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

«УТВЕРЖДАЮ
Проректор по учебной работе
и экономическому развитию
_____ Д.А. Зубцов

Рабочая программа дисциплины (модуля)

по дисциплине: Проектирование программных систем
по направлению: Прикладная математика и информатика (бакалавриат)
профиль подготовки: Прикладная математика и информатика (общий)
факультет инноваций и высоких технологий
кафедра алгоритмов и технологий программирования
курс: 4
квалификация: бакалавр

Семестр, формы промежуточной аттестации: 7(Осенний) - Экзамен

Аудиторных часов: 60 всего, в том числе:
лекции: 30 час.
практические и семинарские занятия: 30 час.
лабораторные занятия: 0 час.

Самостоятельная работа: 45 час.

Подготовка к экзамену: 30 час.

Всего часов: 135, всего зач. ед.: 3

Количество курсовых работ, заданий: 2

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

Программа обсуждена на заседании кафедры

23 февраля 2017 г.

СОГЛАСОВАНО:

Заведующий кафедрой М.А. Ройтберг

Начальник учебного управления И.Р. Гарайшина

Декан факультета В.Е. Кривцов

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. Место дисциплины (модуля) в структуре образовательной программы

Данная дисциплина относится к вариативной части образовательной программы

Дисциплина «Проектирование программных систем» базируется на дисциплинах:

Алгоритмы и структуры данных;

Английский язык;

Объектно-ориентированное программирование;

Формальные языки и трансляции.

Дисциплина «Проектирование программных систем» предшествует изучению дисциплин:

Тестирование и верификация ПО;

Конструирование программного обеспечения;

Анализ требований к программным системам.

3. Перечень планируемых результатов обучения по дисциплине (модулю), соотнесенных с планируемыми результатами освоения образовательной программы

Освоение дисциплины направлено на формирование следующих общекультурных, общепрофессиональных и профессиональных компетенций:

способность работать в команде, толерантно воспринимая социальные, этнические, конфессиональные и культурные различия (ОК-6);

способность к самоорганизации и самообразованию (ОК-7);

способность использовать базовые знания естественных наук, математики и информатики, основные факты, концепции, принципы теорий, связанных с прикладной математикой и информатикой (ОПК-1);

способность приобретать новые научные и профессиональные знания, используя современные образовательные и информационные технологии (ОПК-2);

способность к разработке алгоритмических и программных решений в области системного и прикладного программирования, математических, информационных и имитационных моделей, созданию информационных ресурсов глобальных сетей, образовательного контента, прикладных баз данных, тестов и средств тестирования систем и средств на соответствие стандартам и исходным требованиям (ОПК-3);

способность осуществлять целенаправленный поиск информации о новейших научных и технологических достижениях в сети Интернет и из других источников (ПК-5);

способность к разработке и применению алгоритмических и программных решений в области системного и прикладного программного обеспечения (ПК-7).

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, обмены 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), нотациию 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	Виды учебных занятий, включая самостоятельную работу				
		Лекции	Практичес-кие и семинарские занятия	Лаборат. работы	Задания, курсовые работы	Самост. работа
1	Введение в инженерию программного обеспече-ния	1	2			
2	Применение моделей в разработке программ	3	5			8
3	Статическое представление модели	5	5			
4	Динамическое представление модели	5	4			
5	Семестровая контрольная работа	2	2			
6	Методы структурного проектирования	2	2			
7	Введение в архитектуру ПО	3				
8	Методы и паттерны объектно-ориентированного проектирования	8	6			
9	Документирование архитектуры и дизайна	1				
10	Курсовой проект. Консультации по проектам		4			37
Итого часов		30	30			45
Подготовка к экзамену		30 час.				
Общая трудоёмкость		135 час., 3 зач.ед.				

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

Семестр: 7 (Осенний)

1. Введение в инженерию программного обеспече-ния

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

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. Потоки 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 основной и дополнительной литературы, необходимой для освоения дисциплины (модуля)

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. Сборник задач по проектированию программных систем [Текст] : [учеб. пособие для вузов] / А. С. Хританков, А. Н. Штукатуров, А. И. Андрианов .— М. : [Б. и.], 2012 .— 104 с.

3. Приемы объектно-ориентированного проектирования. Паттерны проектирования [Текст] : [учеб. пособие для вузов] / Э. Гамма [и др.] ; [пер. с англ. А. Слинкина] .— СПб. : Питер, 2012 .— 368 с.

4. Технологии разработки программного обеспечения. Разработка сложных программных систем [Текст] : учеб. пособие для вузов / С. А. Орлов .— 2-е изд. — СПб. : Питер, 2003 .— 480 с.

5. Функциональное и логическое программирование [Текст] : учеб. пособие для вузов / А. Л. Ездаков .— М. : БИНОМ. Лаб. знаний, 2009 .— 119 с.

6. Применение UML 2.0 и шаблонов проектирования: Введение в объективно-ориентированный анализ, проектирование и итеративную разработку/Applying UML and Patterns : практическое руководство / К. Ларман .— 3-е изд. — М;СПб;Киев : Изд. дом "Вильямс", 2009 .— 736 с.

Дополнительная литература

1. Проектирование и конструирование компиляторов [Текст] : [учеб. пособие для вузов] / Р. Хантер ; пер. с англ. С. М. Круговой ; под ред. В. М. Савинкова .— М. : Финансы и статистика, 1984 .— 544 с.

7. Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине (модулю)

1. Сборник задач по проектированию программных систем [Текст] : [учеб. пособие для вузов] / А. С. Хританков, А. Н. Штукатуров, А. И. Андрианов .— М. : [Б. и.], 2012 .— 104 с.

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

1. Веб-страничка курса, http://pps-fall2012.dyndns.org
1. Введение в гибкое моделирование (www.agilemodeling.com)
2. INTUIT.RU: Интернет-Университет Информационных Технологий: Программирование: Объектно-ориентированное программирование (http://www.intuit.ru/catalog/se/objectprog/).
3. InformIT. Статьи по ИТ и разработке (www.informit.com)
4. Спецификация UML 2.4 на сайте OMG. (http://www.omg.org/spec/UML/2.4/)

9. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень 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)

10. Методические указания для обучающихся по освоению дисциплины

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.

Литература для самостоятельного изучения:

1. R. Pressman. Software Engineering: A Practitioner's Approach, 6th Ed. - McGraw Hill, 2005
2. Г, Буч, Д. Рамбо, А. Джекобсон. UML. Классика CS. 2-е изд. / Пер. с англ.; Под общей редакцией проф. С. Орлова – СПб.: Питер, 2006. – 736 с.: ил.
3. Л. Басс, П. Клементс, Р. Кацман. Архитектура программного обеспечения на практике. 2-е изд. – Л.:Питер, 2006. – 576 с.
4. С. Амблер, Гибкие технологии: экстремальное программирова¬ние и унифицированный процесс разработки. – СПб.: Питер, 2005. – 412 с.: ил.
5. Мейер Б. Объектно-ориентированное конструирование программных систем. – М.: Интер-нет-университет информационных технологий, 2005.
6. B. Liskov, J. Guttag. Program Development in Java: Abstraction, Specification and Object-Oriented Design. - Addison-Wesley, 2000.
7. Rebecca Wirfs-Brock and Alan McKean. Object Design: Roles, Responsibilities, and Collaborations, Addison-Wesley 2003
8. D. Esposito, A. Saltarello. Microsoft .NET - Architecting Applications for the Enterprise (Developer Reference) 2nd Ed. – MS Press, 2016
9. Хританков А.С. Объектно-ориентированные технологии проектирования и язык UML. Учебно-методическое пособие. – М.: МФТИ, 2012.

11. Фонд оценочных средств для проведения промежуточной аттестации по итогам обучения

Приложение

ПРИЛОЖЕНИЕ

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ ПРОВЕДЕНИЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ОБУЧАЮЩИХСЯ ПО ДИСЦИПЛИНЕ

по направлению: Прикладная математика и информатика (бакалавриат)
профиль подготовки: Прикладная математика и информатика (общий)
Факультет инноваций и высоких технологий
кафедра алгоритмов и технологий программирован
курс: 4
квалификация: бакалавр

Семестр, формы промежуточной аттестации: 7(Осенний) - Экзамен

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

1. Компетенции, формируемые в процессе изучения дисциплины

Освоение дисциплины направлено на формирование у обучающегося следующих общекультурных (ОК), общепрофессиональных (ОПК) и профессиональных (ПК) компетенций:

способность работать в команде, толерантно воспринимая социальные, этнические, конфессиональные и культурные различия (ОК-6);

способность к самоорганизации и самообразованию (ОК-7);

способность использовать базовые знания естественных наук, математики и информатики, основные факты, концепции, принципы теорий, связанных с прикладной математикой и информатикой (ОПК-1);

способность приобретать новые научные и профессиональные знания, используя современные образовательные и информационные технологии (ОПК-2);

способность к разработке алгоритмических и программных решений в области системного и прикладного программирования, математических, информационных и имитационных моделей, созданию информационных ресурсов глобальных сетей, образовательного контента, прикладных баз данных, тестов и средств тестирования систем и средств на соответствие стандартам и исходным требованиям (ОПК-3);

способность осуществлять целенаправленный поиск информации о новейших научных и технологических достижениях в сети Интернет и из других источников (ПК-5);

способность к разработке и применению алгоритмических и программных решений в области системного и прикладного программного обеспечения (ПК-7).

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, обмены 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), нотациию 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.

3. Перечень типовых контрольных заданий, используемых для оценки знаний, умений, навыков

Курсовые задания:

В ходе освоения дисциплины студенты выполняют два 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 контрольных вопросов для сдачи экзамена:

1. Структура метамодели UML. Основные понятия: элемент, классификатор, черта, отношение, пространство имен. Представления модели, виды диаграмм.
2. Диаграммы классов. Понятия класса, интерфейса, типа данных. Виды отношений: ассоциация, зависимость, абстракция, реализация и другие.
3. Абстрактные классы, обобщение и множество обобщений. Контекст переопределения. Объектные диаграммы. Экземпляры классов и связи.
4. Варианты использования (прецеденты). Описание требований при помощи вариантов использования и сценариев. Текстовое описание.
5. Структурированный классификатор. Композит и часть. Диаграммы внутренней структуры. Компонент, порт, делегирующий и сборочный соединитель. Динамический порт. Кооперации.
6. Моделирование поведения. Основные определения. Классы метамодели, отвечающие за моделирование поведения.
7. Представление взаимодействия. Основные понятия: роль, спецификация выполнения, сообщение. Синхронные и асинхронные сообщения. Фрагменты. Семантика взаимодействия в UML. Последовательность сообщений.
8. Представление деятельности. Сети Петри. Виды действий, разделы. Контекст выполнения. Потоки управления и данных (объектные). Параметры деятельности. Управляющие действия. Семантика моделей деятельности в UML. Переход до завершения.
10. Представление конечных автоматов. Диаграммы схем состояний. Состояние, переход, псевдосостояния, составные состояния. Ортогональные состояния и вложенные автоматы. Семантика конечных автоматов в UML. Обработка событий, выполнение до завершения.
11. Моделирование жизненного цикла классификатора с помощью конечных автоматов.
12. Пакеты. Управление моделью. Размещение. Узел, артефакт, материализация. Путь коммуникации. Спецификация развертывания.
13. Понятие качества ПО. Характеристики качества программного продукта. Понятие о декомпозиции. Определение модуля. Степени связности и сходства.
14. Структурное проектирование. Основная теорема структурного программирования, метод структурного проектирования Джексона, структурные схемы.
15. Проектирование систем обработки данных. Представление потоков данных. Нотация DFD (Gane-Sarson). Метод SSA/SD.
16. Методы построения модели предметной области. Метод Аббота. Метод именных групп. Контрольные списки
17. Объектно-ориентированный анализ. Проектирование на основе обязанностей (RDD). Метод CRC.
18. Критерии и эвристики декомпозиции: anticipate change, information hiding, separation of concerns.
19. Метод постепенного уточнения (stepwise refinement). Пример с ферзями.
20. Принципы проектирования. OCP, LSP, DIP, ISP, SRP.
21. Применение паттернов проектирования: расширение обязанностей классов (наследование, Template Method, Decorator), реализация схем состояний (State, switch),
22. Применение паттернов проектирования: обход и выполнение действий на графе (Visitor, Iterator), создание экземпляров и семейств экземпляров (Builder, Abstract Factory)
23. Абстрактные типы данных. Определение. Применение в ООП.

24. Понятие об архитектуре. Архитектурные стили: Call-and-Return, Pipes-and-Filters, Layered, Multitier. SOA/Workflow, Client-Server, Blackboard/Data-centered, Event Sourcing, P2P.

25. Методы разработки архитектуры систем обработки данных

26. Количественные показатели (метрики) программных продуктов и проектов. Метрики модульной структуры, fan-in, fan-out. Сложность Халстеда, цикломатическая сложность.

27. Количественные показатели качества ОО дизайна, набор показателей Чидамбера-Кемерера.

4. Критерии оценивания

Итоговый балл формируется следующим образом:

ИБ = 0,25 * ПР + 0,5 * КЗ + 0,5 * КР,

где ПР – оценка по десятибалльной шкале за активность на семинарах, выполнение текущих домашних заданий и домашнее чтение, КЗ – оценка по десятибалльной шкале за курсовые задания (нужно сдать оба), КР – оценка по десятибалльной шкале за семестровую контрольную работу.

Соответствие баллов итоговой оценке по курсу

Балл	Оценка за экзамен	Оценка
От 0 до 4.0	Менее 5	от 0 до 3
	от 5 до 8	от 2 до 5
	от 8 до 10	от 4 до 6
От 4.1 до 6.0	Менее 5	от 1 до 5
	от 5 до 7	от 3 до 7
	от 7 до 10	от 5 до 8
От 6.1 до 8.0	Без сдачи	от 3 до 4
	Менее 4	от 2 до 6
	от 4 до 6	от 5 до 7
	от 6 до 10	от 6 до 9
Свыше 8.1	Без сдачи	от 5 до 8
	Менее 4	от 4 до 7
	от 4 до 7	от 6 до 10
	от 7 до 10	от 9 до 10

Получение итоговой оценки по курсу на экзамене без ответа на билет возможно в случае, если оценка за контрольную работу не ниже 3 баллов.

Итоговая оценка по курсу выставляется на экзамене на усмотрение преподавателя с учетом приведенных рекомендаций.

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

Для проведения экзамена готовятся билеты по два вопроса. Время подготовки к ответу рекомендуется устанавливать не менее 30 минут. Время на ответ – не более 15 минут на каждый вопрос. Суммарное время проведения экзамена для одного студента не должно превышать 90 минут (двух академических часов).