователей, поскольку они помогают руководить всем процессом: от планирования релизов до сеансов глубокого тестирования.

Найдите приложения для интеграции чата с вашей любимой платформой для опросов и получайте обратную связь в виде индекса потребительской лояльности (NPS). Для получения обратной связи в режиме реального времени можно выполнить интеграцию чата с Twitter и (или) Facebook. Если вам необходим обширный мониторинг обратной связи из социальных сетей, стоит использовать платформу для управления социальными сетями, которая позволяет формировать статистические отчеты.

Поначалу может показаться, что анализ обратной связи и корректировка на его основе замедляют темпы разработки, однако в долгосрочной перспективе это более эффективно, чем выпуск новых, но никому не нужных возможностей.

# ПРЕИМУЩЕСТВА DEVOPS

По данным опроса Atlassian «Тенденции DevOps» за 2020 год, 99 % респондентов считают, что методика DevOps принесла пользу их организациям. К преимуществам DevOps относятся: более быстрый и простой выпуск релизов, эффективная работа команды, повышение безопасности и качества продуктов и, как следствие, рост удовлетворенности команд и клиентов.

Преимущества методики DevOps:

* Скорость

По данным опроса Atlassian «Тенденции DevOps» за 2020 год, 99 % респондентов считают, что методика DevOps принесла пользу их организациям. К преимуществам DevOps относятся: более быстрый и простой выпуск релизов, эффективная работа команды, повышение безопасности и качества продуктов и, как следствие, рост удовлетворенности команд и клиентов.

* Более эффективная совместная работы

Основой DevOps является культура сотрудничества команд по разработке и эксплуатации, которые несут общую ответственность и объединяют свою работу. При этом повышается эффективность работы команд и сокращаются затраты времени на передачу работы и создание кода, который разрабатывается для среды, в которой выполняется.

* Быстрое развертывание

Благодаря более частому и быстрому выпуску релизов команды DevOps быстро совершенствуют продукты. Быстро выпуская новые возможности и исправляя баги, можно получить конкурентное преимущество.

* Качество и надежность

Непрерывная интеграция и непрерывная поставка делают изменения эффективными и безопасными и повышают качество программного продукта. Мониторинг позволяет командам получать информацию о производительности в режиме реального времени.

* Безопасность

После внедрения безопасности в конвейер непрерывной интеграции, непрерывной поставки и непрерывного развертывания DevSecOps становится активной и неотъемлемой частью процесса разработки. Безопасность встраивается в продукт путем внедрения активных проверок и тестирования безопасности в рабочие процессы agile-разработки и DevOps.

# ТРУДНОСТИ ВНЕДРЕНИЯ DEVOPS

От прежних привычек трудно избавиться. Команды с укоренившимися привычками к изолированной работе могут испытывать трудности и даже противиться перестройке структуры команды для внедрения методик DevOps. Некоторые команды могут ошибочно считать, что для внедрения DevOps достаточно новых инструментов. Однако DevOps — это сочетание людей, инструментов и культуры. Каждый участник команды DevOps должен знать весь поток создания ценности — от идеи до разработки и взаимодействия с конечным пользователем. Для совместной работы на протяжении всего жизненного цикла продукта необходимо преодолеть разобщенность.

Переход от устаревшей инфраструктуры к использованию инфраструктуры как кода (IaC-обработки) и микросервисы позволит ускорить разработку и внедрение инноваций, однако увеличение рабочей нагрузки может оказаться серьезным испытанием. Прочной основой, которая поможет снизить нагрузку, должны стать автоматизация, управление конфигурацией и непрерывная поставка.

Чрезмерная зависимость от инструментов может отвлечь команды от обязательных основ DevOps — команды и организационной структуры. После создания структуры нужно заняться процессами и командой, а уже затем — инструментами.

# МЕТОДИКИ DEVOPS

* Непрерывная интеграция

Непрерывная интеграция — это методика, направленная на автоматизацию интеграции изменений кода в программный проект. Благодаря ей разработчики могут чаще проводить слияние изменений кода в центральный репозиторий, где выполняются сборка и тестирование. Непрерывная интеграция помогает командам DevOps быстрее устранять баги, повышать качество программного обеспечения и сокращать сроки проверки и выпуска новых обновлений ПО.

* Непрерывная поставка

Непрерывная поставка является продолжением непрерывной интеграции и используется для автоматического развертывания изменений кода в среде тестирования и рабочей среде. Создается конвейер непрерывной поставки, в котором автоматизированные процессы сборки, тестирования и развертывания организуются в единый процесс выпуска релизов.

* Ситуационная осведомленность

Очень важно, чтобы каждый участник организации имел доступ к данным, необходимым для максимально быстрого и эффективного выполнения его работы. Участники команды должны получать оповещения о сбоях в конвейере развертывания (включая системные сбои и ошибки тестирования), а также актуальную информацию о работоспособности и производительности приложений, выполняющихся в рабочей среде. Показатели, журналы, трассировки, мониторинг и оповещения — все это важные источники обратной связи, необходимые командам для работы.

* Автоматизация

Автоматизация является одной из важнейших методик DevOps. Она позволяет командам значительно ускорить процесс разработки и развертывания программного обеспечения высокого качества. Благодаря автоматизации простая отправка изменений кода в репозиторий исходного кода может запустить процесс сборки, тестирования и развертывания, что значительно сокращает затраты времени на выполнение этих действий.

* Инфраструктура как код

Независимо от того, имеется ли у организации локальный ЦОД или все работы выполняются в облаке, возможность быстро и бесперебойно предоставлять и настраивать инфраструктуру, а также управлять ею имеет решающее значение для успешного внедрения DevOps. Инфраструктура как код (IaC-обработка) — это не только написание скриптов для различных конфигураций инфраструктуры. С определениями инфраструктуры обращаются как с обычным кодом, то есть используют управление версиями, проверку кода, тестирование и т. д.

* Микросервисы

Микросервисы — это архитектурный прием создания приложения в виде набора небольших служб, которые можно развертывать и эксплуатировать независимо друг от друга. Каждая служба имеет собственные процессы и взаимодействует с другими службами через интерфейс. Такое разделение задач и раздельное независимое функционирование позволяют использовать методики DevOps, например непрерывную поставку и непрерывную интеграцию.

* Мониторинг

Команды DevOps отслеживают весь жизненный цикл разработки, от планирования, разработки и интеграции до тестирования, развертывания и эксплуатации. Поэтому они могут быстро и автоматически реагировать на любое ухудшение качества обслуживания клиентов. Что еще более важно, команды могут «смещаться влево», к более ранним этапам разработки, чтобы минимизировать последствия неудачных изменений в рабочей среде.

# ПРИМЕР ПРИМЕНЕНИЯ МЕТОДИКИ

Рассмотрим интеграцию методики DevOps на примере компании «N», которая занимается продуктовой разработкой инновационных веб-приложений. Далее будет описаны состав команд разработки в компании, шаги по внедрению методики DevOps и выводы по внедрению.

## ТЕКУЩЕЕ СОСТОЯНИЕ

Компания "N" – разработчик программного обеспечения, занимающаяся созданием инновационных продуктов. В настоящее время компания функционирует без использования методики DevOps. Общее количество сотрудников в компании составляет 180 человек.

1. Разработка – отдел, занимающийся разработкой нового функционала и поддержкой старого
   1. Специалисты по фронтенд разработке – 30 человек
   2. Специалисты по бэкенд разработке – 50 человек
2. Тестирование – отдел, занимающийся проверкой функционала, разрабатываемого отделом разработки, на наличие некорректной работы
   1. Специалисты по функциональному тестированию – 20 человек
   2. Специалисты по автоматизированному тестированию – 20 человек
3. Дизайн UI/UX – отдел, занимающийся разработкой дизайна интерфейса и пользовательского опыта
   1. Графические дизайнеры – 15 человек
   2. Специалисты по UX/UI – 15 человек
4. Координаторы – отдел, занимающийся связью с клиентами и координированием работы всех отделов для наилучшего решения клиентских запросов – 30 человек

В данный момент в компании длительные циклы разработки(Среднее время между релизами составляет 3 месяца, что замедляет реакцию на рыночные требования и конкурентные изменения), низкая стабильность развертывания(Высокая частота ошибок после развертывания приводит к нестабильности системы и потере доверия со стороны пользователей, в среднем на один релиз приходится. 15 ошибок) и ограниченная прозрачность(Коммуникация между командами ограничена, что приводит к неэффективной координации и дублированию усилий).

## ШАГИ ПО ВНЕДРЕНИЮ МЕТОДИКИ DEVOPS

Предварительным шагом по внедрению методик DevOps было проведение найма сотрудников на должность «DevOps инженер», которые и будут ответственны за внедрение данной методики, а также будут поддерживать в рабочем состоянии автоматизацию всех процессов.

Первым шагом внедрения DevOps был анализ текущих процессов. Группа DevOps инженеров провела тщательный аудит, выявив узкие места в циклах разработки и проблемы в координации между командами, а так же составила описание всех выявленных процессов и проблем в Confluence.

Следующим шагом было проведено обучение сотрудников. DevOps инженеры должны были организовать воркшопы по основам DevOps, а сотрудники принять участие в обучающих мероприятиях для освоения новых концепций.

Для внедрения автоматизации DevOps инженеры выбрали инструменты для от компании Atlassian. Эти инструменты были интегрированы в процессы, а сотрудники были обучены их использованию.

Использование продуктов Atlassian во внедрении DevOps:

* **Jira:** Внедрение Jira для отслеживания задач, управления проектами и прозрачной отчетности.
* **Confluence:** Создание документации и обмен знаниями между командами.
* **Bitbucket:** Управление исходным кодом с возможностью интеграции с Bamboo для непрерывной интеграции.
* **Bamboo:** Настройка средства для автоматизации сборки и развертывания, обеспечивая непрерывную поставку.
* **Opsgenie:** Внедрение средства для управления инцидентами и быстрого реагирования на проблемы.

Интеграция команд стала ключевым шагом, где DevOps инженеры организовали мероприятия для укрепления командной динамики. Все сотрудники участвовали в тренингах для улучшения коммуникации.

Внедрение непрерывной поставки включало в себя создание конвейера непрерывной интеграции и поставки с использованием инструментов от Atlassian, таких как Bitbucket и Bamboo. Так же были внедрены мониторинг и автоматическое тестирование.

Использование продуктов Atlassian, таких как Jira для отслеживания задач и управления проектами, Confluence для создания документации и обмена знанием, а также Bitbucket и Bamboo для управления исходным кодом и автоматизации, значительно упростило и улучшило процессы.

## ИЗМЕНЕНИЯ ПОСЛЕ ВНЕДРЕНИЯ МЕТОДИКИ

После внедрения методики DevOps в компании произошло большое количество изменений:

1. Снижение времени между релизами

Сроки сократились до 2 недель, что способствует более оперативной поставке нового программного обеспечения.

1. Улучшение качества развертывания

Удалось снизить количество ошибок до 5 на каждый релиз, что привело к повышению стабильности и уверенности пользователей.

1. Повышение уровня коммуникации и прозрачности в процессах

За счет использования новых и единых инструментов по документации проектов и ведения задач повысился уровень коммуникации и прозрачности между сотрудниками. Так, за счет использования Confluence, все разработчики знают, где находится подробная информация о проекте и могут быстро ее найти и изучить. Также сотрудники, за счет использования Jira, могут отслеживать прогресс по задачам.

1. Найм DevOps инженеров

Для успешной реализации методики DevOps в компанию были привлечены 10 DevOps инженеров. Их задачи включают в себя анализ существующих процессов, внедрение инструментов автоматизации, обучение персонала и обеспечение стабильности внедренных решений.

1. Внедрение новых инструментов

Были внедрены продукты Atlassian, такие как Jira для отслеживания задач, Confluence для создания документации, Bitbucket для управления исходным кодом и Bamboo для автоматизации сборки и развертывания. Эти инструменты упростили процессы и обеспечили более эффективное взаимодействие между командами.

## ПРЕИМУЩЕСТВА ПЕРЕХОДА

1. Ускорение цикла разработки

Сокращение времени между релизами позволяет быстрее реагировать на изменения требований рынка и предоставлять новый функционал пользователям.

1. Улучшение качества развертывания

Внедрение практик непрерывной интеграции и поставки способствует уменьшению числа ошибок после развертывания, что повышает стабильность системы.

1. Повышение уровня коммуникации

DevOps способствует сближению команд разработки, тестирования и операций, что улучшает коммуникацию и согласование усилий.

1. Прозрачность в процессах

Внедрение DevOps улучшает прозрачность в процессах, что способствует лучшей координации и пониманию статуса проектов.

1. Непрерывная поставка

Непрерывная поставка позволяет компании быстро реагировать на обратную связь пользователей и оперативно внедрять изменения.

1. Эффективное использование ресурсов

Автоматизация процессов позволяет оптимизировать использование ресурсов и снижает риск человеческих ошибок.

1. Улучшение культуры команды

DevOps способствует созданию единой команды, что создает позитивное рабочее окружение и улучшает мораль сотрудников.

## НЕДОСТАТКИ ПЕРЕХОДА

1. Сложность внедрения

Начальная стадия внедрения может быть сложной и требует времени и ресурсов для обучения персонала и изменения культуры компании.

1. Необходимость изменения процессов

Внедрение DevOps может потребовать значительного изменения существующих процессов, что вызывает сопротивление со стороны сотрудников.

1. Зависимость от инструментов

Отсутствие адекватной интеграции и поддержки инструментов может стать преградой для успешного внедрения DevOps.

1. Безопасность и конфиденциальность данных

Увеличение автоматизации может повысить риски в области безопасности и требовать дополнительных мер по защите данных.

## НОВЫЕ ПРОЦЕССЫ В КОМПАНИИ

После внедрения методики DevOps в компании появились новые процессы, связанные с разработкой и развертыванием кода.

В первую очередь изменился подход к постановке задач: задачи ставятся с использованием системы управления проектами Jira. Каждая задача имеет четко определенные требования и приоритет, а также привязана к конкретному этапу цикла разработки. Вся команда участвует в обсуждении задач, и важные аспекты фиксируются в Confluence, обеспечивая легкость доступа к информации.

Документирование также ведется с использованием Confluence: здесь создаются страницы с описанием функционала, архитектуры, API, схем данных и других важных аспектов продукта. Информация регулярно обновляется и согласовывается с изменениями в коде, обеспечивая актуальность документации.

Инструмент Bamboo используется для настройки и автоматизации процесса сборки и развертывания. Каждое изменение в коде тестируется автоматически, и в случае успешного прохождения тестов, система автоматически развертывает обновление. Отчеты о статусе развертывания и результаты тестов доступны в реальном времени через интерфейс Bamboo, обеспечивая прозрачность и оперативность в процессе развертывания. Opsgenie используется для управления инцидентами, в случае возникновения проблем во время развертывания. DevOps инженеры сразу получают уведомления и могут оперативно реагировать на сбои.

Используются хуки Bitbucket, чтобы автоматически запускать сборку и тестирование при каждом коммите в репозиторий. Это обеспечивает раннее выявление ошибок и минимизацию времени между написанием кода и его проверкой. Результаты тестов и сборки видны в интерфейсе Bitbucket, что обеспечивает инженерам мгновенный обзор состояния кода.

Используется система мониторинга для отслеживания производительности приложения и выявления потенциальных проблем. Полученные данные интегрируются в Confluence, что позволяет командам эффективно анализировать результаты и предпринимать меры по оптимизации кода или архитектуры.

Новые процессы обеспечивают более эффективное управление проектами, улучшенную коммуникацию, сокращение времени между разработкой и развертыванием, а также повышение стабильности и надежности программного продукта.

# ЗАКЛЮЧЕНИЕ

Методика DevOps представляет собой мощный инструмент для современных организаций, стремящихся к более эффективной, гибкой и высококачественной поставке программного обеспечения. В ходе исследования данной методологии мы обнаружили, что DevOps не просто набор практик и инструментов, но скорее культурное и философское движение, направленное на объединение команд разработки и операций для достижения общей цели - улучшения жизненного цикла программного продукта.

Эволюция DevOps началась с осознания проблем в традиционной модели разработки и привнесла новые подходы, такие как непрерывная поставка, автоматизация процессов, итеративность и сильное внимание к коммуникации и сотрудничеству. Она подтверждена успехами многих крупных компаний, которые реализовали эту методику и достигли высокой степени гибкости, отзывчивости и стабильности в своих проектах.

Однако, внедрение DevOps не лишено сложностей. Это требует изменений в корпоративной культуре, обучения персонала, внедрения новых инструментов и технологий. Поэтому, при рассмотрении реализации DevOps, важно учитывать индивидуальные особенности каждой организации и внимательно планировать процесс внедрения.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. DevOps : что это такое простыми словами, где применяется технология – <https://blog.skillfactory.ru/glossary/devops/>
2. Что такое DevOps и зачем он нужен разработчикам – <https://education.yandex.ru/journal/chto-takoe-devops-i-zachem-on-nuzhen-razrabotchikam>
3. Что такое DevOps и зачем он нужен в IT-компании – <https://selectel.ru/blog/what-is-devops/>
4. DevOps и AWS - <https://aws.amazon.com/ru/devops/>
5. "DevOps Handbook" – Gene Kim, Patrick Debois, John Willis, and Jez Humble.
6. Ресурсы и отчеты по исследованиям в области DevOps. – <https://dora.dev/>