

МИНОБРНАУКИ РОССИИ
ВЛАДИВОСТОКСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

**РАБОЧАЯ ПРОГРАММА
УЧЕБНОЙ ДИСЦИПЛИНЫ**

МДК.03.01 Моделирование и анализ программного обеспечения

программы подготовки специалистов среднего звена /
квалифицированных рабочих и служащих

09.02.07 Информационные системы и программирование

Форма обучения: *очная*

Находка 2026

Рабочая программа учебной дисциплины *МДК.03.01 Моделирование и анализ программного обеспечения* разработана в соответствии с требованиями Федерального государственного образовательного стандарта среднего профессионального образования по специальности / профессии *09.02.07 Информационные системы и программирование*, утвержденного приказом Минобрнауки России от «09» декабря 2016 г. № 1547, примерной образовательной программой.

Разработчик(и): *Лихацкий М.Б. преподаватель ОСПО филиала ФГБОУ ВО «ВВГУ» в г. Находке*

Рассмотрено и одобрено на заседании цикловой методической комиссии

Протокол № 10 от «29» июня 2026 г.

Председатель ЦМК _____ *В.В Плахотина*
подпись

Содержание

- 1 ОБЩАЯ ХАРАКТЕРИСТИКА РАБОЧЕЙ ПРОГРАММЫ
УЧЕБНОЙ ДИСЦИПЛИНЫ**
- 2 СТРУКТУРА И СОДЕРЖАНИЕ УЧЕБНОЙ ДИСЦИПЛИНЫ**
- 3 УСЛОВИЯ РЕАЛИЗАЦИИ УЧЕБНОЙ ДИСЦИПЛИНЫ**
- 4 КОНТРОЛЬ И ОЦЕНКА РЕЗУЛЬТАТОВ ОСВОЕНИЯ
УЧЕБНОЙ ДИСЦИПЛИНЫ**

1. ОБЩАЯ ХАРАКТЕРИСТИКА РАБОЧЕЙ ПРОГРАММЫ УЧЕБНОЙ ДИСЦИПЛИНЫ

1.1 Место дисциплины в структуре основной образовательной программы

Учебная дисциплина *МДК.03.01 Моделирование и анализ программного обеспечения* является частью учебного цикла основной образовательной программы (далее ООП) в соответствии с ФГОС СПО по специальности/профессии *09.02.07 Информационные системы и программирование*.

1.2 Цель и планируемые результаты освоения дисциплины

По итогам освоения дисциплины, обучающиеся должны продемонстрировать результаты обучения, соотнесённые с результатами освоения ООП СПО, приведенные в таблице.

Код компетенции	Умения	Знания
ОК 1 ОК 2	Применять UML для построения моделей ПО (диаграммы классов, последовательности, состояний).	Методологии моделирования (структурный, объектно-ориентированный, компонентный).
ОК 4 ОК 5	Использовать CASE-средства для создания и документирования моделей. Проводить анализ требований и формализацию в Use Case.	Нотации UML, IDEF0, DFD, ERD, BPMN. Принципы объектно-ориентированного проектирования (SOLID, GRASP, паттерны).
ОК 9 ОК 10	Оценивать качество моделей по метрикам (сложность, связность, сцепление). Выполнять статический и динамический анализ кода.	Метрики качества ПО и методы тестирования моделей. Стандарты документирования проектных решений.
ПК 3.1 ПК 3.2 ПК 3.3 ПК 3.4 ПК 3.5 ПК 3.6 ПК 5.1 ПК 5.2 ПК 5.3 ПК 5.4 ПК 5.5 ПК 5.6	Собирать данные для проектирования модулей. Разрабатывать модели компонентов и взаимодействий. Проводить анализ производительности и оптимизацию моделей. Оценивать риски и сложность реализации. Документировать модели в составе проектной документации.	Состав проектной документации. Технологии разработки модулей на основе моделей. Методы оценки трудоёмкости и планирования. Принципы построения архитектуры ИС. Требования к верификации и валидации моделей.

2 Структура и содержание учебной дисциплины

2.1 Объем учебной дисциплины и виды учебной работы

Таблица 2 - Распределение трудоемкости дисциплины по видам учебной работы

Вид учебной работы	Объем часов
Объем образовательной программы учебной дисциплины	35
в том числе:	
– теоретическое обучение	16
– практические занятия (<i>если предусмотрено</i>)	18
– самостоятельная работа	
– консультации	
– промежуточная аттестация – (<i>дифференцируемый зачет</i>)	1

2.2 Тематический план и содержание учебной дисциплины «МДК.03.01 Моделирование и анализ программного обеспечения»

Таблица 3 - Тематический план дисциплины

Наименование разделов и тем	Содержание учебного материала и формы организации деятельности обучающихся	Объем часов	Коды компетенций, формированию которых способствует элемент программы
1	2	3	4
Раздел 1. Моделирование и анализ программного обеспечения		35	
Тема 1.1 Моделирование и анализ в разработке программного обеспечения	Содержание учебного материала: Жизненный цикл программного обеспечения. Инструменты анализа программного обеспечения. Цели и роль моделирования в разработке программного обеспечения. Понятие ревьюирования программного обеспечения. Визуальное моделирование. Деление методов моделирования на основе подхода к декомпозиции системы: «снизу вверх» и «сверху вниз». Структурное моделирование. Семейство стандартов IDEF. Объектное моделирование. Автоматизированная поддержка стадий разработки ПО. Case-средства.	5	ОК 1, ОК 2, ОК 4, ОК 5, ОК 9, ОК 10, ПК 3.1, ПК 3.2, ПК 3.3, ПК 3.4, ПК 3.5, ПК 3.6, ПК 5.1, ПК 5.2, ПК 5.3, ПК 5.4, ПК 5.5, ПК 5.6
	Практическая работа №1: Стандарт моделирования IDEF0. Практическая работа №2: Функциональное моделирование в Ramus	5	
	Самостоятельная работа обучающихся	Не предусмотрено	
	Лабораторное занятие	Не предусмотрено	

Наименование разделов и тем	Содержание учебного материала и формы организации деятельности обучающихся	Объем часов	Коды компетенций, формированию которых способствует элемент программы
1	2	3	4
Тема 1.2. Моделирование структуры ПО с помощью структурных диаграмм UML	Содержание учебного материала: UML – графический язык моделирования ПО. Основные элементы нотации UML. Виды диаграмм и их использование на всех этапах жизненного цикла ПО. Последовательность проектирования приложения. Анализ проблем с помощью моделирование использования. Семантика и нотация моделей использования. Описание функций системы. Моделирование структуры. Диаграмма классов. Диаграмма компонентов. Диаграмма развертывания.	5	ОК 1, ОК 2, ОК 4, ОК 5, ОК 9, ОК 10, ПК 3.1, ПК 3.2, ПК 3.3, ПК 3.4, ПК 3.5, ПК 3.6, ПК 5.1, ПК 5.2, ПК 5.3, ПК 5.4, ПК 5.5, ПК 5.6
	Практическая работа №3: Построение диаграмм вариантов использования/ Диаграммы классов. Диаграмма компонентов. Диаграмма развертывания.	5	
	Самостоятельная работа обучающихся	Не предусмотрено	
	Лабораторное занятие	Не предусмотрено	
Тема 1.3 Моделирование поведения ПО с помощью диаграмм деятельности UML	Содержание учебного материала: Диаграммы поведения в UML . Диаграммы прецедентов для описания организации поведения системы. Диаграммы последовательностей для описания временной упорядоченности сообщений. Диаграммы кооперации для описания структурной организации объектов, посылающих и получающих сообщения. Диаграммы состояний для описания изменения состояния системы в ответ на события. Диаграммы деятельности для	6	ОК 1, ОК 2, ОК 4, ОК 5, ОК 9, ОК 10, ПК 3.1, ПК 3.2, ПК 3.3, ПК 3.4, ПК 3.5, ПК 3.6, ПК 5.1, ПК 5.2, ПК 5.3, ПК 5.4, ПК 5.5, ПК 5.6
	Практическая работа №4: Построение диаграмм последовательности.	8	
	Самостоятельная работа обучающихся	Не предусмотрено	
	Лабораторное занятие	Не предусмотрено	
Промежуточная аттестация - дифференцированный зачет		1	

ИТОГО часов		35	
Теоретические занятия		16	
Практические занятия		18	
Самостоятельная работа		0	
Дифференцированный зачет		1	

3 Условия реализации учебной дисциплины

3.1 Для реализации программы учебной дисциплины «МДК.03.01 Моделирование и анализ программного обеспечения» образовательной организации, предусмотрено наличие следующих специальных помещений:

Для реализации программы учебной дисциплины предусмотрено наличие следующих специальных помещений:

Компьютерный класс информационных технологий: количество посадочных мест - 28 шт., доска меловая - 1 шт., стол для преподавателя - 1 шт., стул для преподавателя - 1 шт.; учебно-методические пособия, дидактические и презентационные материалы

Компьютерный класс имеет следующее оборудование:

Учебно-лабораторное оборудование: Процессор Intel Количество ядер - 2 . Объем кэш-памяти - 8Мб • Базовая частота процессора - 2,0 ГГц. Объем установленной памяти - 4Гб Накопитель твердотельный объемом 512 видео карта - встроенная. Максимальное разрешение - 1920x1080 пикс. Диагональ монитора 23" количество мониторов 1 шт. Разрешение: FHD (1920x1080 пикс.) клавиатура мышь ОС Microsoft Windows 7 Pro Мультимедийный комплект в составе: проектор Sanyo, крепление Kromax, экран Lumien Eco Picture: Технология проецирования изображения - микрозеркала на полупроводниковом чипе. Базовое разрешение проектора: 1280x800 пикс. Входы - VGA, HDMI 2 шт., Порты - RS-232C, USB 2 шт., LAN Размер полотна: ширина 200 см, высота 127 см. ПО: Microsoft Office 2010, Пакет Adobe CC 2019, Pixologic Zbrush Academic Volume License, Adobe Substance Painter, Autodesk 3Ds max, Maya, Blender, Unity, Unreal Engine

3.2 Информационное обеспечение реализации программы

Для реализации программы учебной дисциплины библиотечный фонд ВВГУ укомплектован печатными и электронными изданиями.

Обучающиеся из числа инвалидов и лиц с ограниченными возможностями здоровья обеспечены печатными и (или) электронными образовательными ресурсами в формах, адаптированных к ограничениям их здоровья.

Основная литература:

1. Бизяев, А.А. Информационные технологии: практикум: [Электронный ресурс]/А.А. Бизяев, К.А. Куратов ; Новосибирский государственный технический университет. - Новосибирск: Новосибирский государственный технический университет, 2022. - 96 с.: ил., табл. - Режим доступа: <http://biblioclub.ru/index.php?page=book&id=575330>

2. Информационные технологии [Электронный ресурс] : метод. рекомендации по самостоят. изучению дисциплины / Г.И. Попов, С.И. Бажинов, В.Г. Конюхов, Е.Н. Яшкина, В.А. Пирогов .- М. : РГУФКСМиТ, 2022 .-167 с. : ил. - Режим доступа: <https://lib.rucont.ru/efd/671375>

3. Информационные технологии в профессиональной деятельности [Электронный ресурс] / Т.В. Марзаева .— Улан-Удэ : Бурятский государственный университет, 2025 .- 71 с. -Режим доступа: <https://lib.rucont.ru/efd/348195>

4. Информационные технологии: учеб. пособие [Электронный ресурс]/ Л.Г. Гагарина, Я.О. Теплова, Е.Л. Румянцева, А.М. Баин / под ред. Л.Г. Гагариной. — Москва: ИД «ФОРУМ»: ИНФРА- М, 2021. — 320 с. -Режим доступа: <https://znanium.com/catalog/product/471464>.

5. Филимонова, Е.В. Информационные технологии в профессиональной деятельности : учебник / Филимонова Е.В. — Москва : КноРус, 2022. — 482 с. — (СПО). — ISBN 978-5-406-04887-0. — URL: <https://book.ru/book/922139> (дата обращения:

06.08.2024). — Текст : электронный.

6. Хныкина, А.Г. Информационные технологии [Электронный ресурс] : учеб. пособие / Т.В. Минкина, А.Г. Хныкина .— Ставрополь : изд-во СКФУ, 2023 .— 126 с. : ил. — Режим доступа: <https://lib.rucont.ru/efd/671178>

Дополнительная литература:

1. Богданова, С.В. Информационные технологии [Электронный ресурс]: учебное пособие / С.В. Богданова, А.Н. Ермакова. - Ставрополь: Сервисшкола, 2024. - 211 с. - Режим доступа: <https://znanium.com/catalog/product/514867>

2. Самойленко, А.П. Информационные технологии статистической обработки данных [Электронный ресурс] : учеб. пособие / О.А. Усенко, Южный федеральный ун-т, А.П. Самойленко .— Ростов н/Д. : Изд-во ЮФУ, 2024 .— 128 с. : ил. — ISBN 978-5-9275-2521-8 .— Режим доступа: <https://lib.rucont.ru/efd/692393>

3. Суханова, О.Н. Информационные технологии [Электронный ресурс] : лаб. практикум / О.В.

4. Ментюкова, О.Н. Суханова .— Пенза : РИО ПГСХА, 2022 .- 116 с. : ил. - Режим доступа: <https://lib.rucont.ru/efd/323727>.

Интернет-ресурсы:

1. www.informatics.mccme.ru/moodle/ - дистанционная подготовка по информатике.
2. www.ixbt.com - сайт о высоких технологиях, новости индустрии из мира компьютерного «железа», тестовые испытания и обзоры оборудования. Конференция. Платформа ПК. Цифровой звук. Цифровое фото. Мобильные устройства. Форум.
3. www.lib.ru/INTERMET/termwww.txt - толковый словарь терминов.
4. www.computerra.ru - новости про компьютеры, железо, новые технологии, информационные технологии. Рубрики: В поле зрения; Компьютерралаб; Дюжина полезных программ; Реклама на сайте; Блоги; Наука и жизнь. Архив номеров журнала.
5. www.novtex.ru/IT/arhiv.htm
6. <https://www.planttext.com/>
7. <https://kroki.io/#>
8. <https://www.canva.com/ru ru/>
9. <https://prezi.com/>
10. www.jir.ru/ - электронный научный журнал.

4. Контроль результатов освоения учебной дисциплины

Результаты обучения	Критерии оценки	Методы оценки
- актуальный профессиональный и социальный контекст, в котором приходится работать и жить; - основные источники информации и ресурсы для решения задач и проблем в профессиональном и/или социальном контексте; - алгоритмы выполнения работ в профессиональной и смежных областях; - методы работы в профессиональной и смежных сферах; - структуру плана для решения задач; - порядок оценки результатов решения задач профессиональной деятельности - основные методы и средства сбора, обработки, хранения, передачи и накопления информации; - технологию поиска информации в сети Интернет; - номенклатура информационных источников применяемых в профессиональной деятельности; приемы структурирования информации; - формат оформления результатов поиска информации - содержание актуальной нормативно-правовой документации;	Уровень правильных ответов при письменном и устном контроле, четкость изложения материала. Быстрота ориентации в представляемом материале, быстрота реакции на вопросы. Правильность, полнота выполнения заданий, точность формулировок, точность расчетов, соответствие требованиям.	устный опрос, практические работы

- современная научная и профессиональная терминология; - возможные траектории профессионального развития и самообразования - психологические основы деятельности коллектива, - психологические особенности личности; - основы проектной деятельности - особенности социального и культурного контекста; - правила оформления документов и построения устных сообщений - назначение, состав, основные характеристики организационной и компьютерной техники;		
- основные компоненты информационных систем, принципы внедрения информационных систем, организацию межсетевого взаимодействия; - назначение и принципы использования системного и прикладного программного обеспечения; - принципы защиты информации от несанкционированного доступа; - правовые аспекты использования информационных технологий и программного обеспечения; - основные понятия	Оптимальность выбора способов действий, методов, последовательностей действий. Соответствие требованиям инструкций, регламентов, рациональность действий. Точность распознавания проблемных ситуаций в различных контекстах; Адекватность анализа сложных ситуаций при решении задач профессиональной деятельности, эффективность поиска. Точность определения источников нужных ресурсов. Разработка детального плана действий, правильность оценки рисков на каждом шагу; Точность оценки плюсов и минусов полученного результата, своего плана и его реализации, предложение критериев оценки и рекомендаций по улучшению плана	Текущий контроль: экспертное наблюдение и оценка в процессе выполнения заданий для практических занятий;

автоматизированной обработки информации;		
---	--	--

Для оценки достижения запланированных результатов обучения по дисциплине разработаны контрольно-оценочные средства для проведения текущего контроля и промежуточной аттестации, которые прилагаются к рабочей программе дисциплины.

Для оценки достижения запланированных результатов обучения по дисциплине разработаны контрольно-оценочные средства для проведения текущего контроля и промежуточной аттестации, которые прилагаются к рабочей программе дисциплины.

Оценка результатов обучения производится при помощи бально-рейтинговой системы.

Основные критерии оценки:

- конспектирование 0,5 баллов;
- активность на уроке 0,5 баллов;
- посещаемость занятий 0,5 баллов;
- выполнение индивидуальных домашних заданий 1 задание- 1 балл;
- выполнение самостоятельной работы 1 балл;
- лабораторная работа 3 балла;
- контрольная работа 1 задание -1 балл;
- участие в научно-исследовательской работе 15 баллов;
- дифференцированный зачет 20 баллов.

МИНОБРНАУКИ РОССИИ
ВЛАДИВОСТОКСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

для проведения текущего контроля и промежуточной аттестации
по учебной дисциплине

МДК.03.01 Моделирование и анализ программного обеспечения
программы подготовки специалистов среднего звена /
квалифицированных рабочих и служащих

09.02.07 Информационные системы и программирование

Форма обучения: очная

Находка 2026

Оценочные средства для проведения текущего контроля и промежуточной аттестации по учебной дисциплине *МДК.03.01 Моделирование и анализ программного обеспечения* разработаны в соответствии с требованиями ФГОС СПО по специальности / профессии *09.02.07 Информационные системы и программирование*, утвержденного приказом Минобрнауки России от «09» декабря 2016 г. № 1547, примерной образовательной программой.

Разработчик(и): *Лихацкий М.Б. преподаватель ОСПО филиала ФГБОУ ВО «ВВГУ» в г. Находке*

Рассмотрено и одобрено на заседании цикловой методической комиссии

Протокол № 10 от «29» июня 2026 г.

Председатель ЦМК _____ *В.В Плахотина*
подпись

1 Общие сведения

Фонд оценочных средств (далее – ФОС) предназначены для контроля и оценки образовательных достижений обучающихся, освоивших программу учебной дисциплины *МДК.03.01 «Моделирование и анализ программного обеспечения»*. ФОС включают в себя контрольные материалы для проведения текущего контроля успеваемости и промежуточной аттестации по дисциплине, которая проводится в форме дифференцированного зачёта (с использованием оценочного средства - *тестирование*)

2 Планируемые результаты обучения по дисциплине, обеспечивающие результаты освоения образовательной программы

Код ОК, ПК	Код результата обучения	Наименование результата обучения ¹
ОК 1	31	методологии моделирования ПО (UML, IDEF0, DFD, ERD)
ОК 2	32	принципы объектно-ориентированного проектирования (SOLID, GRASP, паттерны)
ОК 4		
ОК 5	33	инструментальные средства моделирования и анализа (CASE-средства)
ОК 9		
ОК 10	34	метрики качества ПО и методы тестирования моделей
ПК 5.1	35	состав проектной документации и методы оценки трудоёмкости
ПК 5.2		
ПК 5.6	У1	применять UML для построения диаграмм (классов, последовательности, состояний)
ПК 6.3		
	У2	использовать CASE-средства для документирования моделей
	У3	проводить анализ требований и оценку качества моделей
	У4	разрабатывать модели компонентов и взаимодействий, документировать результаты

3 Соответствие оценочных средств контролируемым результатам обучения

Краткое наименование раздела (модуля) / темы дисциплины	Код результата обучения	Показатель ² овладения результатами обучения	Наименование оценочного средства и представление его в КОС	
			Текущий контроль	Промежуточная аттестация
Тема 1.1 Моделирование и анализ в разработке программного обеспечения	31	Способность применять методологии моделирования (UML, IDEF0, DFD) для описания ПО	<i>Устный опрос (п 5.1 вопросы 1-5) Реферат (п 5.2 темы 1-10) Практическая работа №1-2 (п 5.3)</i>	<i>Устный опрос (п 6.1 вопросы 1-18) Тест вариант №1-2 (п 6.2)</i>
	33	Способность использовать CASE-средства для создания и документирования моделей		
	У1	Способность строить диаграммы UML (вариантов использования, классов)		
Тема 1.2. Моделирование структуры ПО с помощью структурных диаграмм UML	32	Способность описать принципы объектно-ориентированного проектирования (SOLID, GRASP)	<i>Устный опрос (п 5.1 вопросы 6-10) Реферат (п 5.2 темы 11-29) Практическая работа №3 (п 5.3)</i>	<i>Устный опрос (п 6.1 вопросы 1-18) Тест вариант №1-2 (п 6.2)</i>
	У2	Способность разрабатывать модели классов и компонентов в CASE-средствах		
	У3	Способность создавать диаграммы деятельности и последовательности для описания бизнес-процессов		
Тема 1.3 Моделирование поведения ПО с помощью диаграмм деятельности и UML	34	Способность применять метрики качества для оценки моделей	<i>Устный опрос (п 5.1 вопросы 11-18) Реферат (п 5.2 темы 20-30) Практическая работа №4 (п 5.3)</i>	<i>Устный опрос (п 6.1 вопросы 1-18) Тест вариант №1-2 (п 6.2)</i>
	У3	Способность создавать диаграммы деятельности и последовательности для описания бизнес-процессов		

4. Описание процедуры оценивания

Результаты обучения по дисциплине, уровень сформированности компетенций оцениваются по четырёх бальной шкале оценками: «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

Текущая аттестация по дисциплине проводится с целью систематической проверки достижений обучающихся. Объектами оценивания являются: степень усвоения теоретических знаний, уровень овладения практическими умениями и навыками по всем видам учебной работы, качество выполнения самостоятельной работы, учебная дисциплина (активность на занятиях, своевременность выполнения различных видов заданий, посещаемость всех видов занятий по аттестуемой дисциплине).

При проведении промежуточной аттестации оценивается достижение студентом запланированных по дисциплине результатов обучения, обеспечивающих результаты освоения образовательной программы в целом. *Оценка на зачете выставляется с учетом оценок, полученных при прохождении текущей аттестации.*

Критерии оценивания устного ответа

(оценочные средства: *собеседование, устное сообщение*)

5 баллов - ответ показывает прочные знания основных процессов изучаемой предметной области, отличается глубиной и полнотой раскрытия темы; владение терминологическим аппаратом; умение объяснять сущность, явлений, процессов, событий, делать выводы и обобщения, давать аргументированные ответы, приводить примеры; свободное владение монологической речью, логичность и последовательность ответа; умение приводить примеры современных проблем изучаемой области.

4 балла - ответ, обнаруживающий прочные знания основных процессов изучаемой предметной области, отличается глубиной и полнотой раскрытия темы; владение терминологическим аппаратом; умение объяснять сущность, явлений, процессов, событий, делать выводы и обобщения, давать аргументированные ответы, приводить примеры; свободное владение монологической речью, логичность и последовательность ответа. Однако допускается одна - две неточности в ответе.

3 балла – ответ, свидетельствующий в основном о знании процессов изучаемой предметной области, отличающийся недостаточной глубиной и полнотой раскрытия темы; знанием основных вопросов теории; слабо сформированными навыками анализа явлений, процессов, недостаточным умением давать аргументированные ответы и приводить примеры; недостаточно свободным владением монологической речью, логичностью и последовательностью ответа. Допускается несколько ошибок в содержании ответа; неумение привести пример развития ситуации, провести связь с другими аспектами изучаемой области.

2 балла – ответ, обнаруживающий незнание процессов изучаемой предметной области, отличающийся неглубоким раскрытием темы; незнанием основных вопросов теории, несформированными навыками анализа явлений, процессов; неумением давать аргументированные ответы, слабым владением монологической речью, отсутствием логичности и последовательности. Допускаются серьезные ошибки в содержании ответа; незнание современной проблематики изучаемой области.

Критерии оценивания письменной работы

(оценочные средства: *реферат, конспект, контрольная работа, расчетно-графическая работа, доклад (сообщение), в том числе выполненный в форме презентации, творческое задание*).

5 баллов - студент выразил своё мнение по сформулированной проблеме, аргументировал его, точно определив ее содержание и составляющие. Проблема раскрыта полностью, выводы обоснованы. Приведены данные отечественной и зарубежной литературы, статистические сведения, информация нормативно-правового характера. Студент владеет навыком самостоятельной работы по заданной теме; методами и приемами анализа теоретических и/или практических аспектов изучаемой области. Фактических ошибок, связанных с пониманием проблемы, нет; графическая работа оформлена правильно.

4 балла - работа характеризуется смысловой цельностью, связностью и последовательностью изложения; допущено не более 1 ошибки при объяснении смысла или содержания проблемы. Проблема раскрыта. Не все выводы сделаны и/или обоснованы. Для аргументации приводятся данные отечественных и зарубежных авторов. Продемонстрированы исследовательские умения и навыки. Фактических ошибок, связанных с пониманием проблемы, нет. Допущены одна-две ошибки в оформлении работы.

3 балла – студент проводит достаточно самостоятельный анализ основных этапов и смысловых составляющих проблемы; понимает базовые основы и теоретическое обоснование выбранной темы. Проблема раскрыта не полностью. Выводы не сделаны и/или выводы не обоснованы. Проведен анализ проблемы без привлечения дополнительной литературы. Допущено не более 2 ошибок в смысле или содержании проблемы, оформлении работы.

2 балла - работа представляет собой пересказанный или полностью переписанный исходный текст без каких бы то ни было комментариев, анализа. Не раскрыта структура и теоретическая составляющая темы. Проблема не раскрыта. Выводы отсутствуют. Допущено три или более трех ошибок в смысловом содержании раскрываемой проблемы, в оформлении работы.

Критерии оценивания тестового задания

Оценка	Отлично	Хорошо	Удовлетворительно	Неудовлетворительно
Количество правильных ответов	91 % и $\geq$	от 81% до 90,9 %	не менее 70%	менее 70%

Критерии выставления оценки студенту на зачете

(оценочные средства: устный опрос в форме ответов на вопросы билетов, устный опрос в форме собеседования, выполнение письменных разноуровневых задач и заданий, комплексная расчетно-графическая работа, творческое задание, кейс-задача, портфолио, проект и т.п.)

Оценка по промежуточной аттестации	Характеристика качества сформированности компетенций
«зачтено» / «отлично»	Студент демонстрирует сформированность дисциплинарных компетенций на продвинутом уровне: обнаруживает всестороннее, систематическое и глубокое знание учебного материала, умеет тесно увязывать теорию с практикой, свободно справляется с задачами, вопросами и другими видами применения знаний, причем не затрудняется с ответом при видоизменении заданий, правильно обосновывает принятое решение, владеет разносторонними навыками и приемами выполнения практических задач.
«зачтено» / «хорошо»	Студент демонстрирует сформированность дисциплинарных компетенций на базовом уровне: основные знания, умения освоены, но допускаются незначительные ошибки, неточности, затруднения при аналитических операциях, переносе знаний и умений на новые, нестандартные ситуации.
«зачтено» / «удовлетворительно»	Студент демонстрирует сформированность дисциплинарных компетенций на пороговом уровне: имеет знания только основного материала, но не усвоил его деталей, в ходе контрольных мероприятий допускаются значительные ошибки, недостаточно правильные формулировки, нарушения логической последовательности в изложении программного

	материала, испытывает затруднения при выполнении практических работ, при оперировании знаниями и умениями при их переносе на новые ситуации.
«не зачтено» / «неудовлетворительно»	Студент демонстрирует сформированность дисциплинарных компетенций на уровне ниже порогового: выявляется полное или практически полное отсутствие знаний значительной части программного материала, студент допускает существенные ошибки, неуверенно, с большими затруднениями выполняет практические работы, умения и навыки не сформированы.

5. Примеры оценочных средств для проведения текущей аттестации

5.1 Вопросы для устного опроса

1. Что такое модель в контексте разработки программного обеспечения и каковы цели моделирования?
2. Какие преимущества дает использование моделей на этапе проектирования ПО?
3. Что такое UML и для каких целей он используется в разработке ПО?
4. Назовите основные группы диаграмм UML (структурные и поведенческие) и приведите примеры диаграмм из каждой группы.
5. Опишите основные принципы объектно-ориентированного подхода, которые находят отражение в UML-моделировании.
6. Что показывает диаграмма классов (Class Diagram) и какие основные элементы на ней изображаются?
7. Как на диаграмме классов обозначаются модификаторы доступа (public, private, protected)?
8. Что такое ассоциация между классами? Какие виды ассоциаций вы знаете (агрегация, композиция)?
9. Чем агрегация отличается от композиции? Приведите примеры.
10. Что такое диаграмма компонентов (Component Diagram) и для чего она используется?
11. Что такое диаграмма развертывания (Deployment Diagram) и какую информацию она передает?
12. Что такое диаграмма пакетов (Package Diagram) и как она помогает организовать структуру проекта?
13. Что такое диаграмма деятельности (Activity Diagram) и для чего она применяется?
14. Какие основные элементы используются на диаграмме деятельности (действия, решения, ветвления, слияния)?
15. Как обозначается начало (initial node) и конец (final node) потока на диаграмме деятельности?
16. Что такое "решение" (decision node) и "слияние" (merge node) на диаграмме деятельности?
17. Как изображается параллельное выполнение действий (fork и join) на диаграмме деятельности?
18. Что такое "дорожки" (swimlanes) на диаграмме деятельности и для чего они используются?

5.2 Темы рефератов

Тема 1.1. Моделирование и анализ в разработке программного обеспечения

1. Жизненный цикл программного обеспечения: этапы, модели и их применение на практике.
2. Понятие моделирования в разработке ПО: цели, задачи и значение процесса.
3. Классификация моделей в программной инженерии: структурные, объектные, функциональные.
4. Сравнительный анализ методов и средств моделирования программных систем.
5. Структурное моделирование: основные принципы, схемы данных и архитектуры.
6. Объектное моделирование: современные подходы и инструменты.
7. Методология функционального моделирования IDEF0: нотация, принципы и практическое применение.
8. Обзор и сравнительный анализ CASE-средств для моделирования (Ramus, BPwin, ERwin).
9. Стандарты, регламентирующие процесс моделирования в разработке ПО (ГОСТ, ISO).
10. Роль глоссария и словаря данных при построении моделей программного обеспечения.

Тема 1.2. Моделирование структур данных и процессов структурными методами

11. Унифицированный язык моделирования (UML): история создания, назначение и области применения.
12. Основные понятия UML: системы, модели, объекты и субъекты, их характеристики.
13. Объектная модель и её основные свойства: инкапсуляция, наследование, полиморфизм.
14. Классы и объекты в UML: принципы построения и визуализация в диаграммах.
15. Принципы моделирования структур данных: от концептуальной до физической модели.
16. Модель «сущность-связь» (ER-модель): применение при проектировании баз данных.
17. Методы декомпозиции систем при структурном моделировании.
18. Диаграммы потоков данных (DFD): правила построения и назначение.
19. Сравнительный анализ структурного и объектного подходов к моделированию.

Тема 1.3. Моделирование ПО с помощью диаграмм UML

20. Диаграмма вариантов использования (Use Case): построение, элементы и практические примеры.
21. Диаграммы последовательности: отражение динамики взаимодействия объектов во времени.
22. Диаграмма деятельности (Activity): моделирование бизнес-процессов и алгоритмов.

23. Диаграмма состояний (State Machine): жизненный цикл объекта и переходы.
24. Диаграмма классов (Class Diagram): статическая структура программной системы.
25. Диаграмма компонентов и развертывания: физическая архитектура системы.
26. Применение диаграмм UML для документирования требований и проектирования.
27. Практические аспекты построения диаграмм UML в современных CASE-средствах (StarUML, Visual Paradigm).
28. Связь UML-моделей с кодом: генерация кода и обратное проектирование.
29. Комплексное использование диаграмм UML для описания архитектуры информационной системы.
30. Ошибки и типичные проблемы при построении UML-моделей и способы их предотвращения.

5.3 Примеры практических работ

ПРАКТИЧЕСКАЯ РАБОТА №1

Тема: Стандарт моделирования IDEF0

Цель:

Изучить методологию функционального моделирования IDEF0 и освоить построение контекстных диаграмм.

Задание:

Разработать функциональную модель IDEF0 для заданного бизнес-процесса (например, «Обработка заказа в интернет-магазине», «Управление складом», «Процесс согласования документов»). Модель должна содержать:

- Контекстную диаграмму (уровень A-0).
- Диаграмму декомпозиции первого уровня (не менее 3-х функциональных блоков).
- Описание входов, выходов, механизмов и управлений для каждого блока.

Порядок выполнения:

1. Изучить методологию IDEF0 (нотация, основные элементы, правила построения).
2. Выбрать бизнес-процесс (по заданию преподавателя).
3. Определить границы модели (что входит в систему, что остаётся за её пределами).
4. Построить контекстную диаграмму A-0:
 - Один функциональный блок.
 - Входные данные (стрелки слева).
 - Выходные данные (стрелки справа).
 - Управления (стрелки сверху).
 - Механизмы (стрелки снизу).
5. Выполнить декомпозицию (разбить процесс на 3–5 подпроцессов).
6. Оформить модель в одном из инструментов:
 - Ramus Educational (бесплатный).
 - BPwin.
 - [Draw.io](https://draw.io) (с нотацией IDEF0).
7. Подготовить отчёт.

Содержание отчёта:

- Описание бизнес-процесса и его границ.
- Контекстная диаграмма A-0 (скриншот).
- Диаграмма декомпозиции первого уровня.

- Таблица с описанием стрелок (входы, выходы, управления, механизмы).
- Выводы.

Контрольные вопросы:

1. Что такое IDEF0 и для каких задач она применяется?
2. Что такое контекстная диаграмма и для чего она нужна?
3. Как обозначаются входы, выходы, управления и механизмы в IDEF0?
4. В чём смысл декомпозиции функциональных блоков?

ПРАКТИЧЕСКАЯ РАБОТА №2

Тема: Функциональное моделирование в Ramus

Цель:

Освоить работу с инструментом Ramus Educational для создания функциональных моделей и научиться связывать диаграммы в иерархию.

Задание:

В среде Ramus Educational создать функциональную модель IDEF0 для процесса «Регистрация и обработка заявок в службе поддержки». Модель должна включать:

- Контекстную диаграмму A-0.
- Декомпозицию на 3–4 блока (уровень A0).
- Детализацию минимум одного блока на следующем уровне (A1, A2 и т.д.).
- Связи между диаграммами с помощью ICOM-кодов.

Порядок выполнения:

1. Установить и запустить Ramus Educational.
2. Создать новый проект.
3. Построить контекстную диаграмму:
 - Добавить функциональный блок.
 - Добавить стрелки (входы, выходы, управления, механизмы).
 - Задать названия и типы стрелок.
4. Выполнить декомпозицию контекстного блока:
 - Нажать «Перейти на нижний уровень».
 - Разбить процесс на 3–4 блока.
 - Провести соединения между блоками.
5. Выполнить детализацию одного из блоков (создать диаграмму следующего уровня).
6. Проверить ICOM-коды (связи между уровнями).
7. Сгенерировать отчёт и сохранить проект.
8. Предоставить скриншоты и файл проекта преподавателю.

Содержание отчёта:

- Скриншоты всех созданных диаграмм.
- Описание назначения каждого блока.
- Таблица ICOM-кодов (связи между уровнями).
- Краткое описание работы в Ramus.

Контрольные вопросы:

1. Какие функциональные возможности Ramus Educational вы использовали?
2. Как выполняется переход на нижний уровень декомпозиции?
3. Что такое ICOM-коды и для чего они нужны?
4. Какие требования предъявляются к диаграммам декомпозиции в IDEF0?

ПРАКТИЧЕСКАЯ РАБОТА №3

Тема: Построение диаграмм структурного моделирования (UML)

Цель:

Освоить построение структурных диаграмм UML (вариантов использования, классов, компонентов, развертывания) для описания статической архитектуры программной системы.

Задание:

Для заданной предметной области (например, «Система управления библиотекой», «Интернет-магазин», «Система бронирования билетов») разработать:

1. Диаграмму вариантов использования (Use Case Diagram).
2. Диаграмму классов (Class Diagram) – не менее 5 классов.
3. Диаграмму компонентов (Component Diagram).
4. Диаграмму развертывания (Deployment Diagram).

Порядок выполнения:

1. Изучить постановку задачи (выдаётся преподавателем).
2. Определить актантов (пользователей и внешние системы).
3. Определить функциональные требования (варианты использования).
4. В средстве моделирования (StarUML, [Draw.io](https://draw.io), PlantUML) построить:
 - Use Case Diagram – актанты, варианты использования, связи include/extend.
 - Class Diagram – классы, атрибуты, методы, связи (ассоциации, агрегация, композиция, наследование).
 - Component Diagram – компоненты и их интерфейсы.
 - Deployment Diagram – узлы (серверы, клиенты), артефакты и связи.
5. Оформить отчёт.

Содержание отчёта:

- Описание предметной области.
- Все 4 диаграммы (скриншоты).
- Краткое пояснение к каждой диаграмме (ключевые элементы и связи).
- Выводы.

Контрольные вопросы:

1. Какие типы связей между классами вы использовали и почему?
2. Чем диаграмма компонентов отличается от диаграммы классов?
3. Какую информацию содержит диаграмма развертывания?
4. В чём разница между include и extend на диаграмме вариантов использования?

ПРАКТИЧЕСКАЯ РАБОТА №4

Тема: Построение диаграмм последовательности UML

Цель:

Освоить моделирование временной упорядоченности взаимодействий между объектами с использованием диаграмм последовательности.

Задание:

На основе разработанной в работе №3 диаграммы вариантов использования детализировать один из сценариев (например, «Оформление заказа», «Поиск книги», «Регистрация пользователя») в виде диаграммы последовательности.

Порядок выполнения:

1. Выбрать один вариант использования из работы №3.
2. Описать основной поток событий (текстовый сценарий).
3. Определить участвующие объекты (классы из диаграммы классов).
4. В средстве моделирования построить диаграмму последовательности:
 - Добавить линии жизни для каждого объекта.
 - Добавить синхронные и асинхронные сообщения между объектами.
 - Указать временную последовательность (сверху вниз).
 - Добавить фрагменты альтернатив (alt), циклов (loop), параллелизма (par), если требуется.
5. Оформить отчёт.

Содержание отчёта:

- Описание выбранного сценария (текстовое).
- Диаграмма последовательности (скриншот).

- Пояснение ключевых сообщений и их последовательности.
- Выводы.

Контрольные вопросы:

1. Чем диаграмма последовательности отличается от диаграммы деятельности?
2. Что такое синхронное и асинхронное сообщение?
3. Как изображается цикл на диаграмме последовательности?
4. Как на диаграмме последовательности показать альтернативные ветви (if-else)?
5. Какую информацию даёт фрагмент par (parallel)?

6. Оценочные средства для проведения промежуточной аттестации

6.1 Вопросы для собеседования при проведении дифференцированного зачета

1. Дайте определение моделирования в контексте разработки ПО. Какие задачи решает моделирование?
2. Что такое UML? Назовите основные группы диаграмм и их назначение.
3. Что такое диаграмма классов? Перечислите основные элементы и их обозначения.
4. Какие виды отношений между классами существуют в UML? Дайте характеристику каждому.
5. В чём разница между агрегацией и композицией? Приведите примеры.
6. Что такое диаграмма компонентов и для каких целей она применяется?
7. Что такое диаграмма развертывания и какую информацию она содержит?
8. Что такое диаграмма деятельности и какие задачи она решает?
9. Назовите основные элементы диаграммы деятельности и их обозначения.
10. Как на диаграмме деятельности изображается ветвление (решение) и слияние?
11. Как моделируется параллельное выполнение действий на диаграмме деятельности?
12. Что такое дорожки (swimlanes) и для чего они используются?
13. Что такое диаграмма последовательности и для чего она применяется?
14. Какие элементы используются на диаграмме последовательности?
15. Что такое диаграмма состояний и в каких случаях её применяют?

16. Что такое диаграмма вариантов использования и какие элементы она содержит?
17. Что такое Model-Driven Architecture (MDA) и какова её роль в разработке ПО?
18. Перечислите основные метрики качества моделей ПО. Что такое связность и сцепление?

6.2 Тестовые материалы

Вариант №1

Часть 1. Выбор одного правильного ответа (1–3)

1. Какой вид диаграммы UML используется для описания статической структуры системы (классы, атрибуты, методы)?

- А) Диаграмма деятельности
- Б) Диаграмма классов
- В) Диаграмма последовательности
- Г) Диаграмма состояний

2. Что такое CASE-средство?

- А) Язык программирования
- Б) Инструмент для автоматизации процессов моделирования и разработки ПО
- В) Методология управления проектами
- Г) Система управления базами данных

3. Какой принцип объектно-ориентированного проектирования подразумевает, что класс должен иметь только одну причину для изменения?

- А) Принцип подстановки Барбары Лисков (LSP)
- Б) Принцип открытости/закрытости (OCP)
- В) Принцип единственной ответственности (SRP)
- Г) Принцип инверсии зависимостей (DIP)

Часть 2. Установление соответствия (4–6)

4. Установите соответствие между типом диаграммы UML и её назначением.

Тип диаграммы	Назначение
1. Диаграмма вариантов использования	А. Описывает последовательность обмена сообщениями между объектами
2. Диаграмма последовательности	Б. Показывает переходы между состояниями объекта

3. Диаграмма состояний	В. Описывает функциональные требования и взаимодействие пользователя с системой
------------------------	---

Ответ запишите в виде сочетания цифр и букв (например, 1-В, 2-А, 3-Б).

5. Установите соответствие между метрикой качества ПО и её описанием.

Метрика	Описание
1. Связность (Cohesion)	А. Степень взаимозависимости между модулями (чем меньше, тем лучше)
2. Сцепление (Coupling)	Б. Степень, в которой элементы модуля принадлежат друг другу (чем выше, тем лучше)
3. Цикломатическая сложность	В. Количество независимых путей в коде, отражает сложность тестирования

6. Установите соответствие между этапом моделирования и его артефактом.

Этап	Артефакт
1. Анализ требований	А. Диаграмма компонентов
2. Проектирование архитектуры	Б. Use Case (варианты использования)
3. Детальное проектирование	В. Диаграмма классов с детальными атрибутами и методами

Часть 3. Установление последовательности (7–9)

7. Установите правильную последовательность этапов объектно-ориентированного анализа и проектирования (по Rational Unified Process):

1. Реализация
2. Анализ требований
3. Проектирование
4. Моделирование бизнес-процессов

5. Развёртывание

8. Расставьте шаги создания диаграммы классов в правильном порядке:

1. Определение связей (ассоциации, агрегации, наследования)
 2. Выделение классов на основе требований
 3. Определение атрибутов и методов для каждого класса
 4. Уточнение видимости (public, private, protected)
 5. Проверка модели на соответствие принципам SOLID
 9. **Установите последовательность действий при использовании CASE-средства для генерации кода:**
 1. Настройка параметров генерации
 2. Создание модели в инструменте
 3. Проверка модели на ошибки
 4. Генерация кода (например, Java, C#)
 5. Импорт сгенерированного кода в проект
-

Часть 4. Открытые вопросы (10–12)

10. **Что такое диаграмма деятельности UML? Для каких целей она используется? Приведите пример.**
11. **Перечислите основные принципы SOLID и кратко охарактеризуйте каждый.**
12. **В чём разница между функциональными и нефункциональными требованиями? Приведите по одному примеру.**

Вариант №2

Часть 1. Выбор одного правильного ответа (1–3)

1. **Какой тип диаграммы UML лучше всего подходит для описания алгоритма работы функции или бизнес-процесса?**
 - А) Диаграмма классов
 - Б) Диаграмма последовательности
 - В) Диаграмма деятельности
 - Г) Диаграмма компонентов

2. **Что такое метрика «цикломатическая сложность»?**

- А) Количество строк кода
- Б) Количество независимых путей выполнения
- В) Количество классов в системе
- Г) Время выполнения программы

3. **Какое из перечисленных относится к нефункциональным требованиям?**

- А) Система должна позволять добавлять товары в корзину
- Б) Система должна обрабатывать до 1000 запросов в секунду
- В) Система должна предоставлять отчёты по продажам
- Г) Система должна поддерживать регистрацию пользователей

Часть 2. Установление соответствия (4–6)

4. **Установите соответствие между принципом SOLID и его описанием.**

Принцип	Описание
1. Открытости/закрытости (OCP)	А. Класс должен иметь только одну причину для изменения
2. Единственной ответственности (SRP)	Б. Программные сущности должны быть открыты для расширения, но закрыты для модификации
3. Инверсии зависимостей (DIP)	В. Модули верхнего уровня не должны зависеть от модулей нижнего уровня, оба должны зависеть от абстракций

5. **Установите соответствие между видом анализа и его задачей.**

Вид анализа	Задача
1. Статический анализ	А. Выявление ошибок во время выполнения (утечки памяти, производительность)
2. Динамический анализ	Б. Проверка кода на соответствие правилам, стилю и потенциальным ошибкам без запуска
3. Анализ требований	В. Выявление противоречий, неполноты и неоднозначности в требованиях к ПО

6. **Установите соответствие между элементом модели и его назначением.**

Элемент	Назначение
1. Актор (Actor)	А. Описывает взаимодействие объектов, упорядоченное по времени

2. Сообщение	Б. Внешняя сущность, взаимодействующая с системой
3. Линия жизни (Lifeline)	В. Передача информации между объектами в диаграмме последовательности

Часть 3. Установление последовательности (7–9)

7. Установите правильную последовательность шагов разработки модели в методологии IDEF0:

1. Определение контекста и цели моделирования
2. Построение иерархии диаграмм (декомпозиция)
3. Выделение основных функций системы
4. Сбор информации и интервью с экспертами
5. Проверка модели на согласованность

8. Расставьте в правильном порядке этапы тестирования модели на соответствие требованиям (верификация):

1. Проверка синтаксической корректности модели
2. Проверка полноты (все требования отражены)
3. Проверка согласованности (нет противоречий между диаграммами)
4. Проверка выполнимости (реализуемость)
5. Оценка соответствия заданным метрикам качества

9. Установите порядок действий при переходе от модели к коду (генерация кода из UML):

1. Выбор языка программирования и платформы
2. Настройка отображения типов UML на типы языка
3. Создание профиля для генерации
4. Генерация скелета классов и интерфейсов
5. Добавление бизнес-логики вручную

Часть 4. Открытые вопросы (10–12)

10. **Объясните, для чего используется диаграмма последовательности в UML. Приведите пример её применения.**
11. **Какие типы связей между классами существуют в UML? Опишите их (наследование, агрегация, композиция, ассоциация). Приведите примеры.**
12. **Что такое CASE-средства? Какие задачи они решают и какие возможности предоставляют разработчику?**

6.3 Ключи к тестовым заданиям для проведения промежуточной аттестации

Вариант №1

Часть 1 (выбор):

1 – Б, 2 – Б, 3 – В

Часть 2 (соответствие):

4 – 1-В, 2-А, 3-Б

5 – 1-Б, 2-А, 3-В

6 – 1-Б, 2-А, 3-В

Часть 3 (последовательность):

7 – 4, 2, 3, 1, 5 (или 2,3,4,1,5 – зависит от конкретной методологии, но логично: сначала моделирование бизнес-процессов, затем анализ требований, потом проектирование, реализация, развёртывание) – я предлагаю: 2, 4, 3, 1, 5 (анализ требований → моделирование бизнес-процессов → проектирование → реализация → развёртывание).

8 – 2, 1, 3, 4, 5

9 – 2, 3, 1, 4, 5

Вариант №2

Часть 1 (выбор):

1 – В, 2 – Б, 3 – Б

Часть 2 (соответствие):

4 – 1-Б, 2-А, 3-В

5 – 1-Б, 2-А, 3-В

6 – 1-Б, 2-В, 3-А

Часть 3 (последовательность):

7 – 1, 4, 3, 2, 5 (или 1,3,4,2,5 – я выберу: 1, 4, 3, 2, 5)

8 – 1, 2, 3, 4, 5

9 – 1, 2, 3, 4, 5

Примеры ответов на открытые вопросы

Вариант №1

10. Диаграмма деятельности UML – это диаграмма, показывающая поток управления от одной активности к другой. Используется для моделирования бизнес-процессов, алгоритмов, рабочих потоков. Пример: диаграмма процесса оформления заказа в интернет-магазине (проверка корзины, оплата, отправка подтверждения).

11. Принципы SOLID:

- **S** – единственной ответственности: у класса одна задача.
- **O** – открытости/закрытости: класс должен быть открыт для расширения, закрыт для модификации.
- **L** – подстановки Лисков: наследники должны заменять родителей без нарушения логики.
- **I** – разделения интерфейсов: интерфейсы должны быть специфичными, не нужно заставлять классы реализовывать ненужные методы.
- **D** – инверсии зависимостей: зависимости строятся на абстракциях, а не на конкретных классах.

12. Функциональные требования – описывают, что система должна делать (например, «система должна позволять пользователю регистрироваться»). **Нефункциональные** – описывают качества системы (например, «система должна обрабатывать до 1000 запросов в секунду»).

Вариант №2

10. Диаграмма последовательности – показывает взаимодействие объектов во времени, обмен сообщениями. Используется для моделирования сценариев (например, для заказа в магазине: клиент → контроллер → сервис → БД, с ответами). Это позволяет увидеть порядок вызовов и взаимодействие компонентов.

11. Типы связей:

- **Ассоциация** – общая связь между классами (например, Студент учится в Группе).
- **Наследование** (обобщение) – «is a»: Автомобиль – Транспортное средство.
- **Агрегация** – «has a» (часть-целое, но части могут существовать отдельно): например, Департамент содержит Сотрудников.
- **Композиция** – «has a» с жёсткой зависимостью: например, Дом состоит из Комнат, при разрушении дома комнаты исчезают.

13. CASE-средства – инструменты, автоматизирующие процесс моделирования, разработки и документирования ПО. Возможности: построение диаграмм, генерация кода, анализ моделей, управление требованиями, проверка согласованности, реверсинжиниринг.

Все ответы могут быть сформулированы в другой близкой по смыслу формулировке

6.4 Ключи к вопросам для собеседования при проведении дифференцированного зачета

1. Дайте определение моделирования в контексте разработки ПО. Какие задачи решает моделирование?

Моделирование – это процесс создания абстрактного представления (модели) разрабатываемой системы, которое отражает её структуру, поведение или данные на нужном уровне детализации.

Задачи моделирования:

- Визуализация – представление сложной системы в наглядной форме для всех участников проекта.
- Спецификация – формальное описание того, что система должна делать и как она устроена.
- Документирование – фиксация проектных решений в виде, доступном для последующего использования.
- Анализ и верификация – проверка корректности, полноты и непротиворечивости требований и архитектуры.
- Генерация артефактов – автоматическое создание кода, тестов и документации на основе моделей.

2. Что такое UML? Назовите основные группы диаграмм и их назначение.

UML (Unified Modeling Language) – стандартизированный язык визуального моделирования для описания, визуализации, проектирования и документирования программных систем.

Основные группы диаграмм:

Структурные (статические):

- Диаграмма классов – структура классов и их связи.
- Диаграмма компонентов – архитектура на уровне модулей.
- Диаграмма развертывания – физическое размещение на узлах.
- Диаграмма пакетов – организация модели в пакеты.
- Диаграмма объектов – экземпляры классов в конкретный момент.

Поведенческие (динамические):

- Диаграмма деятельности – потоки управления и данных.
- Диаграмма последовательности – взаимодействие объектов во времени.
- Диаграмма состояний – жизненный цикл объектов.
- Диаграмма вариантов использования – функциональные требования системы.
- Диаграмма коммуникации – взаимодействие объектов в контексте структуры.

3. Что такое диаграмма классов? Перечислите основные элементы и их обозначения.

Диаграмма классов – структурная диаграмма UML, описывающая статическую структуру системы: классы, их атрибуты, методы и взаимосвязи.

Основные элементы:

- Класс – прямоугольник, разделённый на 3 секции: имя, атрибуты, методы.
- Атрибуты – данные класса: -имя: Тип (например, -name: String).
- Методы – операции: +имя(): Тип (например, +getName(): String).
- Интерфейс – контракт, который реализует класс.
- Связи: ассоциация (----), агрегация (◇----), композиция (◆----), наследование (△----), реализация (△---- пунктир).

Модификаторы доступа: + (public), - (private), # (protected), ~ (package).

4. Какие виды отношений между классами существуют в UML? Дайте характеристику каждому.

Отношение	Обозначение	Характеристика
-----------	-------------	----------------

Ассоциация	----	Связь между объектами классов. Может быть двусторонней.
Наследование (общение)	△----	Отношение «является разновидностью». Дочерний класс наследует родительский.
Реализация	△---- (пунктир)	Класс реализует интерфейс.
Агрегация	◇----	Отношение «часть-целое», где часть существует отдельно от целого.
Композиция	◆----	Жёсткая агрегация: часть не существует без целого.
Зависимость	---- > (пунктир)	Временная связь: один класс использует другой (например, в параметре метода).

5. В чём разница между агрегацией и композицией? Приведите примеры.

Агрегация – отношение «часть-целое», где части могут существовать независимо от целого. Обозначается пустым ромбом.

Пример: Автомобиль и колёса. Колёса можно снять и использовать на другом автомобиле.

Композиция – более жёсткая форма агрегации, где части не могут существовать без целого. Обозначается закрашенным ромбом.

Пример: Заказ и его позиции (строки заказа). Если удалить заказ, позиции тоже удаляются.

6. Что такое диаграмма компонентов и для каких целей она применяется?

Диаграмма компонентов – структурная диаграмма UML, показывающая компоненты системы и зависимости между ними.

Компонент – заменяемый модуль, который имеет чётко определённые интерфейсы и может быть заменён независимо.

Применение:

- Визуализация архитектуры на уровне крупных модулей.
- Описание интерфейсов и зависимостей между подсистемами.
- Планирование развёртывания (какие компоненты на какие серверы).
- Документирование структуры кода и конфигураций.

7. Что такое диаграмма развёртывания и какую информацию она содержит?

Диаграмма развёртывания – структурная диаграмма UML, описывающая физическое размещение артефактов (компонентов, файлов) на узлах (серверах, устройствах).

Содержит:

- Узлы (Node) – физические устройства (сервер, ПК, мобильное устройство).
- Артефакты – файлы (JAR, DLL, исполняемые файлы).
- Связи между узлами – протоколы, порты, сетевые соединения.
- Информацию о среде выполнения – ОС, версия, ресурсы.

Применяется для документирования инфраструктуры и планирования развёртывания.

8. Что такое диаграмма деятельности и какие задачи она решает?

Диаграмма деятельности – поведенческая диаграмма UML, описывающая поток управления и данных от одного действия к другому. Аналог блок-схемы, но с более мощной семантикой.

Решаемые задачи:

- Моделирование бизнес-процессов и рабочих процедур.
- Документирование алгоритмов внутри методов.
- Описание параллельных и альтернативных потоков.
- Визуализация сложных сценариев использования.
- Анализ и оптимизация процессов.

9. Назовите основные элементы диаграммы деятельности и их обозначения.

Элемент	Обозначение	Описание
Действие (Action)	Прямоугольник со скруглёнными углами	Выполняемая операция.
Начальный узел (Initial)	Закрашенный кружок	Точка начала потока.
Конечный узел (Final)	Кружок в круге	Точка завершения деятельности.
Решение (Decision)	Ромб	Точка ветвления по условию.
Слияние (Merge)	Ромб	Объединение нескольких потоков.
Fork (разветвление)	Горизонтальная линия	Разделение на параллельные потоки.
Join (объединение)	Горизонтальная линия	Объединение параллельных потоков.
Дорожка (Swimlane)	Вертикальная/горизонтальная полоса	Группировка по ответственности.

10. Как на диаграмме деятельности изображается ветвление (решение) и слияние?

- Решение (Decision Node) – изображается ромбом с одним входящим потоком и несколькими (минимум двумя) исходящими. Каждый исходящий поток помечается условием (например, [да], [нет]). Поток направляется только по одному из путей в зависимости от выполнения условия.
- Слияние (Merge Node) – изображается ромбом с несколькими входящими потоками и одним исходящим. Используется для объединения альтернативных путей после решения. Не проверяет условия, просто соединяет потоки в один.

11. Как моделируется параллельное выполнение действий на диаграмме деятельности?

Параллельное выполнение моделируется с помощью двух специальных узлов:

- Fork (разветвление) – жирная горизонтальная линия с одним входящим и несколькими исходящими потоками. Разделяет поток на несколько параллельных.
- Join (объединение) – жирная горизонтальная линия с несколькими входящими и одним исходящим потоком. Объединяет параллельные потоки. Исходящий поток выполняется только после завершения всех входящих.

Пример: Система одновременно получает данные и обновляет интерфейс; после завершения обоих действий переходит к следующему шагу.

12. Что такое дорожки (swimlanes) и для чего они используются?

Дорожки (Swimlanes) – это вертикальные или горизонтальные полосы на диаграмме деятельности, которые разделяют действия по исполнителям, ролям, отделам или системам.

Использование:

- Показывают, кто выполняет каждое действие.
- Отображают распределение ответственности в процессе.
- Улучшают читаемость и связь с организационной структурой.

Пример: Дорожки «Клиент», «Менеджер», «Склад» показывают, как процесс оформления заказа распределён между участниками.

13. Что такое диаграмма последовательности и для чего она применяется?

Диаграмма последовательности (Sequence Diagram) – поведенческая диаграмма UML, показывающая взаимодействие объектов во времени. Она детализирует сценарии использования, показывая порядок обмена сообщениями между объектами.

Применение:

- Детализация сценариев вариантов использования.
- Моделирование протоколов взаимодействия между компонентами.

- Визуализация временной последовательности вызовов.
- Анализ критических сценариев и поиск узких мест.

14. Какие элементы используются на диаграмме последовательности?

- Актанты (Actor) – внешние сущности, взаимодействующие с системой.
- Линии жизни (Lifeline) – вертикальная пунктирная линия, представляющая существование объекта (или актанта) во времени.
- Фокус управления (Execution Specification) – прямоугольник на линии жизни, обозначающий период выполнения операции.
- Сообщения (Messages):
 - Синхронное (запрос с ожиданием ответа) – сплошная стрелка.
 - Асинхронное – сплошная стрелка с открытым наконечником.
 - Возврат – пунктирная стрелка.
- Альтернативные фрагменты (alt) – условные ветвления (аналог if-else).
- Циклы (loop) – повторяющиеся фрагменты.
- Фрагменты параллельного выполнения (par) – параллельные потоки.

15. Что такое диаграмма состояний и в каких случаях её применяют?

Диаграмма состояний (State Machine Diagram) – поведенческая диаграмма UML, описывающая жизненный цикл объекта: состояния, переходы между ними, события и действия.

Применение:

- Моделирование поведения объектов со сложной логикой переходов (например, заказ, документ, жизненный цикл задачи).
- Описание реактивных систем (встроенные системы, управление устройствами).
- Документирование бизнес-правил, связанных с состояниями.
- Улучшение понимания того, как объект реагирует на события в разные моменты времени.

16. Что такое диаграмма вариантов использования и какие элементы она содержит?

Диаграмма вариантов использования (Use Case Diagram) – поведенческая диаграмма UML, описывающая функциональные требования системы с точки зрения пользователя (актантов). Она показывает, какие функции система предоставляет и кто ими пользуется.

Элементы:

- Актант (Actor) – внешняя сущность (человек, система), взаимодействующая с системой. Изображается стикменом.
- Вариант использования (Use Case) – отдельная функция системы. Изображается овалом с названием.
- Границы системы (System Boundary) – прямоугольник, внутри которого находятся варианты использования.
- Отношения:
 - Ассоциация (между актантом и вариантом) – сплошная линия.
 - include – один вариант обязательно включает другой (пунктир со стрелкой).
 - extend – расширение функциональности (дополнительный вариант при условии).

17. Что такое Model-Driven Architecture (MDA) и какова её роль в разработке ПО?

Model-Driven Architecture (MDA) – подход к разработке ПО, предложенный OMG, при котором модели являются основными артефактами, а код и документация генерируются из них автоматически.

Основные уровни моделей:

- CIM (Computation Independent Model) – модель на уровне бизнеса, без учёта технологий.
- PIM (Platform Independent Model) – модель, независимая от платформы реализации.

- PSM (Platform Specific Model) – модель, адаптированная к конкретной платформе (Java, .NET).

Роль: MDA автоматизирует преобразование моделей и генерацию кода, повышает производительность, обеспечивает согласованность между документацией и кодом, упрощает адаптацию к новым технологиям.

18. Перечислите основные метрики качества моделей ПО. Что такое связность и сцепление?

Основные метрики качества моделей:

Метрика	Описание
Связность (Cohesion)	Степень, в которой элементы внутри модуля (класса) связаны друг с другом. Высокая связность – хорошо (модуль делает одну задачу).
Сцепление (Coupling)	Степень зависимостей между модулями (классами). Низкое сцепление – хорошо (минимум зависимостей).
Цикломатическая сложность	Число линейно независимых путей в коде (для анализа сложности методов).
Глубина наследования	Количество уровней в иерархии наследования.
Количество методов	Число методов в классе (признак перегруженности).
Количество зависимостей	Число связей с другими классами.

Связность – показывает, насколько тесно связаны элементы внутри одного класса/модуля. Высокая связность означает, что модуль делает одну задачу, легко поддерживается и тестируется.

Сцепление – показывает, как много другие модули знают о внутреннем устройстве данного. Низкое сцепление означает, что изменения в одном модуле слабо влияют на другие, что упрощает модификацию и повторное использование.

Все ответы могут быть сформулированы в другой близкой по смыслу формулировке

6.5 Ключи на вопросы для устного опроса

1. Что такое модель в контексте разработки программного обеспечения и каковы цели моделирования?

Модель – это абстрактное представление разрабатываемой системы, отражающее её структуру, поведение или данные с определённой степенью детализации. Модель упрощает реальность, выделяя важные аспекты и скрывая второстепенные детали.

Цели моделирования:

- Визуализация – сделать проект понятным для всех участников (заказчиков, аналитиков, разработчиков).
- Спецификация – точно описать, что и как должна делать система.
- Документирование – создание единого источника истины для всей команды.
- Анализ – проверка корректности, полноты и непротиворечивости требований.
- Генерация кода – автоматическое создание программного кода из моделей (MDA).

2. Какие преимущества дает использование моделей на этапе проектирования ПО?

- Снижение сложности – модель помогает управлять сложностью системы, разбивая её на понятные компоненты.
- Обнаружение ошибок на ранних этапах – выявление несоответствий и противоречий до начала кодирования, что удешевляет исправление.
- Улучшение коммуникации – модели служат общим языком между заказчиками и разработчиками.

- Повторное использование – готовые модели (или их части) могут быть использованы в других проектах.
- Стандартизация – обеспечивает единый подход к проектированию в команде.
- Автоматизация – возможность генерации кода, документации и тестов на основе моделей.

3. Что такое UML и для каких целей он используется в разработке ПО?

UML (Unified Modeling Language) – это стандартизированный язык визуального моделирования, предназначенный для описания, визуализации, проектирования и документирования программных систем.

Цели использования:

- Создание наглядных схем (диаграмм) для всех участников проекта.
- Формализация требований и проектных решений.
- Унификация обозначений, чтобы все разработчики понимали модели одинаково.
- Документирование архитектуры, структуры и поведения системы.
- Использование в качестве основы для генерации кода и тестов.

4. Назовите основные группы диаграмм UML и приведите примеры из каждой группы.

UML 2.x делит диаграммы на две основные группы:

1. Структурные диаграммы – описывают статическую структуру системы:

- Диаграмма классов (Class Diagram)
- Диаграмма компонентов (Component Diagram)
- Диаграмма развертывания (Deployment Diagram)
- Диаграмма пакетов (Package Diagram)
- Диаграмма объектов (Object Diagram)

2. Поведенческие диаграммы – описывают динамическое поведение системы:

- Диаграмма вариантов использования (Use Case Diagram)
- Диаграмма деятельности (Activity Diagram)
- Диаграмма последовательности (Sequence Diagram)
- Диаграмма состояний (State Machine Diagram)
- Диаграмма коммуникации (Communication Diagram)

5. Опишите основные принципы объектно-ориентированного подхода, которые находят отражение в UML-моделировании.

- Инкапсуляция – скрытие внутреннего устройства объекта и предоставление доступа только через открытый интерфейс (в UML отражается с помощью модификаторов доступа +, -, #).
- Наследование – возможность создания нового класса на основе существующего с добавлением или переопределением свойств (в UML обозначается стрелкой с пустым треугольником).
- Полиморфизм – способность объектов разных классов реагировать на одни и те же вызовы по-разному (отражается через реализацию интерфейсов и переопределение методов).
- Абстракция – выделение существенных характеристик объекта и игнорирование несущественных (в UML представляется абстрактными классами и интерфейсами).

6. Что показывает диаграмма классов и какие основные элементы на ней изображаются?

Диаграмма классов – это структурная диаграмма UML, описывающая статическую структуру системы: классы, их атрибуты, методы и связи между ними.

Основные элементы:

- Класс – прямоугольник, разделённый на три секции: имя класса, атрибуты, методы.
- Атрибуты – свойства класса (например, -name: String).
- Методы (операции) – действия, которые может выполнять класс (например, +getName(): String).

- Интерфейс – объявление контракта, который реализуют классы.
- Связи: ассоциация, агрегация, композиция, наследование, реализация.

7. Как на диаграмме классов обозначаются модификаторы доступа?

В UML используются следующие обозначения перед атрибутами и методами:

- + – public (общедоступный)
- - – private (закрытый, доступен только внутри класса)
- # – protected (защищённый, доступен внутри класса и его наследникам)
- ~ – package (доступен в пределах пакета)

Пример: -balance: double – приватное поле balance с типом double.

8. Что такое ассоциация между классами? Какие виды ассоциаций вы знаете?

Ассоциация – это отношение между классами, показывающее, что объекты одного класса связаны с объектами другого класса.

Виды ассоциаций:

- Обычная ассоциация – двусторонняя связь между классами (например, «Студент учится в Университете»).
- Агрегация – отношение «часть-целое», где часть может существовать отдельно от целого (ромб пустой).
- Композиция – более жёсткая форма агрегации, где часть не может существовать без целого (ромб закрашенный).
- Зависимость – временная связь, показывающая, что один класс использует другой (пунктирная стрелка).
- Наследование (обобщение) – отношение «является разновидностью» (стрелка с пустым треугольником).
- Реализация – класс реализует интерфейс (пунктирная стрелка с пустым треугольником).

9. Чем агрегация отличается от композиции? Приведите примеры.

- Агрегация – отношение «часть-целое», где части могут существовать независимо от целого. Обозначается пустым ромбом.

Пример: Автомобиль и колёса. Если автомобиль утилизировать, колёса можно использовать на другом автомобиле.

- Композиция – жёсткая форма агрегации, где части не могут существовать без целого. Обозначается закрашенным ромбом.

Пример: Заказ и товарные позиции в заказе. Если заказ удалить, его позиции также исчезают.

10. Что такое диаграмма компонентов и для чего она используется?

Диаграмма компонентов – структурная диаграмма UML, показывающая компоненты системы и зависимости между ними.

Компонент – модуль системы с чётко определёнными интерфейсами, который может быть заменён независимо.

Для чего используется:

- Визуализация архитектуры системы на уровне модулей.
- Описание интерфейсов и зависимостей между компонентами.
- Организация кода в проекте (разбивка по подсистемам).
- Документирование развёртывания компонентов на серверах.

11. Что такое диаграмма развертывания и какую информацию она передает?

Диаграмма развертывания – структурная диаграмма UML, описывающая физическое размещение компонентов системы на вычислительных узлах (серверах, устройствах).

Передаваемая информация:

- Топология сети (какие узлы используются).
- Какие компоненты установлены на каждом узле.
- Протоколы и способы связи между узлами.
- Ресурсы (память, процессор) и их распределение.

Используется на этапе планирования инфраструктуры и для документирования эксплуатационной модели.

12. Что такое диаграмма пакетов и как она помогает организовать структуру проекта?

Диаграмма пакетов – структурная диаграмма, показывающая организацию модели в виде пакетов и зависимости между ними.

Пакет – контейнер для группировки элементов модели (классов, компонентов, диаграмм).

Как помогает:

- Упрощает навигацию по большой модели.
- Отражает иерархию и модульную структуру проекта.
- Показывает зависимости между подсистемами.
- Служит основой для организации кода в пространствах имён.

13. Что такое диаграмма деятельности и для чего она применяется?

Диаграмма деятельности – поведенческая диаграмма UML, описывающая поток управления и данных от одной деятельности (действия) к другой. Она похожа на блок-схему, но более мощная.

Для чего применяется:

- Моделирование бизнес-процессов и рабочих процедур.
- Описание алгоритмов в рамках методов классов.
- Документирование потоков данных и управления в системе.
- Анализ параллельных процессов.
- Согласование требований с заказчиками через наглядные схемы.

14. Какие основные элементы используются на диаграмме деятельности?

- Действие (Action) – выполняемая операция (прямоугольник со скруглёнными углами).
- Узел решения (Decision Node) – точка ветвления потока по условию (ромб).
- Узел слияния (Merge Node) – точка слияния нескольких потоков (ромб).
- Начало (Initial Node) – стартовый узел (закрашенный кружок).
- Конец (Final Node) – конечный узел (закрашенный кружок в круге).
- Узлы параллельного выполнения – Fork (разветвление) и Join (объединение) – жирные горизонтальные линии.
- Дорожки (Swimlanes) – разделение диаграммы по исполнителям или подсистемам.

15. Как обозначается начало и конец потока на диаграмме деятельности?

- Начало потока (Initial Node) – закрашенный кружок (●) – обозначает точку старта.
- Конец потока (Flow Final Node) – закрашенный кружок в круге (⊕) – обозначает завершение конкретного потока.

- Конец деятельности (Final Node) – закрашенный кружок в круге (⊕) с толстым ободком – обозначает полное завершение деятельности (всех потоков).

На практике часто используют просто кружок в круге.

16. Что такое "решение" и "слияние" на диаграмме деятельности?

- Решение (Decision Node) – точка ветвления, где поток расходится по одному из нескольких путей в зависимости от условия. Изображается ромбом с входящим и несколькими (минимум двумя) исходящими потоками. Исходящие потоки помечаются условиями (например, «[да]» и «[нет]»).

- Слияние (Merge Node) – точка объединения нескольких альтернативных потоков в один. Изображается ромбом с несколькими входящими и одним исходящим потоком. Используется для соединения ветвей после разветвления.

17. Как изображается параллельное выполнение действий на диаграмме деятельности?

Параллельное выполнение изображается с помощью двух узлов:

- Fork (разветвление) – разделяет один поток на несколько параллельных потоков. Обозначается жирной горизонтальной линией с одним входящим и несколькими исходящими потоками.

- Join (объединение) – соединяет несколько параллельных потоков в один. Обозначается жирной горизонтальной линией с несколькими входящими и одним исходящим потоком. Все входящие потоки должны завершиться, прежде чем выполнится исходящий.

Пример: система одновременно загружает данные и обновляет интерфейс – после завершения обоих действий переходит к финальной обработке.

18. Что такое "дорожки" (swimlanes) на диаграмме деятельности и для чего они используются?

Дорожки (Swimlanes) – это вертикальные или горизонтальные полосы, которые разделяют диаграмму деятельности на зоны ответственности.

Использование:

- Группируют действия по исполнителям (ролям, отделам, системам).
- Показывают, кто выполняет какие действия.
- Улучшают читаемость диаграммы и связывают действия с организационной структурой.

Пример: дорожки «Клиент», «Менеджер», «Склад» — показывают, как процесс оформления заказа распределён между этими участниками.

Все ответы могут быть сформулированы в другой близкой по смыслу формулировке