МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
Учебно-методическое объединение по образованию
в области информатики и радиоэлектроники

УТВЕРЖДЕНО
Первым заместителем Министра образования
Республики Беларусь
А.Г. Бахановичем
05.05.2026

Регистрационный **№ 6-05-06-096/пр.**

ИНСТРУМЕНТЫ ПОДДЕРЖКИ ПРОМЫШЛЕННОЙ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Примерная учебная программа по учебной дисциплине
для специальности
6-05-0611-01 Информационные системы и технологии

СОГЛАСОВАНО
Председателем Учебно-
методического объединения
по образованию в области
информатики и радиоэлектроники
В.А. Богушем

СОГЛАСОВАНО
Начальником Главного управления
профессионального образования
Министерства образования
Республики Беларусь
С.Н. Пищовым

СОГЛАСОВАНО
Проректором по научно-методической
работе Государственного учреждения
образования «Республиканский
институт высшей школы»
И.В. Титовичем

Эксперт-нормоконтролер
Т.А. Богомья

Минск 2026

СОСТАВИТЕЛИ:
В.Н.Комличенко, доцент кафедры экономической информатики учреждения образования «Белорусский государственный университет информатики и радиоэлектроники», кандидат технических наук, доцент;
А.А.Ефремов, заведующий кафедрой экономической информатики учреждения образования «Белорусский государственный университет информатики и радиоэлектроники», кандидат экономических наук, доцент;
Н.О.Петрович, старший преподаватель кафедры экономической информатики учреждения образования «Белорусский государственный университет информатики и радиоэлектроники», магистр экономических наук, исследователь технических наук;
Д.А.Сторожев, старший преподаватель кафедры экономической информатики учреждения образования «Белорусский государственный университет информатики и радиоэлектроники», магистр экономических наук, исследователь технических наук

РЕЦЕНЗЕНТЫ:
Кафедра автоматизированных систем управления производством учреждения образования «Белорусский государственный аграрный технический университет» (протокол № 5 от 27.11.2025);
В.А.Грушев, заместитель директора по производству иностранного унитарного научно-производственного предприятия «САМСОЛЮШНС», кандидат технических наук

РЕКОМЕНДОВАНА К УТВЕРЖДЕНИЮ В КАЧЕСТВЕ ПРИМЕРНОЙ:
Кафедрой экономической информатики учреждения образования «Белорусский государственный университет информатики и радиоэлектроники» (протокол № 5 от 05.11.2025);
Научно-методическим советом учреждения образования «Белорусский государственный университет информатики и радиоэлектроники» (протокол № 4 от 19.12.2025);
Научно-методическим советом по прикладным информационным системам и технологиям Учебно-методического объединения по образованию в области информатики и радиоэлектроники (протокол № 4 от 10.12.2025)

Ответственный за редакцию: С.С.Шишпаронок

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

ХАРАКТЕРИСТИКА УЧЕБНОЙ ДИСЦИПЛИНЫ

Примерная учебная программа по учебной дисциплине «Инструменты поддержки промышленной разработки программного обеспечения» разработана для студентов учреждений высшего образования, обучающихся по специальности 6-05-0611-01 «Информационные системы и технологии», в соответствии с требованиями образовательного стандарта общего высшего образования и примерного учебного плана вышеуказанной специальности.

Учебная дисциплина «Инструменты поддержки промышленной разработки программного обеспечения» является одной из дисциплин по информационным системам, изучаемых студентами данной специальности, изучение которой способствует развитию знаний, полученных в процессе изучения объектно-ориентированного программирования и дисциплин конструирования информационного обеспечения.

Освоение учебной дисциплины «Инструменты поддержки промышленной разработки программного обеспечения» обеспечивает подготовку специалиста, владеющего базовыми знаниями и основными практическими навыками в области методов инструментов и средств промышленной разработки программного обеспечения.

Воспитательное значение учебной дисциплины «Инструменты поддержки промышленной разработки программного обеспечения» заключается в формировании у обучающихся математической культуры и научного мировоззрения; развитии исследовательских умений, аналитических способностей, креативности, необходимых для решения научных и практических задач; развитии познавательных способностей и активности: творческой инициативы, самостоятельности, ответственности и организованности, работе в коллективе; формировании способностей к саморазвитию, самосовершенствованию и самореализации.

Изучение данной учебной дисциплины способствует созданию условий для формирования интеллектуально развитой личности обучающегося, которой присущи стремление к профессиональному совершенствованию, активному участию в экономической и социально-культурной жизни страны, гражданская ответственность и патриотизм.

ЦЕЛЬ, ЗАДАЧИ УЧЕБНОЙ ДИСЦИПЛИНЫ

Цель учебной дисциплины: сформировать у обучающихся системное представление о принципах промышленной разработки программного обеспечения, включая архитектурное проектирование, командную разработку, автоматизацию процессов сборки, тестирования, интеграции, доставки и эксплуатации программных продуктов с использованием современных инструментов и технологий CI/CD.

Задачи учебной дисциплины:

ознакомить обучающихся с эффективными архитектурными подходами к разработке программного обеспечения, научить применять системы контроля версий для совместной разработки программного обеспечения;

раскрыть концепцию непрерывной интеграции (CI), её роль и основные задачи в современной разработке ПО;

сформировать навыки написания и запуска модульных и интеграционных тестов и средств логирования и обеспечить на этой основе качественную разработку ПО;

ознакомить с базовыми технологиями и средствами автоматизации процессов непрерывной интеграции, поставки и развертывания программного обеспечения в промышленном программировании.

Базовыми учебными дисциплинами для учебной дисциплины «Инструменты поддержки промышленной разработки программного обеспечения» являются такие учебные дисциплины, как «Объектно-ориентированное проектирование и программирование», «Программирование сетевых приложений». В свою очередь, учебная дисциплина «Инструменты поддержки промышленной разработки программного обеспечения» является базой для такой учебной дисциплины компонента учреждения образования, как «Распределенные информационные системы».

ТРЕБОВАНИЯ К УРОВНЮ ОСВОЕНИЯ СОДЕРЖАНИЯ УЧЕБНОЙ ДИСЦИПЛИНЫ

В результате изучения учебной дисциплины «Инструменты поддержки промышленной разработки программного обеспечения» формируются следующие компетенции:

универсальная: решать стандартные задачи профессиональной деятельности на основе применения информационно-коммуникационных технологий;

базовая профессиональная: применять современные инструменты промышленной разработки программного обеспечения.

В результате изучения учебной дисциплины «Инструменты поддержки промышленной разработки программного обеспечения» обучающийся должен:

знать:

принципы, методы и инструменты построения слабо связанных архитектур;

основы работы с системами контроля версий и командной разработки;

структуру и конфигурирование сборочных систем и основы концепции CI/CD и DevOps;

методы обеспечения качества, тестирования, логирования разрабатываемого ПО;

уметь:

настраивать и использовать инструментальные средства командной разработки ПО;

конфигурировать инфраструктуру разработки для автоматизации сборки, управления зависимостями;

применять методы и средства разработки качественного программного кода, в их числе автоматизированное тестирование и логирование;

создавать и запускать процессы непрерывной интеграции, поставки и развертывания программного обеспечения;

иметь навык:

работы с основными методами и техниками программной инженерии, базовыми технологиями и средствами обеспечения жизненного цикла разработки программных приложений;

разработки качественного программного обеспечения;

командной разработки промышленных проектов с использованием современных средств автоматизации.

Примерная учебная программа рассчитана на 108 учебных часов, из них – 40 аудиторных. Примерное распределение аудиторных часов по видам занятий: лекции – 16 часов, лабораторные занятия – 24 часа.

ПРИМЕРНЫЙ ТЕМАТИЧЕСКИЙ ПЛАН

Название раздела, темы	Всего аудиторных часов	Лекции	Лабораторные занятия
Тема 1. Введение в промышленную разработку ПО. Проектирование и преимущества слабо связанных архитектур	6	2	4
Тема 2. Системы контроля версий как основа промышленной разработки и процессов CI/CD	2	2	
Тема 3. Инструменты и средства разработки сценариев в системах непрерывной интеграции (ANT, Maven, Gradle)	6	2	4
Тема 4. Непрерывная интеграция CI как основа современной концепции разработки программных проектов, настройка Jenkins, язык сценариев и базовый конвейер CI/CD	6	2	4
Тема 5. Технология и инструменты автоматизации CI/CD процессов непрерывной интеграции и доставки (Jenkins, GitHub Actions)	6	2	4
Тема 6. Качество промышленного кода: статический анализ, обработка исключений и логгирование	6	2	4
Тема 7. Методы и инструменты промышленного развертывания и эксплуатации программных систем в контексте CI/CD	2	2	
Тема 8. Ключевые инструменты наблюдаемости и мониторинга в CI/CD	6	2	4
Итого	**40**	**16**	**24**

СОДЕРЖАНИЕ УЧЕБНОГО МАТЕРИАЛА

Тема 1. ВВЕДЕНИЕ В ПРОМЫШЛЕННУЮ РАЗРАБОТКУ ПО. ПРОЕКТИРОВАНИЕ И ПРЕИМУЩЕСТВА СЛАБО СВЯЗАННЫХ АРХИТЕКТУР

Введение в учебную дисциплину. Отличия промышленной разработки от учебных и исследовательских проектов: масштаб, командная работа, требования к качеству и поддержке. Особенности промышленной разработки ПО. Управление сложностью: необходимость архитектурных решений и стандартизации. Совместная работа команд: распределённая разработка, контроль версий, процессы ревью. Требования к качеству: тестируемость, поддерживаемость, документированность

Принципы построения слабо связанных архитектур: модульность: разделение системы на независимые компоненты. Инверсия управления (IoC): передача контроля от компонентов к контейнеру или фреймворку, внедрение зависимостей (DI): гибкость и тестируемость за счет подмены зависимостей. Применение основных принципов, в их числе SOLID, как основа устойчивости архитектуры ПО.

Преимущества слабой связности: возможность изолированного тестирования модулей, добавление новых функций без изменения существующего кода, локализация ошибок и минимизация их влияния на систему, масштабируемость: адаптация архитектуры к росту нагрузки и функциональности. Связь архитектурных решений с CI/CD: независимость модулей упрощает автоматизацию сборки, тестирования и доставки.

Тема 2. СИСТЕМЫ КОНТРОЛЯ ВЕРСИЙ КАК ОСНОВА ПРОМЫШЛЕННОЙ РАЗРАБОТКИ И ПРОЦЕССОВ CI/CD

Назначение систем контроля версий. Отслеживание изменений в коде и документации. Совместная работа команд разработчиков в распределённых проектах. Возможность отката к стабильным версиям. Ветвление как инструмент параллельной разработки.

Сравнение моделей контроля версий. Централизованная модель (Subversion): единый сервер, контроль доступа, ограничения масштабируемости. Распределённая модель (Git): локальные копии, гибкость ветвления, работа офлайн. Преимущества Git для CI/CD процессов.

Работа с ветками и командная разработка. Стратегии ветвления: Git Flow, Feature Branches, Trunk-Based Development. Разрешение конфликтов при слиянии. Практики командной разработки: pull requests, code review, continuous integration checks.

Интеграция систем контроля версий в CI/CD. Webhooks и триггеры: запуск сборки при коммите или pull request. Автоматизация процессов: сборка, тестирование, деплой из репозитория. Связь Git с Jenkins, GitHub Actions, GitLab CI/CD. Управление артефактами и версиями в CI/CD pipeline.

Практики промышленного применения. Code review как элемент обеспечения качества и стандартизации кода. Использование Git для

управления зависимостями и конфигурациями. Примеры интеграции: автоматическая проверка стиля кода, запуск тестов при pull request. Роль систем контроля версий в DevOps и Agile-подходах. Современные тенденции. GitOps: управление инфраструктурой через Git-репозитории. Интеграция с системами безопасности (DevSecOps).

Тема 3. ИНСТРУМЕНТЫ И СРЕДСТВА РАЗРАБОТКИ СЦЕНАРИЕВ В СИСТЕМАХ НЕПРЕРЫВНОЙ ИНТЕГРАЦИИ (ANT, Maven, Gradle)

Назначение сборочных систем: автоматизация компиляции, тестирования, упаковки и доставки ПО. Роль сборочных систем в промышленной разработке и CI/CD процессах.

Apache Ant. Язык сценариев и структура проекта. Основные элементы build.xml: targets, tasks, properties. Процедурный подход к описанию сборки. Примеры автоматизации: компиляция, запуск тестов, упаковка артефактов.

Apache Maven. Структура Maven-проекта: pom.xml, зависимости, плагины. Декларативный подход к описанию сборки. Управление зависимостями и репозитории артефактов. Профили сборки, цели (goals) и жизненный цикл проекта. Интеграция Maven с IDE (IntelliJ IDEA, Eclipse) и CI-системами (Jenkins, GitHub Actions).

Сравнение Ant и Maven. Процедурный vs декларативный подход. Гибкость Ant и стандартизированность Maven. Эволюция инструментов сборки в промышленной разработке. Практики промышленного применения. Использование Ant и Maven в CI/CD конвейерах. Автоматизация тестирования и генерации отчётов. Управление зависимостями и версиями библиотек в командной разработке. Связь сборочных систем с DevOps практиками (build → test → deploy). Перспективы и современные альтернативы. Gradle как современный инструмент сборки. Тенденции развития инструментов автоматизации в промышленном программировании.

Тема 4. НЕПРЕРЫВНАЯ ИНТЕГРАЦИЯ CI КАК ОСНОВА СОВРЕМЕННОЙ КОНЦЕПЦИИ РАЗРАБОТКИ ПРОГРАММНЫХ ПРОЕКТОВ, НАСТРОЙКА JENKINS, ЯЗЫК СЦЕНАРИЕВ И БАЗОВЫЙ КОНВЕЙЕР CI/CD

Понятие CI, автоматическая сборка, тестирование, проверка при каждом изменении. Цели CI: раннее выявление ошибок, ускорение обратной связи, стабильность кода. Тестирование в CI, виды тестирования: модульное, интеграционное, регрессионное. Роль автоматизированного тестирования в CI. Генерация и анализ отчётов о тестах. Основные этапы CI-процесса: сборка проекта, запуск тестов, анализ результатов.

Возможные триггеры запуска: коммит в систему контроля версий, расписание, ручной запуск. Интеграция CI с системами контроля версий (GIT GitHub, GitLab).

Jenkins как инструмент CI/CD. Архитектура Jenkins: master/agent, плагины. Настройка базового конвейера CI/CD. Анализ логов сборки и тестов.

Groovy и Jenkins pipelines. Роль языка Groovy в описании Jenkins pipelines (DSL). Декларативный и скриптовый подходы к созданию конвейеров.

Примеры простых Jenkinsfile на Groovy: этапы build → test → deploy. Использование Groovy для расширения функциональности и интеграции с внешними системами.

CI в контексте DevOps и Agile. Роль CI в ускорении итераций разработки. CI как фундамент для CD и практик DevOps. Влияние CI на командную работу и качество промышленного кода.

Тема 5. ТЕХНОЛОГИЯ И ИНСТРУМЕНТЫ АВТОМАТИЗАЦИИ CI/CD ПРОЦЕССОВ НЕПРЕРЫВНОЙ ИНТЕГРАЦИИ И ДОСТАВКИ (JENKINS, GITHUB ACTIONS)

Отличие CI от CD: CI: автоматическая сборка, тестирование, проверка при каждом изменении. CD: автоматическая доставка и деплой, rollback при ошибках.

Архитектура: master/agent, плагины, интеграция с системами контроля версий. Язык разработки и основные команды языка Groovy DSL, пайплайны (Pipeline DSL: этапы, шаги (команды), агенты). Настройка пайплайна: сборка → тест → деплой.

GitHub Actions: Workflow-файлы: структура, шаги, условия выполнения. Jobs и runners: настройка окружений. Интеграция с Maven, Docker, тестовыми фреймворками. Автоматизация деплоя в облачные среды (GitHub Pages, Heroku, AWS). Безопасность: управление секретами, ограничение прав доступа. Сравнение Jenkins и GitHub Actions: Jenkins: гибкость, расширяемость, использование Groovy DSL, интеграция в корпоративные среды. GitHub Actions: простота настройки, облачная модель, тесная интеграция с GitHub.

Расширенные возможности: параллельные шаги, условия выполнения, интеграция с Docker и Kubernetes.

Тема 6. КАЧЕСТВО ПРОМЫШЛЕННОГО КОДА: СТАТИЧЕСКИЙ АНАЛИЗ, ОБРАБОТКА ИСКЛЮЧЕНИЙ И ЛОГГИРОВАНИЕ.

Роль статического анализа в промышленной разработке ПО. Инструменты анализа качества кода: SonarQube, PMD, Checkstyle, SpotBugs. Типичные ошибки и нарушения стиля в Java-коде. Обработка исключений: виды исключений, корректное проектирование, типичные ошибки (пустые catch, игнорирование ошибок, чрезмерное использование checked exceptions).

Логгирование: роль в промышленном коде, уровни логов (INFO, WARN, ERROR, DEBUG), best practices, форматирование (настройка шаблонов), ротация логов (автоматическое архивирование и удаление старых логов.), проверка правил логгирования средствами статического анализа. Практика выявления проблем: настройка правил в SonarQube/PMD/Checkstyle. Исправление нарушений и повторный анализ. Влияние обработки исключений и логгирования на качество и сопровождаемость ПО. Конфигурация Logback через XML. Интеграция логирования с CI/CD и мониторингом. Использование логов для отладки и аудита.

Тема 7. МЕТОДЫ И ИНСТРУМЕНТЫ ПРОМЫШЛЕННОГО РАЗВЕРТЫВАНИЯ И ЭКСПЛУАТАЦИИ ПРОГРАММНЫХ СИСТЕМ В КОНТЕКСТЕ CI/CD

Ручное развертывание, особенности, ограничения и риски. Скриптовое развертывание. Архивная доставка (ZIP, JAR, WAR): традиционные подходы и их недостатки в масштабируемых системах. Использование систем управления конфигурациями. Применение Jenkins: настройка пайплайнов для автоматического деплоя, GitHub Actions, GitLab CI/CD: облачные сценарии доставки и интеграции. Связь систем контроля версий (Git) с процессами развертывания. Использование контейнеров в DevOps и CI/CD. Образы и контейнеры: концепция, преимущества (портативность, изоляция, масштабируемость). Dockerfile: описание окружения и зависимостей. Docker Compose: управление многоконтейнерными приложениями, настройка сетей и томов. Kubernetes: поды, сервисы, деплойменты, YAML-манифестёы: декларативное описание инфраструктуры. Интеграция Kubernetes с CI/CD конвейерами. Современные тенденции развития управления инфраструктурой, облачные среды, безопасность.

Тема 8. КЛЮЧЕВЫЕ ИНСТРУМЕНТЫ НАБЛЮДАЕМОСТИ И МОНИТОРИНГА В CI/CD

Роль наблюдаемости в промышленной разработке и CI/CD. Понятие наблюдаемости (observability) и отличие от мониторинга. Значение метрик, логов и трассировок для качества и стабильности ПО. Наблюдаемость как часть DevOps и Site Reliability Engineering (SRE).

Prometheus: сбор и хранение метрик. Архитектура Prometheus: сервер, exporters, targets. Модель данных и язык запросов PromQL. Настройка мониторинга сервисов и инфраструктуры. Алертинг: правила оповещений и интеграция с системами уведомлений.

Grafana: визуализация и аналитика. Архитектура и возможности Grafana Подключение к источникам данных (Prometheus, Loki, Elastic и др.). Создание дашбордов и панелей мониторинга. Настройка алертов и интеграция с CI/CD процессами.

Интеграция наблюдаемости в CI/CD pipeline. Автоматическая проверка метрик после деплоя. Мониторинг производительности и доступности сервисов. Использование дашбордов для анализа качества релизов. Реакция на инциденты: алерты, rollback, автоматизация отката.

Практика промышленного применения. Настройка мониторинга микросервисов и контейнеризированных приложений. Использование Prometheus и Grafana в Kubernetes-средах. Примеры метрик: время отклика, нагрузка на CPU/память, ошибки запросов. Построение дашбордов для команд разработки и эксплуатации. Тенденции развития мониторинга в DevOps и SRE.

ИНФОРМАЦИОННО-МЕТОДИЧЕСКАЯ ЧАСТЬ

ЛИТЕРАТУРА

ОСНОВНАЯ

1. Таненбаум, Э. С. Распределенные системы. Принципы и парадигмы / Э. С. Таненбаум, М. В. Стеен. – Москва : ДМК-Пресс, 2021. – 584 с.
2. Язык IDEF0: стандарт. Методология функционального моделирования idef0. [Электронный ресурс]. – Режим доступа: https://www.amspur.ru/idef. – Дата доступа: 15.10.2025.
3. BPwin 4.0 уроки примеры задачи [Электронный ресурс]. – Режим доступа: http://specialf.narod.ru/bpwin/urok.html#_Toc483892229. – Дата доступа: 15.10.2025.
4. ГОСТ 19.70190 «ЕСПД. СХЕМЫ АЛГОРИТМОВ, ПРОГРАММ, ДАННЫХ И СИСТЕМ. ОБОЗНАЧЕНИЯ УСЛОВНЫЕ И ПРАВИЛА ВЫПОЛНЕНИЯ» [Электронный ресурс]. – Режим доступа: https://rostest.info/gost/001.001.080.050/gost-19.701-90/?ysclid=ll9p218gjt78611663. – Дата доступа: 15.10.2025.
5. Буч, Г. Объектно-ориентированный анализ и проектирование с примерами приложений (UML 2) / Г. Буч. – Москва : Вильямс, 2010. – 720 с.
6. Jenkins [Электронный ресурс]. – Режим доступа: https://track.habr.com/frontend/skill/jenkins. –Дата доступа: 15.10.2025
7. UML [Электронный ресурс]. – Режим доступа: https://coderlessons.com/?s=UML. –Дата доступа: 15.10.2025.
8. Блинов, И. Н. Java from EPAM : учебно-методическое пособие / И. Н. Блинов, В. С. Романчик. – 2-е издание. – Минск : Четыре четверти, 2021. – 560 с.
9. Блинов, И. Н. Java. Методы программирования : учебно-методическое пособие / И. Н. Блинов, В. С. Романчик. – Минск : Четыре четверти, 2013. – 896 с.
10. Руководство по языку программирования Java [Электронный ресурс]. – Режим доступа: https://metanit.com/java/tutorial/. – Дата доступа: 15.10.2025.
11. Основы паттернов проектирования [Электронный ресурс]. – Режим доступа: https://metanit.com/sharp/patterns/1.1.php. – Дата доступа: 15.10.2025.
12. Guide to Java Reflection [Электронный ресурс]. – Режим доступа: https://www.baeldung.com/java-reflection. – Дата доступа: 15.10.2025.
13. Reflection (Рефлексия) – основы [Электронный ресурс]. – Режим доступа: https://vertex-academy.com/tutorials/ru/reflection-api-v-java-chast1/. – Дата доступа: 15.10.2025.
14. Учебные руководства по Java Reflection API [Электронный ресурс]. – Режим доступа: https://coderlessons.com/tutorials/kompiuternoe-programmirovanie/uchebnik-java/50-uchebnye-rukovodstva-po-java-reflection-api/. –Дата доступа: 15.10.2025.

15. Надежное руководство по принципам SOLID [Электронный ресурс]. – Режим доступа: https://for-each.dev/lessons/b/-solid-principles. – Дата доступа: 15.10.2025.

16. XML Tutorial [Электронный ресурс]. – Режим доступа: https://www.tutorialspoint.com/xml/index.htm. – Дата доступа: 15.10.2025.

17. XML Tutorial [Электронный ресурс]. – Режим доступа: https://www.w3schools.com/xml/default.asp. – Дата доступа: 15.10.2025.

18. Спецификация XSLT 1.0 [Электронный ресурс]. – Режим доступа: https://xsltdev.ru/tr/. – Дата доступа: 15.10.2025.

19. Пышкин, Е. В. Модульное тестирование программного обеспечения. Профессиональный базовый курс с практикой на JUnit / Е. В. Пышкин, М. Глухих. – Санкт-Петербург : Проф. лит. : АйТи-Подготовка, 2015. –617 с.

20. JUnit 5 tutorial – Learn how to write unit tests [Электронный ресурс]. – Режим доступа: https://www.vogella.com/tutorials/JUnit/article.html#junit5. – Дата доступа: 15.10.2025.

21. Мартин, Р. Чистый код. Создание, анализ и рефакторинг / Р. Мартин. – Санкт-Петербург : Питер, 2023. – 400 с.

22. Pro GIT (2nd Edition) [Электронный ресурс]. – Режим доступа: https://git-scm.com/book/ru/v2. – Дата доступа: 15.10.2025.

23. Учебник по Jenkins [Электронный ресурс]. – Режим доступа: https://coderlessons.com/tutorials/devops/uchebnik-po-jenkins/uchebnik-po-jenkins. – Дата доступа: 15.10.2025.

24. Непрерывная интеграция. Краткое руководство [Электронный ресурс]. – Режим доступа: https://coderlessons.com/tutorials/kachestvo-programmnogo-obespecheniia/uznaite-nepreryvnuiu-integratsiiu/nepreryvnaia-integratsiia-kratkoe-rukovodstvo. – Дата доступа: 15.10.2025.

25. Учебное пособие по DevOps [Электронный ресурс]. – Режим доступа: https://coderlessons.com/tutorials/devops/kratkii-kurs-po-devops/1-uchebnoe-posobie-po-devops. – Дата доступа: 15.10.2025.

ДОПОЛНИТЕЛЬНАЯ

26. Денисов, А. А. Современные проблемы системного анализа : информационные основы : учебное пособие /А. А. Денисов. – Санкт-Петербург : СПбГТУ, 2005. – 295 с.

27. Макконнелл, С. Совершенный код. Мастер-класс / С. Макконнелл. – Москва : Русская редакция, 2019. – 896 с.

28. Таненбаум, Э. Компьютерные сети / Э. Таненбаум, Д. Уэзеролл. – 5-е изд. – Санкт-Петербург : Питер, 2016. – 960 с.

29. Сафронов, В. Системный анализ [Электронный ресурс]. – Режим доступа: https://victor-safronov.ru/. – Дата доступа: 15.10.2025.

30. Learning UML 2.0 – Russ Miles.pdf [Электронный ресурс]. – Режим доступа: https://repository.unikom.ac.id/47347/1/Learning%20UML%202.0%20-%20Russ%20Miles.pdf. – Дата доступа: 15.10.2025.

31. Васильев, А. Н. Java. Объектно-ориентированное программирование : для магистров и бакалавров / А. Н. Васильев. – Санкт-Петербург : Питер, 2014. – 400 с.

32. Боггс, У. UML и Rational Rose / У. Боггс, М. Боггс. – Москва : Лори, 2008. – 580 с.

33. Фримен, Эр. Паттерны проектирования / Эр. Фримен, Эл. Фримен. – Санкт-Петербург : Питер, 2011. – 656 с.

34. Олифер, В. Г. Компьютерные сети : принципы, технологии, протоколы : учебное пособие для студентов вузов / В. Г. Олифер, Н. А. Олифер. – 5-е изд. – Санкт-Петербург : Питер, 2016. – 992 с.

35. Мюллер, Р. Д. Базы данных и UML. Проектирование / Р. Д. Мюллер. – Москва : Лори, 2002. – 420 с.

36. Голицына, О. Л. Базы данных : учебное пособие / О. Л. Голицына, Н. В. Максимов, И. И. Попов. – 4-е изд., перераб. и доп. – Москва : ФОРУМ : ИНФРА-М, 2016. – 400 с.

37. Эккель, Б. Философия Java. Библиотека программиста. / Б. Эккель. – Санкт-Петербург : Питер, 2014. – 640 с.

38. Ларман, К. Применение UML 2.0 и шаблонов проектирования. Введение в объектно-ориентированный анализ и проектирование. – 3-е изд. / К. Ларман. – Москва : Вильямс, 2013. – 736 с.

39. Хорстманн, К. С. Java SE 8. Вводный курс / К. С. Хорстманн. – Москва : Вильямс, 2014. – 208 с.

40. Эванс, Б. Java. Новое поколение разработки / Б. Эванс, М. Вербург. – Санкт-Петербург : Питер, 2014. – 560 с.

41. Уидом, Д. Реляционные базы данных / Д. Уидом. – Москва : Лори, 2014. – 384 с.

42. Курняван, Б. Программирование WEB-приложений на языке Java / Б. Курняван. – Москва : Лори, 2014. – 880 с.

43. Берлин, А. Основные протоколы Интернет / А. Берлин. – Москва : Бином, 2008. – 504 с.

44. Агальцов, В. Базы данных : в 2 кн. / В. Агальцов – Москва : Инфра-М, 2014. – Кн. 2. Распределенные и удаленные базы данных. – 272 с.

45. Мюллер, Р. Проектирование баз данных и UML / Р. Мюллер. – Москва : Лори, 2013. – 420 с.

46. Приемы объектно-ориентированного проектирования. Паттерны проектирования /Р. Хелм, Э. Гамма. – Санкт-Петербург : Питер, 2013. – 368 с.

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ОРГАНИЗАЦИИ И ВЫПОЛНЕНИЮ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ

При изучении учебной дисциплины рекомендуется использовать следующие формы самостоятельной работы:

написание рефератов;

участие студентов в научно-исследовательской и методической работе, проводимой на кафедре;

участие в конкурсах студенческих работ и студенческих конференциях.

ПЕРЕЧЕНЬ РЕКОМЕНДУЕМЫХ СРЕДСТВ ДИАГНОСТИКИ КОМПЕТЕНЦИЙ ОБУЧАЮЩИХСЯ

Примерным учебным планом по специальности 6-05-0611-01 «Информационные системы и технологии» в качестве формы промежуточной аттестации по учебной дисциплине «Инструменты поддержки промышленной разработки программного обеспечения» рекомендуется зачет.

Для текущего контроля по учебной дисциплине и диагностики компетенций могут использоваться следующие формы:

устный опрос;

защита лабораторных работ;

собеседование;

защита индивидуальной практической работы.

РЕКОМЕНДУЕМЫЕ МЕТОДЫ (ТЕХНОЛОГИИ) ОБУЧЕНИЯ

Основные рекомендуемые методы (технологии) обучения, отвечающие цели и задачам учебной дисциплины:

обучение с помощью аудиовизуальных технических средств;

компьютерное обучение;

элементы проблемного обучения (проблемное изложение, вариативное изложение, частично-поисковый метод), реализуемые на лекционных занятиях;

элементы учебно-исследовательской деятельности, творческого подхода, реализуемые на лабораторных занятиях.

ПРИМЕРНЫЙ ПЕРЕЧЕНЬ ТЕМ ЛАБОРАТОРНЫХ ЗАНЯТИЙ

1. Проектирование и реализация программных систем с применением сетевых структур классов и принципов слабой связности.
2. Инструменты и средства разработки сценариев в системах непрерывной интеграции
3. Построение базового CI/CD конвейера в Jenkins с использованием Groovy DSL
4. Автоматизация CI/CD процессов на основе Jenkins и систем контроля версий исходного кода проектов.
5. Статический анализ и улучшение качества Java-кода: исключения и логгирование.
6. Мониторинг и наблюдаемость CI/CD процессов с Prometheus и Grafana.

ПРИМЕРНЫЙ ПЕРЕЧЕНЬ КОМПЬЮТЕРНЫХ ПРОГРАММ

1. Операционная система Microsoft Windows 10 или выше.
2. Enterprise Architect 7.1 (и выше) | Plantuml (последние версии).
3. AllFusion Process Modeler 7 (и выше).

4. Java SE 11 (LTS) (и выше).
5. IDE: IntelliJ IDEA|Eclipse|NetBeans|SpringToolSuite4 (одну из IDE, последние версии).
6. MAVEN 3.9.1 (и выше).
7. ANT 1.10.x.
8. Jenkins , версия 2.516.x.
9. Apache Tomcat 7.x (и выше).
10. Веб-браузер (последние версии).
11. Junit, версия 5 (и выше), с совместимой библиотекой «Mockito».
12. GIT, версии 2.0.
13. Сервер (CI/CD) Jenkins (последние версии).
14. SonarQube 25.11 (и выше).
15. PMD 7.19.0 (и выше)
16. Checkstyle 12.1.2 (и выше).
17. SpotBugs 4.9.8 (и выше).
18. Logback 1.5.21 (и выше).
19. Prometheus версия 3.8.0 (и выше).
20. Grafana 12.3 (и выше).
21. Loki 3.6.2 (и выше).
22. Docker Engine 27.5.1. (и выше).
23. Kubernetes 1.34.2. (и выше).