Лабораторная работа №4

по дисциплине: «Технология разработки программного обеспечения и оценка качества»
Тема: «Проектирование архитектуры программного обеспечения с использованием UML»

1. Цель работы:

Освоить основные диаграммы унифицированного языка моделирования (UML) для проектирования архитектуры программного обеспечения. Научиться трансформировать функциональные и процессные модели в конкретные проектные решения, определяющие структуру и поведение будущей системы.

2. Задачи:

- Изучить синтаксис и семантику ключевых диаграмм UML.
- Разработать модель прецедентов (Use Case) для определения границ взаимодействия с системой.
- Спроектировать диаграмму классов, отражающую статическую структуру данных системы.
- Построить диаграмму последовательности для моделирования динамического поведения системы.
- Создать диаграмму состояний для объектов со сложным жизненным циклом.

3. Теоретическая справка:

UML (Unified Modeling Language) — стандартизированный язык графического описания для объектного моделирования в области разработки ПО.

Ключевые диаграммы UML, используемые в работе:

1. **Диаграмма прецедентов (Use Case Diagram):**
    - Показывает взаимодействие между акторами (ролями) и системой.
    - Определяет границы системы и основные функциональные возможности.
2. **Диаграмма классов (Class Diagram):**
    - Описывает статическую структуру системы: классы, их атрибуты, методы и отношения между классами (ассоциация, наследование, агрегация, композиция).
3. **Диаграмма последовательности (Sequence Diagram):**
    - Моделирует взаимодействие объектов во времени.
    - Показывает последовательность сообщений между объектами для выполнения конкретного сценария.
4. **Диаграмма состояний (State Machine Diagram):**
    - Описывает поведение объекта, показывая последовательность состояний, которые объект проходит в ответ на события.

4. Задание:

На основе моделей, разработанных в предыдущих лабораторных работах (ТЗ по ГОСТ, IDEF0, IDEF3), необходимо разработать комплект UML-диаграмм для проектирования автоматизированной системы.

Конкретные требования:

4.1. Диаграмма прецедентов (Use Case Diagram)

- Выделить **не менее 3-4 основных акторов** системы (например, "Читатель", "Библиотекарь", "Администратор").
- Определить **не менее 8-10 прецедентов** использования системы.
- Показать отношения между прецедентами: include, extend (использовать хотя бы по одному разу каждого типа).
- Сгруппировать прецеденты по пакетам или акторам для наглядности.

4.2. Диаграмма классов (Class Diagram)

- Выделить **не менее 8-10 основных классов** предметной области.
- Для каждого класса определить:
    - **Атрибуты** (с указанием типов)
    - **Методы** (основные операции)
- Показать **различные типы отношений** между классами:
    - **Ассоциация** (как минимум 2 с указанием кратности)
    - **Агрегация/Композиция** (как минимум по 1 примеру)
    - **Наследование** (как минимум 1 пример иерархии)
- Использовать соответствующие модификаторы доступа (+, -, #).

4.3. Диаграмма последовательности (Sequence Diagram)

- Выбрать **один значимый сценарий** из диаграммы прецедентов.
- Построить диаграмму последовательности для этого сценария.
- Диаграмма должна содержать:
    - **Не менее 5-7 объектов/экземпляров классов**
    - **Не менее 10-15 сообщений** между объектами
    - Использовать **блоки альтернатив (alt)** для ветвления логики
    - Показать **циклы (loop)** при необходимости

4.4. Диаграмма состояний (State Machine Diagram)

- Выбрать **один класс с интересным жизненным циклом** (например, "Заказ", "Книга", "Абонемент").
- Построить диаграмму состояний, отражающую:

- **Не менее 4-5 состояний** объекта
- **Переходы между состояниями** с указанием событий/условий
- Начальное и конечное состояние

5. Связь с предыдущими работами:

Студент должен продемонстрировать преемственность моделей:

- Акторы на Use Case Diagram соответствуют ролям из IDEF0
- Прецеденты отражают функции из IDEF0 и IDEF3
- Классы и их атрибуты основаны на информационных объектах из ТЗ
- Сценарии на Sequence Diagram соответствуют процессам из IDEF3

6. Требования к оформлению и инструменты:

- **Инструменты:**
  - **Рекомендуется:** Draw.io, Lucidchart, Visual Paradigm, PlantUML, Enterprise Architect.
  - Может использоваться специализированное ПО (например, IDE с поддержкой UML) или онлайн-инструменты.
- **Состав отчета:**
  - Титульный лист
  - Введение с обоснованием выбора конкретных диаграмм
  - Все разработанные UML-диаграммы с пояснениями
  - Сравнительная таблица "Связь UML-моделей с предыдущими работами"
  - Выводы о преимуществах UML для проектирования архитектуры ПО