Лекция 1

Тема: Объектно-ориентированное проектирование ИС

Цель: Изучить основы объектно-ориентированного проектирования (ООП) информационных систем.

План:

1. Основные понятия ООП
2. Три типа моделей ООП
3. Унифицированный язык визуального моделирования UML

Литература:

1. Заботина, Н. Н. Методы и средства проектирования информационных систем: учебное пособие / Н.Н. Заботина. — Москва: ИНФРА-М, 2020. — С. 145-164. + Доп. материалы [Электронный ресурс]. — (Среднее профессиональное образование). - ISBN 978-5-16-104187-1. - Текст: электронный. - URL: https://znanium.com/catalog/product/1043093.
2. Вендеров А.М. Практикум по проектированию программного обеспечения экономических информационных систем: учеб. пособие / А.М. Вендеров. М., 2004. С. 47-154.
3. Гвоздева Т.В. Проектирование информационных систем / Т.В. Гвоздева, Б.А. Баллод. – Р-н-Д: Феникс, 2009. – С. 251-296.
4. Грекул В.И. Проектирование информационных систем: учеб. пособие / В.И. Грекул, Г.Н. Денищенко, Н.Л. Коровкина. М., 2008. С. 196 - 209
5. Мацяшек Л. А. Анализ требований и проектирование систем. Разработка информационных систем с использованием UML / Л. А. Мацяшек. – М.: Вильямс, 2002. – 432 с.
6. Осипов Д. Delphi XE2 / Д. Осипов. - СПб.: БХВ-Петербург, 2012. – 912 с.
7. Рамбо Дж. UML 2.0. Объектно-ориентированное моделирование и разрасботка / Дж. Рамбо, М. Блаха. СПб.: Питер, 2007.
8. Смирнова Г.Н. Проектирование экономических информационных систем: учебник / Г.Н. Смирнова, А.А. Сорокин, Ю.Ф. Тельнов. М., 2003. С. 351 – 373.
9. Соловьев И.В. Проектирование информационных систем. Фундаментальный курс: Учеб. пособие для высшей школы / И.В. Соловьев, А.А. Майоров. - М.: Академический проект, 2009. – С. 305-322.
10. Хорев П.Б. Технологии объектно-ориентированного программирования: учеб. пособие для студ. высш. учеб. заведений/П.Б. Хорев.-2-е изд.-М.: ИЦ "Академия", 2008.-448с.

1. Основные понятия

Объектно-ориентированное проектирование – подход к решению задач с использованием моделей, основанных на понятиях реального мира. Фундаментальным элементом является ***объект***, объединяющий структуру данных с поведением.

Объектная ориентированность в простейшем случае означает представление программного обеспечении в идее дискретных объектов, содержащих в себе структуры данных и поведение. Характеристики объектно-ориентированного подхода: индивидуальность, классификация, наследование и полиморфизм.

Индивидуальность означает, что данные делятся на дискретные сущности, хорошо отличимые друг от друга. Например, мой компьютер, белый ферзь - объекты.

Классификация означает, что объекты с одинаковыми структурами данных (атрибутами) и поведением (операциями) группируются в классы. Например, монитор и шахматная фигура – это классы. Класс – это абстракция, описывающая свойства, важные для конкретного приложения и игнорирующая все остальное.

Наследование – это наличие у разных классов, образующих иерархию, общих атрибутов и операций. ***Суперклас***с задает наиболее общую информацию, которую затем уточняют его ***подклассы***. Каждый подкласс соединяет в себе, т.е. наследует все черты его суперкласса, к которым добавляет собственные уникальные черты. Подклассом не обязательно воспроизводить все черты суперкласса.

Полиморфизм означает, что одна и та же операция может подразумевать разное поведение в разных классах. ***Операция*** – это процедура или трансформация, которую объект выполняет сам или которая осуществляется с данным объектом. Реализация операции в конкретном классе называется ***методом***.

Объектно-ориентированные концепции:

1. ***Абстракция*** означает сосредоточение на важнейших аспектах приложения и игнорирование всех остальных. Сначала принимается решение о том, что представляет собой объект и что он делает, а уже затем подбирается способ его реализации. Использование абстракций позволяет сохранить свободу принятия решений как можно дольше благодаря тому, что детали не фиксируются раньше времени.

2. ***Инкапсуляция***, или, иначе говоря, сокрытие информации, состоит в отделении внешних аспектов объектов, доступных другим объектам, от деталей внутренней реализации, которые от других объектов скрываются. Инкапсуляция исключает возникновение взаимозависимостей участков программы, из-за которых небольшие изменения приводят к значительным непредвиденным последствиям.

3. ***Объединение данных и поведения***. При вызове операций не надо беспокоиться о том, сколько реализаций этой операции существует в системе. Полиморфизм операторов перекладывает ответственность за выбор подходящей реализации с вызывающего кода на иерархию классов.

4. ***Совместное использование***. ОО технологии способствуют совместному использованию сущностей на самых разных уровнях. Наследование структур данных вместе с поведением дает возможность подклассам совместно использовать общий код. Проектировщики должны планировать на будущее, видеть дальше границ текущего приложения и вкладывать дополнительные усилия в получение более универсальных конструкций.

5. ***Выделение сущности объекта***. Объектно-ориентированная технология выделяет то, чем объект является, а не то как он используется. Использование объекта зависит от особенностей приложения и часто изменяется в процессе разработки. По мере уточнения требований, черты объекта остаются более стабильными, чем детали его использования, поэтому системы, основанные на объектной структуре, в конечном счете оказываются более стабильными.

2. Три типа моделей

Майкл Блаха и Джеймс Рамбо [7] предлагают для описания системы различных точек зрения использовать три типа моделей. Модель классов описывает объекты, входящие в состав системы и отношения между ними. Модель состояний описывает историю жизни объектов. Модель взаимодействий описывает взаимодействия между объектами. Каждая модель применяется на всех этапах проектирования и постепенно обрастает деталями. Полное описание системы требует наличие всех трех моделей.

Модель классов описывает статическую структуру объектов системы и их отношения. Эта модель определяет контекст разработки программы, т.е. предметную область. Модель классов изображается на диаграммах классов. ***Диаграмма классов*** – это граф, вершинами которого являются классы, а ребрами – их отношения.

Модель состояний описывает изменяющиеся со временем аспекты объектов. Эта модель реализуется посредством диаграмм состояний. ***Диаграмма состояний*** – это граф, вершинами которого являются состояния, а ребрами переходы между состояниями, инициируемые событиями.

Модель взаимодействия описывает кооперацию объектов системы для достижения лучших результатов. Построение модели начинается с ***вариантов использования***, которые затем уточняются на ***диаграммах последовательностей*** и ***диаграммах деятельности***.

Вариант использования описывает функциональность системы, то есть то, что система делает для пользователей.

Диаграмма последовательности изображает взаимодействие объектов и временную последовательность этого взаимодействия.

Диаграмма деятельности уточняет важные этапы обработки.

3. Унифицированный язык визуального моделирования UML

Толчком к разработке ООП явилось распространение ОО языков программирования в конце 1980-х – начале 1990-х годов. Пользователям хотелось получить единый язык моделирования, который объединил бы в себе мощь объектно-ориентированного подхода

и давал бы четкую модель системы, отражающую все её значимые стороны. К середине 90-х явными лидерами в этой области стали методы Booch (Grady Booch), OMT-2 (Jim Rumbaugh), OOSE – Object-Oriented Software Engineering (Ivar Jacobson). Однако эти три метода имели свои сильные и слабые стороны: OOSE был лучшим на стадии анализа проблемной области и анализа требований к системе, OMT-2 был наиболее предпочтителен на стадиях анализа и разработки информационных систем, Booch лучше всего подходил для стадий дизайна и разработки.

Все шло к созданию единого языка, который объединял бы сильные стороны известных методов и обеспечивал наилучшую поддержку моделирования. Таким языком оказался UML. Создание UML началось в октябре 1994 г., когда Джим Рамбо и Гради Буч из Rational Software Corporation стали работать над объединением своих методов OMT-2 и Booch. Осенью 1995 г. увидела свет первая черновая версия объединенной методологии, которую назвали Unified Method 0.8. После присоединения в конце 1995 г. к Rational Software Corporation Айвара Якобсона и его фирмы Objectory, усилия трех создателей наиболее распространенных объектно-ориентированных методологий были объединены и направлены на создание UML. Система обозначений UML оказалась настолько удачной, что вытеснила практически все другие системы. В 2001 г. члены OMG начали работу над новой версией UML, добавляя в неё недостающие элементы и устраняя недостатки, выявленные в UML1. В 2004 г. была принята новая версия UML 2.0.

UML представляет собой объектно-ориентированный язык моделирования, обладающий следующими основными характеристиками:

- ***является языком визуального моделирования, который обеспечивает разработку репрезентативных моделей для организации взаимодействия заказчика и разработчика ИС, различных групп разработчиков ИС;***
- ***содержит механизмы расширения и специализации базовых концепций языка.***

UML включает внутренний набор средств моделирования. Пользователям языка предоставлены возможности:

- строить модели на основе средств ядра, без использования механизмов расширения для большинства типовых приложений;
- добавлять при необходимости новые элементы и условные обозначения, если они не входят в ядро, или специализировать компоненты, системе условных обозначений (нотацию) и ограничения для конкретных предметных областей.

Вопросы для самопроверки:

1. В чем сущность объектно-ориентированного моделирования и проектирования ИС?

2. Каковы основные характеристики объектно-ориентированного подхода?
3. Какие типы моделей используются при объектно-ориентированном проектировании ИС?
4. Каково назначение модели классов? модели состояний? модели взаимодействий?
5. Что такое UML?
6. Какие возможности дает использование UML?