МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»
(РУТ (МИИТ)

Рабочая программа дисциплины (модуля),
как компонент образовательной программы
базового высшего образования
по направлению подготовки
09.03.01 Информатика и вычислительная техника,
утвержденной директором ФГАОУ ВО РУТ
(МИИТ) Париновым Д.В.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ)

Инфраструктура разработки IT-проектов

Направление подготовки: 09.03.01 Информатика и вычислительная техника

Направленность (профиль): Технологии разработки программного обеспечения

Форма обучения: Очная

Рабочая программа дисциплины (модуля) в виде электронного документа выгружена из единой корпоративной информационной системы управления университетом и соответствует оригиналу

Простая электронная подпись, выданная ФГАОУ ВО РУТ (МИИТ)
ID подписи: 937226
Подписал: директор центра Проневич Ольга Борисовна
Дата: 12.11.2026

1. Общие сведения о дисциплине (модуле).

Дисциплина формирует фундаментальную базу для будущего инженера по инфраструктуре и разработке программного обеспечения. В условиях тотального импортозамещения и перехода на отечественный стек – сертифицированные операционные системы, реляционные СУБД и платформы хостинга кода – рынок испытывает острый дефицит специалистов, понимающих жизненный цикл артефакта и умеющих выстраивать прозрачную командную работу. Студенты освоят системы контроля версий как единый источник истины, научатся проектировать стратегии ветвления, проводить рецензирование кода и поймут базовые принципы контейнеризации и непрерывной интеграции. Курс является пропедевтическим и закладывает архитектурный фундамент для последующего углубленного изучения автоматизации, операционных систем и мониторинга. На практике обучающиеся пройдут путь от индивидуальной работы с кодом до моделирования кросс-функциональной команды, создав связное портфолио с настроенным базовым конвейером сборки.

Целью освоения дисциплины является формирование у обучающихся системного понимания инфраструктуры разработки и практических навыков организации командного взаимодействия, версионирования кода и базовой автоматизации сборки программных продуктов с применением отечественного и открытого программного обеспечения.

Для достижения поставленной цели в рамках дисциплины решается комплекс задач, направленных на формирование у обучающихся способности: организовывать совместную разработку кода с использованием распределенных систем контроля версий, проектировать стратегии ветвления и регламенты рецензирования, автоматизировать локальную сборку и упаковку артефактов в контейнеры, а также выстраивать процессы планирования и ретроспективного анализа с применением гибких методологий.

2. Планируемые результаты обучения по дисциплине (модулю).

Перечень формируемых результатов освоения образовательной программы (компетенций) в результате обучения по дисциплине (модулю):

ДПК-БП-2 - Способен обеспечивать наблюдаемость, производительность, надёжность и эксплуатационную устойчивость распределённого бизнес-приложения;

ОПК-3 - Способен использовать современные информационные технологии и программно-аппаратные средства, в том числе отечественного производства, при решении задач профессиональной деятельности;

ППК-Р3 - Способен выполнять интеграцию программных модулей и компонентов и проверять работоспособность выпусков программного продукта;

ППК-Р6 - Способен участвовать в промышленной разработке программного обеспечения.

Обучение по дисциплине (модулю) предполагает, что по его результатам обучающийся будет:

Знать:

- архитектура распределенных систем контроля версий, включая концепции внутренних объектов, направленного ациклического графа коммитов и механизмов обеспечения целостности данных;
- модели ветвления и алгоритмы слияния изменений с учетом их влияния на читаемость истории и организацию параллельной разработки в команде;
- регламенты и культурные основы рецензирования кода, включая критерии качества запросов на слияние и принципы конструктивной обратной связи;
- фреймворки гибких методологий разработки, их артефакты, роли, события и метрики потока для управления жизненным циклом программного продукта;
- архитектура и жизненный цикл инструментов автоматизации сборки, включая механизмы управления зависимостями, плагинами и генерации финальных артефактов;
- принципы изоляции процессов и ресурсов в контейнеризации приложений, архитектура движков изоляции и методы оптимизации слоев образа;
- основы оркестрации контейнеров и декларативного описания многокомпонентных сред выполнения с использованием виртуальных сетей и томов данных;
- концепции непрерывной интеграции, включая архитектуру пайплайнов, типы исполнителей и триггеры автоматизации на основе событий репозитория;
- синтаксис и семантика декларативных языков описания конфигураций для написания пайплайнов автоматизации;

- подходы к управлению конфигурациями и переменными окружения, обеспечивающие изоляцию данных от исходного кода;

- принципы обеспечения базовой наблюдаемости систем, включая архитектурные паттерны сбора структурированных логов на этапе разработки;

- концепция «Инфраструктура как код», принципы идемпотентности и версионирования конфигурационных файлов;

- регламенты управления производственными инцидентами и структура инженерной документации для описания процессов развертывания;

- архитектура и специфика применения отечественных и импортозамещенных решений в области систем контроля версий и контейнеризации;

- правовые и этические нормы лицензирования программного обеспечения, включая специфику интеграции компонентов с открытым исходным кодом.

Уметь:

- уметь организовывать совместную разработку программного кода при помощи распределенной системы контроля версий в условиях параллельной работы кросс-функциональной команды с соблюдением строгих регламентов коммитов;

- уметь проектировать и применять стратегии ветвления при помощи моделей потоковой работы и политик защиты веток для изоляции разработки фич, подготовки релизов и оперативного исправления инцидентов;

- уметь организовывать процесс рецензирования кода при помощи механизма запросов на слияние в рамках культуры конструктивного ревью и защиты магистральных веток репозитория;

- уметь автоматизировать процессы сборки и упаковки артефактов при помощи декларативных систем управления зависимостями и средств контейнеризации для обеспечения воспроизводимости сред выполнения;

- уметь конструировать декларативные конвейеры непрерывной интеграции при помощи платформ хостинга кода и YAML-синтаксиса для автоматического запуска линтинга, тестирования и сборки при каждом событии_push в репозиторий;

- уметь выстраивать процессы планирования, отслеживания прогресса и ретроспективного анализа при помощи гибких методологий и метрик потока для управления жизненным циклом программного продукта.

3. Объем дисциплины (модуля).

3.1. Общая трудоемкость дисциплины (модуля).

Общая трудоемкость дисциплины (модуля) составляет 3 з.е. (108 академических часа(ов).

3.2. Объем дисциплины (модуля) в форме контактной работы обучающихся с педагогическими работниками и (или) лицами, привлекаемыми к реализации образовательной программы на иных условиях, при проведении учебных занятий:

Тип учебных занятий	Количество часов	
	Всего	Семестр №4
Контактная работа при проведении учебных занятий (всего):	64	64
В том числе:		
Занятия лекционного типа	32	32
Занятия семинарского типа	32	32

3.3. Объем дисциплины (модуля) в форме самостоятельной работы обучающихся, а также в форме контактной работы обучающихся с педагогическими работниками и (или) лицами, привлекаемыми к реализации образовательной программы на иных условиях, при проведении промежуточной аттестации составляет 44 академических часа (ов).

3.4. При обучении по индивидуальному учебному плану, в том числе при ускоренном обучении, объем дисциплины (модуля) может быть реализован полностью в форме самостоятельной работы обучающихся, а также в форме контактной работы обучающихся с педагогическими работниками и (или) лицами, привлекаемыми к реализации образовательной программы на иных условиях, при проведении промежуточной аттестации.

4. Содержание дисциплины (модуля).

4.1. Занятия лекционного типа.

№ п/п	Тематика лекционных занятий / краткое содержание
1	Введение в инфраструктуру разработки и гибкие методологии Рассматриваемые вопросы: - эволюция подходов к разработке и эксплуатации программного обеспечения; - фреймворки гибких методологий, их артефакты, роли и события; - интеграция гибких практик с инженерными процессами жизненного цикла.
2	Архитектура распределенных систем контроля версий Рассматриваемые вопросы:

№ п/п	Тематика лекционных занятий / краткое содержание
	- концепции внутренних объектов и направленного ациклического графа коммитов; - криптографические механизмы обеспечения целостности данных; - различия между централизованными и распределенными системами контроля версий.
3	Продвинутые стратегии ветвления и слияния Рассматриваемые вопросы: - алгоритмы слияния изменений и их влияние на читаемость истории; - модели потоковой работы для организации параллельной разработки; - разрешение конфликтов слияния и безопасный откат изменений.
4	Культура совместной разработки и рецензирование кода Рассматриваемые вопросы: - принципы конструктивного рецензирования в кросс-функциональных командах; - критерии качества запросов на слияние и работа с комментариями; - политики защиты веток и обязательные проверки перед слиянием.
5	Инструменты автоматизации сборки и управления зависимостями Рассматриваемые вопросы: - архитектура и жизненный цикл декларативных систем сборки; - механизмы управления зависимостями и плагинами; - генерация финальных артефактов и их версионирование.
6	Принципы контейнеризации приложений Рассматриваемые вопросы: - концепции изоляции процессов и ресурсов в контейнерах; - архитектура движков изоляции и различия между образами и контейнерами; - методы оптимизации слоев образа и многоступенчатая сборка.
7	Оркестрация контейнеров и многокомпонентные среды Рассматриваемые вопросы: - декларативное описание многокомпонентных сред выполнения; - использование виртуальных сетей и томов данных; - управление переменными окружения и конфигурациями.
8	Концепции непрерывной интеграции и доставки Рассматриваемые вопросы: - принципы непрерывной интеграции и их роль в автоматизации жизненного цикла; - архитектура пайплайнов и типы исполнителей; - триггеры автоматизации и события репозитория.
9	Декларативное описание пайплайнов автоматизации Рассматриваемые вопросы: - синтаксис и семантика декларативных языков для описания конфигураций; - специфика написания пайплайнов для корпоративных платформ хостинга кода; - стадии, задачи и артефакты в декларативных пайплайнах.
10	Интеграция безопасности в процессы сборки Рассматриваемые вопросы: - принципы обеспечения безопасности на ранних этапах жизненного цикла; - инструменты статического анализа кода и уязвимостей зависимостей; - автоматизация проверок безопасности в пайплайнах.
11	Управление конфигурациями и секретами Рассматриваемые вопросы: - подходы к управлению переменными окружения; - обеспечение изоляции конфиденциальных данных от исходного кода; - защищенные переменные и хранилища секретов.
12	Стратегии безопасного развертывания и управление релизами Рассматриваемые вопросы:

№ п/п	Тематика лекционных занятий / краткое содержание
	- методы минимизации рисков при развертывании; - механизмы автоматического отката при сбоях; - практики управления релизами через тегирование.
13	Обеспечение наблюдаемости и мониторинг систем Рассматриваемые вопросы: - принципы наблюдаемости и архитектурные паттерны сбора логов; - агрегация метрик и настройка алертинга; - централизованное логирование и анализ производительности.
14	Управление инцидентами и Infrastructure as Code Рассматриваемые вопросы: - регламенты реагирования на инциденты и структура инженерной документации; - принципы культуры анализа сбоев без поиска виноватых; - концепция инфраструктуры как кода и идемпотентность конфигураций.
15	Импортозамещение в инфраструктуре разработки Рассматриваемые вопросы: - специфика применения отечественных решений в автоматизации и контейнеризации; - архитектура и особенности корпоративных платформ хостинга кода и СУБД; - правовые и этические нормы лицензирования программного обеспечения.
16	Интеграция компонентов и требования реестра Рассматриваемые вопросы: - специфика интеграции компонентов с открытым исходным кодом; - требования единого реестра российского программного обеспечения; - ретроспективный анализ метрик потока и защита итогового портфолио.

4.2. Занятия семинарского типа.

Лабораторные работы

№ п/п	Наименование лабораторных работ / краткое содержание
1	Инициализация репозитория и базовые операции Студент инициализирует локальный репозиторий и настраивает глобальные параметры пользователя. Он создает первые файлы проекта и фиксирует изменения в рабочем каталоге с осмысленными сообщениями. Обучающийся анализирует историю коммитов и изучает три состояния файлов в системе контроля версий. Работа завершается проверкой целостности локального графа изменений.
2	Ветвление и стратегии слияния Студент создает локальные ветки для разработки новых функций. Он выполняет слияние изменений с использованием различных стратегий объединения кода. Обучающийся анализирует полученный граф коммитов и оценивает читаемость истории разработки. Результатом является корректно объединенная ветка без потери данных.
3	Разрешение конфликтов и откат изменений Студент моделирует ситуацию одновременного изменения одних и тех же строк кода в разных ветках. Он вручную разрешает конфликты слияния, выбирая корректную версию кода. Обучающийся применяет команды отката для исправления ошибочных коммитов без нарушения общей истории. Работа завершается успешным слиянием и очисткой репозитория от мусорных данных.
4	Работа с удаленным репозиторием Студент создает проект на платформе хостинга кода и настраивает связь с локальным репозиторием. Он выполняет операции синхронизации локальных и удаленных изменений.

№ п/п	Наименование лабораторных работ / краткое содержание
	Обучающийся настраивает игнорирование временных файлов через конфигурационный файл. Итогом является полностью синхронизированный удаленный репозиторий с актуальной историей.
5	Моделирование командной работы и декомпозиция задач Студент формирует виртуальную кросс-функциональную команду и распределяет роли. Он декомпозирует гипотетический бэклог продукта на конкретные инженерные задачи. Обучающийся создает карточки задач на цифровой доске и оценивает их трудоемкость. Результатом является структурированный план работ на первый спринт.
6	Планирование спринта и определение критериев готовности Студент проводит церемонию планирования и выбирает задачи для выполнения в текущем спринте. Он формулирует четкие критерии готовности для каждой задачи. Обучающийся настраивает рабочий процесс на доске задач для отслеживания статусов. Работа завершается утверждением плана спринта и назначением ответственных.
7	Проектирование стратегии ветвления Студент проектирует схему ветвления для конкретного проекта, выделяя магистральные и релизные ветки. Он документирует правила создания и слияния для каждого типа веток. Обучающийся настраивает шаблоны запросов на слияние в интерфейсе платформы. Итогом является утвержденный регламент командной работы с кодом.
8	Проведение рецензирования кода Студент создает запрос на слияние ветки в основную ветку разработки. Он заполняет шаблон описания изменений и назначает ревьюеров из числа коллег. Обучающийся оставляет конструктивные комментарии к коду и вносит правки на основе полученной обратной связи. Работа завершается успешным одобрением и слиянием запроса.
9	Настройка проекта и управление зависимостями Студент выбирает инструмент сборки и инициализирует конфигурационный файл проекта. Он добавляет необходимые внешние зависимости и настраивает версии плагинов. Обучающийся проверяет корректность загрузки артефактов из репозитория. Результатом является готовый к сборке проект с управляемыми зависимостями.
10	Автоматизация локальной сборки и тестирования Студент настраивает цели сборки для компиляции кода и запуска модульных тестов. Он запускает процесс сборки из командной строки и анализирует вывод консоли. Обучающийся исправляет ошибки компиляции и добивается успешного прохождения всех тестов. Итогом является готовый артефакт, сохраненный в локальной директории.
11	Проектирование и создание образа контейнера Студент анализирует требования к среде выполнения приложения и пишет конфигурационный файл образа. Он применяет практику многоступенчатой сборки для минимизации размера итогового артефакта. Обучающийся собирает образ локально и проверяет его на наличие уязвимостей базовых слоев. Работа завершается созданием оптимизированного и безопасного образа приложения.
12	Оркестрация многокомпонентного окружения Студент пишет декларативный файл для одновременного запуска приложения и базы данных. Он настраивает виртуальную сеть и тома для сохранения постоянных данных. Обучающийся запускает стек сервисов и проверяет их взаимодействие между собой. Результатом является корректно работающее многокомпонентное окружение на локальной машине.
13	Проектирование конвейера непрерывной интеграции Студент анализирует этапы жизненного цикла и проектирует структуру пайплайна. Он выбирает необходимые стадии для анализа кода, сборки и тестирования. Обучающийся описывает логику триггеров для запуска задач при событиях pushes. Итогом является архитектурная схема конвейера, готовая к программной реализации.
14	Реализация декларативного пайплайна Студент создает конфигурационный файл пайплайна в декларативном формате. Он описывает

№ п/п	Наименование лабораторных работ / краткое содержание
	стадии сборки и запуска тестов, привязывая их к конкретным веткам. Обучающийся фиксирует изменения в репозитории и наблюдает за автоматическим запуском задач. Работа завершается успешным прохождением всех стадий конвейера без ручного вмешательства.
15	Настройка политик защиты веток Студент активирует правила защиты для магистральных веток репозитория. Он настраивает обязательное успешное прохождение пайплайна перед разрешением слияния. Обучающийся проверяет корректность работы ограничений, пытаясь выполнить запрещенные действия. Результатом является защищенный репозиторий, исключающий попадание некорректного кода в основную ветку.
16	Проектирование конвейера непрерывной интеграции Студент анализирует историю коммитов и метрики потока за весь период работы. Он готовит краткий отчет о проделанной работе и выявленных проблемах. Обучающийся демонстрирует работоспособность настроенного конвейера и репозитория. Работа завершается защитой итогового связного портфолио перед преподавателем.

4.3. Самостоятельная работа обучающихся.

№ п/п	Вид самостоятельной работы
1	Изучение рекомендованной литературы.
2	Подготовка к лабораторным работам.
3	Подготовка к промежуточной аттестации.
4	Подготовка к текущему контролю.

5. Перечень изданий, которые рекомендуется использовать при освоении дисциплины (модуля).

№ п/п	Библиографическое описание	Место доступа
1	Чакон С., Штрауб Б. Pro Git. Полное руководство по распределенной системе контроля версий [Электронный ресурс].	Официальная русскоязычная версия книги. – URL: https://git-scm.com/book/ru/v2 (дата обращения: 09.06.2026)
2	Херинг, М. DevOps для современного предприятия : учебное пособие / М. Херинг ; перевод с английского М. А. Райтмана.. — Москва : ДМК Пресс, 2020. — 232 с. — ISBN 978-5-97060-836-4. — Текст : электронный	Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/140580 (дата обращения: 19.06.2026)
3	Жматов, Д. В. GIT: Создание прочной основы для эффективной разработки : учебное пособие / Д. В. Жматов. — Москва : РТУ МИРЭА, [б. г.]. — Часть 1 — 2024. — 114 с. — ISBN 978-5-7339-2345-1. — Текст : электронный	Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/457052 (дата обращения: 19.06.2026)

4	Рощин, П. Г. Командная разработка программного обеспечения с помощью системы контроля версий Git: Конспект лекций : учебное пособие / П. Г. Рощин. — Москва : НИЯУ МИФИ, 2022. — 108 с. — ISBN 978-5-7262-2846-4. — Текст : электронный	Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/355550 (дата обращения: 19.06.2026)

6. Перечень современных профессиональных баз данных и информационных справочных систем, которые могут использоваться при освоении дисциплины (модуля).

документация корпоративных платформ хостинга кода и CI/CD;

документация отечественных реляционных СУБД;

электронно-библиотечные системы;

официальная документация движков контейнеризации.

7. Перечень лицензионного и свободно распространяемого программного обеспечения, в том числе отечественного производства, необходимого для освоения дисциплины (модуля).

операционные системы: сертифицированные отечественные дистрибутивы Linux;

офисные пакеты: отечественные офисные приложения из реестра РФ;

платформы хостинга кода: корпоративные системы управления репозиториями и конвейерами сборки;

инструменты сборки: декларативные системы управления зависимостями и артефактами;

средства контейнеризации: движки изоляции процессов на уровне ОС и оркестрации локальных сред;

СУБД: реляционные системы управления базами данных из реестра отечественного ПО;

трекинг задач: корпоративные системы управления проектами и гибкими методологиями;

скриптовые языки: интерпретируемые языки для автоматизации рутинных операций и написания CI-сценариев.

8. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю).

Учебные аудитории для проведения учебных занятий, оснащенные компьютерной техникой и наборами демонстрационного оборудования.

Для лабораторных занятий – наличие персональных компьютеров вычислительного класса.

9. Форма промежуточной аттестации:

Экзамен в 4 семестре.

10. Оценочные материалы.

Оценочные материалы, применяемые при проведении промежуточной аттестации, разрабатываются в соответствии с локальным нормативным актом РУТ (МИИТ).

Авторы:

старший преподаватель кафедры
«Цифровые технологии управления
транспортными процессами»
Института железнодорожного
транспорта И.С. Разживайкин

Согласовано:

Заведующий кафедрой ЦТУТП В.Е. Нутович

Директор ЦС IT - программ ВИШ О.Б. Проневич

Председатель учебно-методической
комиссии Н.А. Андриянова