МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«ТЮМЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ЛАБОРАТОРНЫЙ ПРАКТИКУМ
РАЗРАБОТКА ТРЕБОВАНИЙ И ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

09.03.02 Информационные системы и технологии
Профиль: Интернет-технологии и разработка WEB-приложений
форма обучения очная

ТЕМА 1. Процесс разработки информационной системы. Методология разработки программного обеспечения. Гибкая методология разработки.

Справка

Унифицированный процесс - это широко используемый итеративный процесс разработки ОО систем.

Каждая итерация включает свои этапы анализа, проектирования, реализации и завершается созданием рабочей части системы.

Не рассматривайте итеративный проект в терминах последовательного жизненного цикла

ТЕМА 2. Описание предметной области системы.

Справка

Предметная область — это часть реального мира, рассматриваемая в рамках определённой деятельности.

ТЕМА 3. Определение функциональных требований к системе. Диаграмма прецедентов. Описание сценариев использования системы.

Лабораторная работа 1 Определение прецедентов

Справка

Прецеденты - это повествовательные истории об использовании системы, которые используются для формулировки требований.

Основная идея состоит в исследовании и формулировке функциональных требований путём написания историй “из жизни системы”.

Главный сценарий - В нём описывается типичная последовательность действий, приводящая к успешному завершению сценария и удовлетворяющая потребности всех заинтересованных лиц. Имена исполнителей принято начинать с заглавной буквы для облегчения их идентификации. Повторяющиеся действия выделяются курсивом.

Альтернативный сценарий - Здесь указываются все остальные сценарии или ветви, приводящие и к успешному и к неудачному завершению прецедента.

Задание

На основе описания предметной области, определить функциональные требования в виде диаграммы прецедентов, со спецификацией, спецификация должна содержать: описание, основного исполнителя, основной сценарии, альтернативные сценарии, при необходимости постусловия и предусловия.

Поскольку работа выполняется в командах, то каждый участник описывает свои прецеденты. Допускается что прецеденты, реализованные разными участниками, будут включать (include) или расширяться (extend) прецедентами других участников команды.

ТЕМА 4. Диаграмма концептуальных классов. Диаграмма объектов.

Лабораторная работа 2 Создание модели предметной области

Справка

Модель отображает основные (с точки зрения моделирующего) классы понятий (концептуальные классы) предметной области.

Модель предметной области — это визуальное представление концептуальных классов или объектов реального мира в терминах предметной области.

На языке UML модель предметной области представляется в виде набора диаграмм классов, на которых не определены никакие операции.

Задание

Описать модель предметной области в виде диаграммы классов.

Каждый участник команды создаёт диаграмму классов для своих прецедентов из предыдущей лабораторной работы.

Допускается что диаграммы разных участников будут содержать одни и те же классы.

Работу желательно выполнять совместно, чтобы классы отражающие одни и те же понятия реального мира были согласованы у всех участников команды.

ТЕМА 5. Системная диаграмма последовательностей

Лабораторная работа 3 Создание системных диаграмм последовательностей

Справка

Системные диаграмма последовательностей (sequence diagram) — это быстро и легко создаваемый артефакт, иллюстрирующий входные и выходные события, связанные с разрабатываемой системой.

Исходными данными для создания системных диаграмм последовательности служат описания прецедентов и системные события.

На системной диаграмме последовательностей для определённого хода событий (сценария), описанного в прецеденте, отображаются внешние исполнители, которые взаимодействуют непосредственно с системой, сама система (как «чёрный ящик»), а также системные события, инициируемые исполнителями.

Порядок событий должен соответствовать их последовательности в описании прецедента.

Время на диаграмме последовательности изменяется сверху вниз.

Задание

С помощью диаграммы последовательностей описать потоки сообщений между действующими лицами и системой.

Диаграмма последовательность создаётся для основного сценария, и для наиболее существенных альтернативных сценариев.

Каждый участник команды, создаёт диаграммы для своих сценариев, из первой лабораторной.

Выполнить описания операций.

ТЕМА 6. Логическая архитектура и принцип MVS

Справка

Логическая архитектура описывает систему в терминах её принципиальной организации в виде уровней, пакетов (пространств имён), программных классов и подсистем.

Она называется логической, поскольку не определяет способы развёртывания этих элементов в различных операционных системах или на физических компьютерах в сети (это относится к архитектуре развёртывания).

Уровень— это крупномасштабная группа классов, пакетов или подсистем, имеющих сходные обязанности для большинства аспектов системы. Уровни организуются в группы (например, уровень интерфейса пользователя).

- Интерфейс пользователя
- Уровень приложения или объектов предметной области - программные объекты, представляющие понятия предметной области и обеспечивающие выполнение требований к системе
- Уровень технических служб - объекты и подсистемы общего назначения, обеспечивающие поддержку взаимодействия с базой данных или журналов регистрации ошибок. Эти службы обычно независимы от приложения, и их можно повторно использовать в нескольких системах.

В жёстко структурированной многоуровневой архитектуре объекты каждого уровня могут вызывать службы только уровня, расположенного непосредственно под ним.

ТЕМА 7 Модель проектирования. Диаграммы взаимодействия (последовательностей и коммуникаций)

Справка

В состав языка UML входят обозначения для диаграмм взаимодействий (interaction diagrams), используемых для динамического моделирования объектов. Они иллюстрируют взаимодействия объектов с помощью сообщений.

Термин диаграмма взаимодействия используется в качестве общего названия для двух следующих конкретных типов диаграмм:

- Диаграммы последовательностей (sequence diagram)
- Диаграммы коммуникации (communication diagram)

Лабораторная работа 4. Создание диаграмм коммуникаций

Справка

Диаграммы коммуникации иллюстрируют взаимодействие объектов в формате графа или сети. При этом объекты могут размещаться в любом месте диаграммы.

Задание

С помощью диаграммы коммуникаций отобразить взаимодействие объектов в системе для выполнения системной функции.

Диаграмма коммуникаций создаётся для каждой системной функции всех диаграмм последовательностей, из предыдущей лабораторной.

Каждый участник команды, создаёт диаграммы для системных функций со своих диаграмм последовательностей. Однако допускается что некоторые системные функции будут совпадать, необходимо стремиться чтобы их реализация была одинаковой у всей команды.

ТЕМА 8. Модель проектирования. Диаграмма классов.

Лабораторная работа 5. Создание диаграмм классов проектирования

Справка

Диаграмма классов предназначены для статического моделирования объектов.

Диаграмма классов может быть получена из диаграммы взаимодействия. Это предполагает определённый порядок построения диаграмм: сначала создаются диаграммы взаимодействия, а затем — диаграммы классов. Однако на практике, особенно при использовании гибкого подхода к моделированию, эти взаимодополняющие динамические и статические диаграммы строятся параллельно.

Задание

Создать диаграмму классов проектирования.

Диаграмма классов не должна быть привязана к конкретному языку реализации.

Диаграмма классов должна быть согласована с диаграммами коммуникаций.

Если у разных участников есть одинаковые классы, - диаграммы должны быть согласованы.

ТЕМА 9. Модель проектирования. Распределения обязанностей на основе принципов GRASP.

Справка

Creator

Проблема:

Кто отвечает за создание нового экземпляра некоторого класса A?

Решение:

Назначить классу B обязанность создавать экземпляры класса A, если выполняется одно (или несколько) из следующих условий.

Класс B содержит или агрегирует объекты A.

Класс B записывает экземпляры объектов A.

Класс B активно использует объекты A.

Класс B обладает данными инициализации для объектов A.

Information Expert

Проблема:

Как распределить обязанности между объектами?

Решение:

Назначить обязанность тому классу, который обладает достаточной информацией для её выполнения.

Low Coupling

Проблема:

Как уменьшить влияние вносимых изменений на другие объекты?

Решение:

Минимизировать степень связывания в процессе распределения обязанностей.

Этот принцип используется для оценки различных альтернатив — при прочих равных условиях следует предпочитать проектное решение с более низкой степенью связывания.

Controller

Проблема:

Кто должен отвечать за получение и координацию выполнения системных операций, поступающих от уровня интерфейса пользователя?

Решение:

Присвоить эту обязанность классу, удовлетворяющему одному из следующих условий.

Класс представляет всю систему в целом, корневой объект, устройство или важную подсистему (внешний контроллер).

Класс представляет сценарий некоторого прецедента, в рамках которого выполняется обработка этой системной операции (контроллер прецедента или контроллер сеанса).

High Cohesion

Проблема:

Как обеспечить сфокусированность обязанностей объектов, их управляемость и ясность, а заодно выполнение принципа Low Coupling?

Решение:

Обеспечивать высокий уровень зацепления в процессе распределения обязанностей. Этот принцип нужно использовать для оценки различных альтернатив.

ТЕМА 10. Дополнительные шаблоны GRASP для распределения обязанностей

Справка

Polymorphism

Проблема:

Как обрабатывать альтернативные варианты поведения на основе типа? Как создавать подключаемые программные компоненты?

Решение:

Если поведение объектов одного типа (класса) может изменяться, обязанности распределяются для различных вариантов поведения с использованием полиморфных операций для этого класса.

Pure Fabrication

Проблема:

Объектно-ориентированные системы отличаются тем, что программные классы реализуют понятия предметной области.

Однако существует множество ситуаций, когда распределение обязанностей только между такими классами приводит к проблемам с зацеплением и связыванием, т.е. с невозможностью повторного использования кода.

Решение:

Присвоить группу обязанностей с высокой степенью зацепления искусственному классу, не представляющему конкретного понятия предметной области, т.е. синтезировать искусственную сущность для поддержки высокого зацепления, слабого связывания и повторного использования.

Indirection

Проблема:

Как распределить обязанности, чтобы обеспечить отсутствие прямого связывания; снизить уровень связывания объектов, согласно шаблону Low Coupling, и сохранить высокий потенциал повторного использования?

Решение:

Присвоить обязанности промежуточному объекту для обеспечения связи между другими компонентами или службами, которые не связаны между собой напрямую.

При таком подходе связи перенаправляются к другим компонентам или службам.

Protected Variations

Проблема:

Как спроектировать объекты, подсистемы и систему, чтобы изменение этих элементов не оказывало нежелательного влияния на другие элементы?

Решение:

Идентифицировать точки возможных вариаций или неустойчивости; распределить обязанности таким образом, чтобы обеспечить устойчивый интерфейс.

ТЕМА 11. Проектирование каркаса взаимодействия с базой данных на основе шаблонов. Объектно-реляционное отображение.

Лабораторная работа 6. Создание схемы базы данных.

Справка

Для хранения информации чаще всего применяются реляционные базы данных.

При этом возникает множество проблем, связанных с несоответствием объектно-ориентированного представления данных в системе и их представления в виде записей в базе данных.

Для работы с реляционными базами данных требуются специальные средства преобразования форм представления информации (O-R-преобразования).

Каркас интерфейса с базой данных — многократно используемый и обычно расширяемый набор классов, обеспечивающий обслуживание постоянно хранимых объектов.

Служба интерфейса с базой данных обычно разрабатывается для взаимодействия с реляционными базами данных. В этом случае она называется службой OR-преобразования. Служба интерфейса с базой данных выполняет преобразование объектной формы представления информации в форму записей (или другой структурированный формат данных, типа XML), а также обратные операции.

В терминах многоуровневой архитектуры приложения, служба интерфейса с базой данных — это подсистема уровня технических служб.

Задание

Создать схему базы данных. Допускаются нотации IDEF1X, воронья лапка, UML.

Если разные участники используют одни и те же таблицы, то схемы данных должны быть согласованы.

Допускается что на диаграммах одних участников команды, будут представлены таблицы, созданные другими участниками, например, в случае, если необходимо показать отношения с этой таблицей.

ТЕМА 12. Шаблоны проектирования Gang-of-Four

Справка

Большинство паттернов проектирования предназначены для получения расширяемости системы в определенной плоскости. Причем эта плоскость может бытьполезной для одного приложения и вредной — для другого. Наличие иерархии наследования может добавлять сложности простому приложению, но в случае библиотеки нередко делает решение чересчур сложным.

ТЕМА 13. Преобразование проектного решения в программный код. Критерии качества ПО.

Лабораторная работа 7. Реализация программного обеспечения

Справка

Написание кода на объектно-ориентированном языке программирования, не относится к процессу анализа или проектирования системы — это конечная цель проектирования.

Преимущество объектно-ориентированного подхода к анализу, проектированию и программированию в рамках UP состоит в том, что он обеспечивает полный цикл разработки системы — от формулировки требований до программной реализации. Артефакты последовательно трансформируются в артефакты следующей стадии разработки, постепенно обеспечивая превращение системы в работающее приложение. Не стоит ожидать, что этот процесс окажется гладким или механическим — он достаточно творческий и неоднозначный.

Однако предлагаемый подход обеспечивает отправную точку для экспериментирования и обсуждения.

Значительная часть усилий и творческого потенциала была задействована на стадии проектирования.

Тем не менее, процесс программирования не сводится к примитивной генерации кода. Совсем наоборот: результаты, полученные на стадии проектирования, оказываются далеко не совершенными. В процессе программирования и тестирования наверняка потребуется внести многочисленные изменения, а также выявить и разрешить возникшие проблемы.

Процесс преобразования объектно-ориентированных диаграмм классов в их определения и диаграмм взаимодействия в методы является относительно простым. При этом на стадии программирования разработчик по-прежнему имеет достаточную свободу действий, чтобы принимать дополнительные решения и изменять уже принятые.

Задание

Реализовать программу по выполненному проекту.

Программу необходимо выполнить на языке C#, изменить язык реализации можно только предварительно согласовав это с преподавателем, и только при согласии всех остальных членов команды.

Программа должна соответствовать модели проектирования.

СПИСОК ЛИТЕРАТУРЫ

Основная литература:

1. Сырецкий Г.А. Проектирование автоматизированных систем. Часть 1 [Электронный ресурс]: учебное пособие/ Сырецкий Г.А.— Электрон. текстовые данные.— Новосибирск: Новосибирский государственный технический университет, 2014.— 156 с. — Режим доступа: http://www.iprbookshop.ru/47714.html. — ЭБС «IPRbooks» (дата обращения: 26.04.2020)

Дополнительная литература:

1. Мейер Б., Основы программирования. [Электронные ресурс]: учебник / Б. Мейер – 2-е изд. – Москва: ИНТУИТ, 2016 – 422 с. - Режим доступа: https://e.lanbook.com/book/100317 — ЭБС «Лань» (дата обращения: 26.04.2020)
2. Суркова Н.Е. Проектирование информационных систем [Электронный ресурс]: методические указания к курсовому проекту/ Суркова Н.Е.— Электрон. текстовые данные. — Москва: Российский новый университет, 2010. — 60 с.— Режим доступа: http://www.iprbookshop.ru/21303.html — ЭБС «IPRbooks» (дата обращения: 26.04.2020)

Интернет-ресурсы:

1. Национальный открытый университет «ИНТУИТ» https://intuit.ru/