

**Федеральное государственное автономное образовательное
учреждение высшего образования
«Московский физико-технический институт
(национальный исследовательский университет)»**

УТВЕРЖДЕНО

**Директор физтех-школы
прикладной математики и
информатики**

А.М. Райгородский

| | |
|----------------------------|---|
| | Рабочая программа дисциплины (модуля) |
| по дисциплине: | Проектирование программных систем |
| по направлению: | Прикладная математика и информатика |
| профиль подготовки: | Прикладная математика, компьютерные науки и инженерия Физтех-школа Прикладной Математики и Информатики кафедра алгоритмов и технологий программирования |
| курс: | 4 |
| квалификация: | бакалавр |

Семестр, формы промежуточной аттестации: 7 (осенний) - Дифференцированный зачет

Аудиторных часов: 60 всего, в том числе:

лекции: 30 час.

семинары: 30 час.

лабораторные занятия: 0 час.

Самостоятельная работа: 75 час.

Всего часов: 135, всего зач. ед.: 3

Программу составил: А.С. Хританков, канд. физ.-мат. наук, доцент

Программа обсуждена на заседании кафедры алгоритмов и технологий программирования 01.06.2023

Аннотация

Дисциплина проектирования программного обеспечения (software design) является одной из ключевых для будущих архитекторов, аналитиков, руководителей проектов, инженеров и разработчиков ПО. Область проектирования охватывает концепции, модели, методы, программные средства для создания, анализа, развития, сопровождения структур сложных многокомпонентных программных систем, создаваемых совместно с другими разработчиками, как в небольших, так и в крупномасштабных проектах разработки. Владение и применение полученными при изучении дисциплины знаниями, умениями и навыками повышает качество разрабатываемых систем, снижает затраты на их разработку и сопровождение, позволяет быстрее адаптироваться к изменениям в области разработки программ, понимать тренды развития области и перспективные направления создания программного обеспечения.

1. Цели и задачи

Цель дисциплины

- Познакомить студентов с объектно-ориентированными и структурными методами разработки программных систем с применением технологий моделирования;
- дать представление о существующих методологиях проектирования программного обеспечения и выработать у студентов практические навыки по их применению.

Задачи дисциплины

- освоение студентами базовых знаний в области программной инженерии, моделирования и проектирования программных систем;
- приобретение теоретических знаний в области объектно-ориентированного, структурного проектирования и моделирования программных систем;
- приобретение практических навыков по применению унифицированного языка моделирования;
- приобретение практических навыков командной работы над программными системами;
- приобретение навыков работы с современными инструментами моделирования и проектирования.

2. Перечень формируемых компетенций

Освоение дисциплины направлено на формирование следующих компетенций:

| Код и наименование компетенции | Индикаторы достижения компетенции |
|---|---|
| ОПК-2 Способен использовать современные информационные технологии и программные средства при решении задач профессиональной деятельности, соблюдая требования информационной безопасности | ОПК-2.1 Способен применять современные вычислительную технику и сервисы сети Интернет в области (сфере) профессиональной деятельности |
| | ОПК-2.2 Знает и умеет применять численные математические методы и прикладное программное обеспечение для решения научных задач в профессиональной области |
| | ОПК-2.3 Знает основные требования информационной безопасности |
| ОПК-3 Способен составлять и оформлять научные и (или) технические (технологические, инновационные) отчеты (публикации, проекты) | ОПК-3.1 Знает основные правила оформления научных публикаций и научно-технической документации, в том числе с использованием прикладного программного обеспечения |
| | ОПК-3.2 Владеет на практике методологией составления научно-технических отчетов (проектов) |
| | ОПК-3.3 Владеет методами визуального и графического представления результатов научной (научно-технической, инновационной технологической) деятельности в виде отчетов, научных публикаций |

| | |
|---|--|
| ПК-2 Способен самостоятельно или в качестве члена (руководителя) малого коллектива организовывать и проводить научные исследования и их апробацию | ПК-2.1 Знает принципы построения научной работы, методы сбора и анализа полученного материала, способы аргументации |
| | ПК-2.2 Способен планировать и проводить научные исследования самостоятельно или в качестве члена (руководителя) малого научного коллектива |
| | ПК-2.3 Способен проводить апробацию результатов научно-исследовательской работы посредством публикации научных статей и участия в конференциях |

3. Перечень планируемых результатов обучения по дисциплине (модулю)

В результате освоения дисциплины обучающиеся должны

знать:

- основы и внутреннюю структуру унифицированного языка моделирования UML, основные понятия метамодели языка и отношения между ними;
- средства UML для представления логических и концептуальных моделей, нотацию диаграмм классов;
- представление использования, диаграммы вариантов использования;
- моделирование поведения и динамики информационных систем средствами UML, представление взаимодействия, диаграммы последовательности, обмена сообщениями, фрагменты, семантика взаимодействия в UML;
- средства унифицированного языка для представления внутренней структуры программных систем, повторно-используемых модулей, компонентов;
- представление реализации, воплощение элементов модели в артефактах, размещение артефактов по вычислительным узлам;
- средства для моделирования поведения объектов с помощью схем состояний в представлении конечных автоматов, диаграммы схем состояний, принцип перехода по завершении;
- моделирование потоков работ и вычислительных алгоритмов с помощью сете Петри в представлении деятельности, действия, принцип прохода до завершения;
- средства управления сложностью моделей, механизмы расширения UML, стереотипы, профили;
- методы структурного моделирования и проектирования, метод структурного проектирования Джексона (JSP), метод постепенного уточнения (stepwise refinement), нотацию структурных схем и диаграмм потоков данных DFD;
- метод структурного анализа и проектирования SSA/SD и его варианты;
- виды декомпозиции: процедурная/алгоритмическая, по данным, по сценариям/функциям, критерии качества структуры дизайна: связность и сходство, критерии и эвристики декомпозиции ПО на модули: anticipate change, information hiding, separation of concerns;
- основные архитектурные стили, клиент-сервер, каналы-фильтры, монолитное приложение, слоистая архитектура, обмен сообщениями и др.
- паттерны проектирования GoF и применение к практическим задачам разработки ПО: в том числе Template method, Visitor, Builder, Facade, Decorator, Bridge, State и другие;
- основы объектно-ориентированного анализа, методику проектирования на основе обязанностей, метод класс-контракт-коллеги (CRC), метод Аббота выделения потенциальных классов;
- принципы проектирования. OCP, LSP, DIP, ISP, SRP; эвристики назначения обязанностей GRASP;
- метод проектирования и разработки объектно-ориентированных систем ICONIX
- методы количественной оценки качества программных систем, сложности структуры системы, набор показателей Чидамбера-Кемерера.

уметь:

- обосновать принятые проектные решения в области проектирования ПО;
- самостоятельно разрабатывать согласованную модель программной системы, удовлетворяющую функциональным требованиям;
- представлять выполненный проект для обсуждения в аудитории;
- применять методы проектирования при разработке ПО;
- использовать современные интегрированные средства разработки и проектирования (IDE);
- выбирать наиболее подходящий для решения проблемы метод проектирования;
- применять методы структурного и объектно-ориентированного анализа и проектирования при разработке ПО;
- использовать унифицированный язык моделирования для описания предметных областей и структур программ;
- оценивать качество разработанного дизайна ПО.

владеть:

- навыками самостоятельной работы в современных программных комплексах;
- навыками освоения большого объёма информации;
- навыками совместной командной работы над программными системами.

4. Содержание дисциплины (модуля), структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий

4.1. Разделы дисциплины (модуля) и трудоемкости по видам учебных занятий

| № | Тема (раздел) дисциплины | Трудоемкость по видам учебных занятий, включая самостоятельную работу, час. | | | |
|-----------------------|--|---|----------|-----------------|----------------|
| | | Лекции | Семинары | Лаборат. работы | Самост. работа |
| 1 | Введение в инженерии программного обеспечения | 2 | 2 | | 8 |
| 2 | Применение моделей в разработке программ | 2 | 2 | | 8 |
| 3 | Статическое представление модели | 5 | 5 | | 7 |
| 4 | Динамическое представление модели | 5 | 5 | | 7 |
| 5 | Семестровая контрольная работа | 2 | 2 | | 7 |
| 6 | Методы структурного проектирования | 2 | 2 | | 7 |
| 7 | Введение в архитектуру ПО | 3 | 3 | | 8 |
| 8 | Методы и паттерны объектно-ориентированного проектирования | 5 | 5 | | 8 |
| 9 | Документирование архитектуры и дизайна | 2 | 2 | | 8 |
| 10 | Курсовой проект. Консультации по проектам | 2 | 2 | | 7 |
| Итого часов | | 30 | 30 | | 75 |
| Подготовка к экзамену | | 0 час. | | | |
| Общая трудоёмкость | | 135 час., 3 зач.ед. | | | |

4.2. Содержание дисциплины (модуля), структурированное по темам (разделам)

Семестр: 7 (Осенний)

1. Введение в инженерии программного обеспечения

Различия между программным и аппаратным обеспечением. Показатели качества ПО. Виды ПО. Цели и задачи проектирования. Отличия проектирования от анализа или исследования. Основные определения в инженерии ПО.

2. Применение моделей в разработке программ

Понятие о моделировании и проектировании. Структурное моделирование. История создания UML. Структура и основные понятия метамодели UML. Представления модели, виды диаграмм. Место моделей в процессах жизненного цикла проектов. Обзор методики ICONIX, основные этапы, применяемые методы и эвристики. Применение моделей в гибкой разработке.

3. Статическое представление модели

Диаграммы классов. Понятия класса, интерфейса, типа данных. Виды отношений: ассоциация, зависимость, абстракция, реализация, обобщение и другие. Экземпляры классов. Ограничения. Структурированный классификатор. Композит и часть. Диаграммы внутренней структуры. Архитектурное моделирование, компоненты, порты, соединители. Размещение системы, воплощение компонентов, узлы, артефакты. Профили. Расширение модели. Пакеты. Управление моделью.

4. Динамическое представление модели

Поведение. Основные определения. Варианты использования (прецеденты). Описание требований при помощи прецедентов. Представление взаимодействия. Диаграммы взаимодействия и коммуникации. Основные понятия: роль, спецификация выполнения, сообщение. Кооперация. Описание сценариев вариантов использования. Представление деятельности. Представление о сетях Петри. Виды действий, разделы. Контекст выполнения. Потоки управления и данных (объектные). Представление процессов на диаграммах деятельности. Представление конечных автоматов, схемы состояний. Состояние, переход, псевдосостояния, составные состояния. Обработка событий, переход по завершении. Моделирование жизненного цикла классификатора с помощью конечных автоматов.

5. Семестровая контрольная работа

Проведение контрольной работы.

6. Методы структурного проектирования

Основные понятия. Модуль. Процесс структурного проектирования. Виды методов: сверху-вниз (нисходящие), снизу-вверх (восходящие), итеративные. Модульность. Принципы разделения системы на модули. Метод постепенного уточнения (stepwise refinement), структурные диаграммы (STD). Методика структурного анализа/проектирования (SSA/SD). Элементарные транзакции. Диаграммы потоков данных (DFD). Метод структурного программирования Джексона (JSP). Разбор примеров применения.

7. Введение в архитектуру ПО

Архитектура и структура программной системы. Факторы, влияющие на архитектуру. Заинтересованные лица. Роль архитектуры. Стандартные архитектурные стили: вызов-возврат, каналы-фильтры, многоуровневая / клиент-сервер, сервис-ориентированная, событийно-ориентированная (event-sourcing), распределенная одноранговая, конструктор (toolkit). Применение структурных методов к проектированию архитектуры систем обработки данных. Метод SSA/SD и его варианты.

8. Методы и паттерны объектно-ориентированного проектирования

Введение в объектно-ориентированный анализ. Выделение классов. Методы построения модели предметной области (метод Аббота, метод именных групп, контрольные списки). Абстрактные типы данных. Проектирование на основе обязанностей (RDD). Карточки Класс-Контракт-Коллеги (CRC). Эвристики GRASP. Принципы ООП SOLID (SRP, OCP, LSP, ISP, DIP). Применение паттернов проектирования (GoF). Показатели качества объектно-ориентированной структуры. Комплект показателей Чидамбера-Кемерера. Показатели сложности и объема кода МакКейба и Халстеда. Измерение сложности и сопровождаемости ПО.

9. Документирование архитектуры и дизайна

Роль документирования. Понятие о точке зрения. Система представлений 4+1. Рекомендации и структура документа.

10. Курсовой проект. Консультации по проектам

Обсуждение принципов ООП. Разбор примеров применения паттернов. Разбор примеров построения модели реализации.

Модельно-ориентированный подход к разработке (MDD/MDA). Платформонезависимая модель и платформозависимая модель (PIM/PSM). Сервис-ориентированная архитектура. Поток работ (workflow). Использование каркасов приложений (framework). Проблемы проектирования распределенных объектных систем. Понятие о транзакциях. Другие темы.

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

Необходимое оборудование для лекций и практических занятий:

- компьютерный класс (компьютеры/ноутбуки с выходом в Интернет);
- презентационная техника (мультимедийный проектор, экран, компьютер/ноутбук с выходом в Интернет).

6. Перечень рекомендуемой литературы

Основная литература

1. UML. Основы : Краткое руководство по стандартному языку объектного моделирования [Текст] : [учеб. пособие для вузов] / М. Фаулер ; пер. с англ. А. Петухова ; предисл. К. Кобрина [и др.] . — 3-е изд. — СПб. : Символ-Плюс, 2009 . — 192 с.
2. Проектирование на UML [Текст], сборник задач /А. С. Хританков, В. А. Полежаев, А. И. Андрианов. -Екатеринбург, Издательские решения, 2017
3. Приемы объектно-ориентированного проектирования. Паттерны проектирования [Текст] /Э. Гамма [и др.] ; [пер. с англ. А. Слинкина], [учеб. пособие для вузов]. -СПб., Питер, 2018

Дополнительная литература

1. Проектирование пакетов/И. Ю. Межуев, А. С. Хританков ; Министерство науки и высшего образования Российской Федерации, Московский физико-технический институт (национальный исследовательский университет), Кафедра алгоритмов и технологий программирования , Москва, МФТИ, 2019

7. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", необходимых для освоения дисциплины (модуля)

1. Веб-страничка курса, <http://pps-design.org>
2. Введение в гибкое моделирование (www.agilemodeling.com)
3. INTUIT.RU: Интернет-Университет Информационных Технологий: Программирование: Объектно-ориентированное программирование (<http://www.intuit.ru/catalog/se/objectprog/>).
4. InformIT. Статьи по ИТ и разработке (www.informit.com)
5. Спецификация UML 2.4 на сайте OMG. (<http://www.omg.org/spec/UML/2.4/>)

8. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень необходимого программного обеспечения и информационных справочных систем (при необходимости)

Программное обеспечение:

- Средства моделирования ПО (MagicDraw UML Academic Edition или Modelio или Visual Paradigm UML).
- Офисный пакет (MS Office или LibreOffice).
- Средства разработки ПО (MS Visual Studio, IntelliJ IDEA CE, PyCharm).
- ПО промежуточного контроля освоения студентами дисциплины (доступное через Интернет).

9. Методические указания для обучающихся по освоению дисциплины (модуля)

Успешное освоение курса требует напряжённой самостоятельной работы студента на протяжении всего курса. В программе курса приведено минимально необходимое время для работы студента над темами курса. Самостоятельная работа включает в себя:

- проработку учебного материала (по конспектам лекций, учебной и научной литературе);
- подготовку к практическим занятиям, выполнение домашних заданий
- выполнение курсовых заданий

Промежуточный контроль знаний проводится в виде письменных проверочных работ по теории, семестровой контрольной работы, а также студенту в ходе освоения курса необходимо выполнить две домашние командные работы с их последующей защитой:

1. разработка модели анализа учебной программной системы.
2. разработка модели реализации учебной программной системы и реализация прототипа согласно модели.

Методические указания по выполнению домашних и курсовых заданий см. в списке учебно-методического обеспечения.

Для подготовки к семестровой контрольной работе по курсу рекомендуется пользоваться конспектами лекций или литературой из списка рекомендованной и дополнительной. Настоятельно не рекомендуется использовать непроверенные Интернет-источники.

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)

| | |
|----------------------------|---|
| по направлению: | Прикладная математика и информатика |
| профиль подготовки: | Прикладная математика, компьютерные науки и инженерия Физтех-школа Прикладной Математики и Информатики кафедра алгоритмов и технологий программирования |
| курс: | <u>4</u> |
| квалификация: | бакалавр |

Семестр, формы промежуточной аттестации: 7 (осенний) - Дифференцированный зачет

Разработчик: А.С. Хританков, канд. физ.-мат. наук, доцент

1. Компетенции, формируемые в процессе изучения дисциплины

| Код и наименование компетенции | Индикаторы достижения компетенции |
|---|---|
| ОПК-2 Способен использовать современные информационные технологии и программные средства при решении задач профессиональной деятельности, соблюдая требования информационной безопасности | ОПК-2.1 Способен применять современные вычислительную технику и сервисы сети Интернет в области (сфере) профессиональной деятельности |
| | ОПК-2.2 Знает и умеет применять численные математические методы и прикладное программное обеспечение для решения научных задач в профессиональной области |
| | ОПК-2.3 Знает основные требования информационной безопасности |
| ОПК-3 Способен составлять и оформлять научные и (или) технические (технологические, инновационные) отчеты (публикации, проекты) | ОПК-3.1 Знает основные правила оформления научных публикаций и научно-технической документации, в том числе с использованием прикладного программного обеспечения |
| | ОПК-3.2 Владеет на практике методологией составления научно-технических отчетов (проектов) |
| | ОПК-3.3 Владеет методами визуального и графического представления результатов научной (научно-технической, инновационной технологической) деятельности в виде отчетов, научных публикаций |
| ПК-2 Способен самостоятельно или в качестве члена (руководителя) малого коллектива организовывать и проводить научные исследования и их апробацию | ПК-2.1 Знает принципы построения научной работы, методы сбора и анализа полученного материала, способы аргументации |
| | ПК-2.2 Способен планировать и проводить научные исследования самостоятельно или в качестве члена (руководителя) малого научного коллектива |
| | ПК-2.3 Способен проводить апробацию результатов научно-исследовательской работы посредством публикации научных статей и участия в конференциях |

2. Показатели оценивания компетенций

В результате изучения дисциплины «Проектирование программных систем» обучающийся должен:

знать:

- основы и внутреннюю структуру унифицированного языка моделирования UML, основные понятия метамодели языка и отношения между ними;
- средства UML для представления логических и концептуальных моделей, нотацию диаграмм классов;
- представление использования, диаграммы вариантов использования;
- моделирование поведения и динамики информационных систем средствами UML, представление взаимодействия, диаграммы последовательности, обмена сообщениями, фрагменты, семантика взаимодействия в UML;
- средства унифицированного языка для представления внутренней структуры программных систем, повторно-используемых модулей, компонентов;
- представление реализации, воплощение элементов модели в артефактах, размещение артефактов по вычислительным узлам;
- средства для моделирования поведения объектов с помощью схем состояний в представлении конечных автоматов, диаграммы схем состояний, принцип перехода по завершении;
- моделирование потоков работ и вычислительных алгоритмов с помощью сети Петри в представлении деятельности, действия, принцип прохода до завершения;
- средства управления сложностью моделей, механизмы расширения UML, стереотипы, профили;
- методы структурного моделирования и проектирования, метод структурного проектирования Джексона (JSP), метод постепенного уточнения (stepwise refinement), нотацию структурных схем и диаграмм потоков данных DFD;
- метод структурного анализа и проектирования SSA/SD и его варианты;
- виды декомпозиции: процедурная/алгоритмическая, по данным, по сценариям/функциям, критерии качества структуры дизайна: связность и сходство, критерии и эвристики декомпозиции ПО на модули: anticipate change, information hiding, separation of concerns;
- основные архитектурные стили, клиент-сервер, каналы-фильтры, монолитное приложение, слоистая архитектура, обмен сообщениями и др.
- паттерны проектирования GoF и применение к практическим задачам разработки ПО: в том числе Template method, Visitor, Builder, Facade, Decorator, Bridge, State и другие;
- основы объектно-ориентированного анализа, методику проектирования на основе обязанностей, метод класс-контракт-коллеги (CRC), метод Аббота выделения потенциальных классов;
- принципы проектирования. OCP, LSP, DIP, ISP, SRP; эвристики назначения обязанностей GRASP;
- метод проектирования и разработки объектно-ориентированных систем ICONIX
- методы количественной оценки качества программных систем, сложности структуры системы, набор показателей Чидамбера-Кемерера.

уметь:

- обосновать принятые проектные решения в области проектирования ПО;
- самостоятельно разрабатывать согласованную модель программной системы, удовлетворяющую функциональным требованиям;
- представлять выполненный проект для обсуждения в аудитории;
- применять методы проектирования при разработке ПО;
- использовать современные интегрированные средства разработки и проектирования (IDE);
- выбирать наиболее подходящий для решения проблемы метод проектирования;
- применять методы структурного и объектно-ориентированного анализа и проектирования при разработке ПО;
- использовать унифицированный язык моделирования для описания предметных областей и структур программ;
- оценивать качество разработанного дизайна ПО.

владеть:

- навыками самостоятельной работы в современных программных комплексах;
- навыками освоения большого объема информации;
- навыками совместной командной работы над программными системами.

3. Перечень типовых (примерных) вопросов, заданий, тем для подготовки к текущему контролю

Курсовые задания:

В ходе освоения дисциплины студенты выполняют два командных проектных курсовых задания. Постановки курсовых заданий уточняются ежегодно.

Первое задание включает:

1. Модель анализа предметной области - диаграммы вариантов использования, диаграммы классов, схемы состояний.
2. Описание ключевых вариантов использования в текстовом виде (4-6 штук) и их реализации классами модели.
3. Артефакты разработки (допустимо фотографии с бумаги): списки потенциальных классов, карточки CRC, промежуточные варианты диаграмм

Второе задание включает:

1. Задание 1 (модель анализа, описание вариантов использования, диаграммы, обоснование)
2. Модель реализации – описание и обоснование выбора архитектуры, описание реализации с применением паттернов проектирования и других проектировочных решений.
3. Прототип реализации ключевых вариантов использования
4. План тестирования

Также необходимо обосновать представленную модель, то есть, почему выявлены именно такие классы, почему важен тот или иной вариант использования, почему необходимо моделировать данную ассоциацию и т. п.

Задания направляются преподавателю для проверки в установленный срок, преподаватель согласно методическим рекомендациям выполняет проверку задания и направляет оформленные результаты проверки студенту. Защита задания с учетом выявленных замечаний происходит на семинаре в формате презентации.

Для проверки одного курсового задания из расчета на одного студента следует предусмотреть 30 минут работы преподавателя. Оценка за задания учитывается отдельно в итоговом балле.

Проверочные работы:

Практическая работа студентов на семинарах предусматривает выполнение заданий и решение задач по проектированию и моделированию.

Предусмотрены следующие темы проверочных работ:

1. Статическое представление. Классы. Внутренняя структура.
2. Динамическое представление. Взаимодействия. Деятельность. Схемы состояний.
3. Объектно-ориентированные методы. Паттерны проектирования.

Для выполнения проверочной работы следует предусмотреть от 15 до 20 минут. Для проверки одного задания одного студента следует предусмотреть 15 минут работы преподавателя. Оценка за проверочные работы учитывается в составе оценки за практическую работу студента.

Семестровая контрольная работа:

Письменная контрольная работа, выполняемая студентами в течение двух академических часов на лекции ближе к завершению курса.

Темы контрольной включают большую часть тем курса. Контрольная работа состоит из следующих заданий:

1. Многовариантный тест на знание унифицированного языка моделирования (по аналогии с сертификационными экзаменами по UML).
2. Две задачи на применение методов проектирования и моделирования программных систем по несколько подпунктов в каждой.
3. Вопросы по теории по темам курса.

Оценка за контрольную работу учитывается отдельно в итоговом балле.

Контрольная работа предлагается в четырех вариантах. Задачи для контрольной работы (всего восемь задач) обновляются ежегодно.

Для составления восьми задач для контрольной работы следует предусмотреть 16 часов работы преподавателя.

Для проверки одной контрольной работы из расчета на одного студента следует предусмотреть 30 минут работы преподавателя.

Примерный перечень названий курсовых проектных заданий для командного выполнения:

1. «СмартРитейл». Интегрируем недоверчивых участников рынка ритейла с помощью блокчейна
2. «Ok, Fridge». Холодильник доставит к дивану соки и воды под настроение, дозакажет в магазине
3. «ЗалейсяIS». Управляем сетью авто-автозаправок, работающих без персонала
4. «БезВрача». Контролируем соблюдение условий медицинского страхования, и вообще соблюдения правил ЗОЖ
5. «ПочтаШеринг». Едем вместе на чужой машине и везем посылки. Система обеспечивает работу сервиса
6. «Как?Никак!». Модернизированный контактный центр, интеллектуальные агенты отвечают на типовые вопросы и сами ведут FAQ
7. «ДоставимВсе». Служба доставки формирует из товаров заказы и перемещает по маршрутам. Маршруты формируются вручную, автоматически как план, динамически. Учет веса и стоимости при планировании
8. «Физтех-Такси». Агрегатор такси со скидкой на поездки в МФТИ

4. Перечень типовых (примерных) вопросов и тем для проведения промежуточной аттестации обучающихся

1. История создания UML.
2. Основные понятия: элемент, классификатор, черта, отношение, пространство имен.
3. Представления модели, виды диаграмм.
4. Диаграммы классов. Понятия класса, интерфейса, типа данных.
5. Виды отношений: ассоциация, зависимость, абстракция, реализация и другие.
6. Абстрактные классы, наследование. Множественное наследование.
7. Объектные диаграммы. Экземпляры классов.
8. Варианты использования (прецеденты). Описание требований при помощи прецедентов.
9. Поведение. Основные определения.
10. Структурированный классификатор. Композит и часть. Диаграммы внутренней структуры.
11. Компонент, порт, делегирующий и сборочный соединитель. Динамический порт.
12. Представление взаимодействия. Диаграммы взаимодействия и коммуникации. Основные понятия: роль, спецификация выполнения, сообщение.
13. Синхронные и асинхронные сообщения. Фрагменты.
14. Семантика взаимодействия в UML. Последовательность сообщений.
15. Кооперация. Описание сценариев вариантов использования.
16. Представление деятельности. Сети Петри.
17. Виды действий, разделы. Контекст выполнения.
18. Потоки управления и данных (объектные). Параметры деятельности.
19. Управляющие действия.
20. Представление процессов на диаграммах деятельности.
21. Семантика моделей деятельности в UML. Переход до завершения.
22. Представление конечных автоматов. Диаграммы схем состояний.
23. Состояние, переход, псевдосостояния, составные состояния.
24. Ортогональные состояния и вложенные автоматы.

25. Семантика конечных автоматов в UML. Обработка событий, выполнение до завершения.
26. Моделирование жизненного цикла классификатора с помощью конечных автоматов.
27. Пакеты. Управление моделью.
28. Размещение. Узел, артефакт, материализация. Путь коммуникации. Спецификация развертывания.
29. Понятие качества ПО. Характеристики качества программного продукта
30. Введение в программную инженерию. Модели жизненного цикла ПО. Проект и процесс.
31. Понятие о декомпозиции. Модули. Степени связности и сходства.
32. Структурное проектирование. Основная теорема структурного программирования, метод структурного проектирования Джексона, структурные схемы.
33. Проектирование систем обработки данных. Представление потоков данных. Нотация DFD (Gane-Sarson). Метод SSA/SD.
34. Методы построения модели предметной области. Метод Аббота.
35. Объектно-ориентированный анализ. Понятие об обязанностях. Метод CRC.
36. Представление функциональных требований в виде вариантов использования. Структура описания.
37. Критерии и эвристики декомпозиции: anticipate change, information hiding, separation of concerns.
38. Метод постепенного уточнения (stepwise refinement).
39. Принципы проектирования. OCP, LSP, DIP, ISP, SRP.
40. Применение паттернов проектирования: расширение обязанностей классов (наследование, Template Method, Decorator), реализация схем состояний (State, switch), обход и выполнение действий на графе (Visitor, Iterator), создание экземпляров и семейств экземпляров (Builder, Abstract Factory)
41. Абстрактные типы данных. Определение. Применение в ООП.
42. Понятие об архитектуре. Архитектурные стили: Call-and-Return, Pipes-and-Filters, Layered, Multitier. SOA/Workflow, Client-Server, Blackboard/Data-centered.
43. Количественные показатели (метрики) программных продуктов и проектов. Метрики модульной структуры, fan-in, fan-out. Сложность Халстеда, цикломатическая сложность.
44. Количественные показатели качества ОО дизайнера, набор показателей Чидамбера-Кеммерера.

Критерии оценивания

отлично

10 всесторонние, систематизированные, глубокие знания учебной программы дисциплины и умение уверенно применять их на практике при решении конкретных задач, свободное и правильное обоснование принятых решений;

9 систематизированные, глубокие знания учебной программы дисциплины и умение уверенно применять их на практике при решении конкретных задач, правильное обоснование принятых решений;

8 глубокие знания учебной программы дисциплины и умение применять их на практике при решении конкретных задач, правильное обоснование принятых решений;

хорошо

7 твердо знает материал, грамотно и по существу излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач некоторые неточности;

6 знает материал, грамотно излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач некоторые неточности;

5 знает основной материал, грамотно излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач неточности;

удовлетворительно

4 фрагментарный, разрозненный характер знаний, недостаточно правильные формулировки базовых понятий, нарушения логической последовательности в изложении программного материала, но при этом он владеет основными разделами учебной программы, необходимыми для дальнейшего обучения и может применять полученные знания по образцу в стандартной ситуации;

3 характер знаний достаточен для дальнейшего обучения и может применять полученные знания по образцу в стандартной ситуации;

неудовлетворительно

2 не знает большей части основного содержания учебной программы дисциплины, допускает грубые ошибки в формулировках основных понятий дисциплины и не умеет правильно использовать полученные знания при решении типовых практических задач.

1 не знает формулировок основных понятий дисциплины и не умеет использовать полученные знания при решении типовых практических задач.

5. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности

Время подготовки к ответу рекомендуется устанавливать не менее 30 минут. Время на ответ – не более 15 минут на каждый вопрос. Суммарное время проведения дифференцированного зачета для одного студента не должно превышать 90 минут (двух академических часов).