

УДК 378.147:004.75

Сейдаметова Зарема Сейдалиевна
доктор педагогических наук, профессор
Крымский инженерно-педагогический университет, г. Симферополь

Абдураманов Зиннур Шевкетович
старший преподаватель
Крымский инженерно-педагогический университет, г. Симферополь

Асанова Усние Бахтияровна
аспирант
Крымский инженерно-педагогический университет, г. Симферополь

ПЕДАГОГИЧЕСКИЕ ОСОБЕННОСТИ ФОРМИРОВАНИЯ ОБЪЕКТНОГО МЫШЛЕНИЯ И СТИЛЯ ПРОГРАММИРОВАНИЯ

В статье рассматриваются вопросы обучения методологии объектно-ориентированного проектирования. Выделены два вида проблем, с которыми приходится сталкиваться в процессе преподавания объектно-ориентированного программирования (понимание базовых понятий, выбор языка программирования).

Представлена возможная разбивка учебного материала, связанного с изучением методологий объектно-ориентированной разработки, на модули. Первый модуль содержит тематику, связанную с введением в объектно-ориентированный подход. Второй – предполагает знакомство с универсальным языком моделирования UML. Третий – изучение объектно-ориентированных систем анализа и проектирования. Четвертый модуль ориентирован на студентов, владеющих одним из языков программирования, и включает вопросы объектно-ориентированной реализации.

Ключевые слова: объектно-ориентированное программирование, объектно-ориентированная парадигма, ОО-концепции, языки программирования, компоненты методологии объектно-ориентированной разработки.

Seidametova Zarema

Doctor of science (in pedagogy)

Professor, Crimean Engineering and Pedagogical University, Simferopol

Abduramanov Zinnur

Lecturer, Crimean Engineering and Pedagogical University, Simferopol

Asanova Usnie

Teaching assistant, Crimean Engineering and Pedagogical University, Simferopol

PEDAGOGICAL ASPECTS OF OBJECT-ORIENTED THOUGHT AND PROGRAMMING PROCESS DEVELOPING

We discuss the methodology of teaching object-oriented design and programming. We highlight two types of problems encountered in the teaching and learning of the object-oriented programming (understanding of basic concepts, choice of the programming language). We show a possible split of the educational material, connected with the study of methodologies for object-oriented development, on modules. The first module includes the introduction of object-oriented design and programming. Second module helps to learn universal modeling language UML. The third one shows how to study object-oriented systems analysis and design. The fourth module, created for students familiar with one of the programming languages, helps to learn object-oriented design and implementation.

Keywords: object-oriented programming, object-oriented paradigm, object-oriented concepts, programming languages, components of object-oriented development methodology.

В последние десятилетия в разработке программных приложений наиболее используемой является технология объектно-ориентированного программирования (ООП). В классической книге Энтони Элиенса [1] представлено описание преимуществ и недостатков использования объектно-ориентированной парадигмы (ОО-парадигмы) в программной инженерии. Кроме того, в [1] последовательно рассмотрено использование ОО-парадигмы на каждой стадии жизненного цикла разработки программного продукта: от

анализа требований до сопровождения и обновления версий продукта. В [1], [2], [3] на примерах показано, как можно увязать соответствующие принципы ООП с разработкой продукта с помощью различных языков программирования, например, Java и C++. Также в [1], [2], [3] обсуждаются особенности реализации объектно-ориентированного подхода в объектно-ориентированных языках Smalltalk, Eiffel, C++, Java, а также языке проектирования UML и ОО-технологии CORBA. Энтони Элиенс в [2], [3] предлагает следующую формулу ОО-подхода:

$$\text{ОО-подход} = \text{Объекты} + \text{Наследование} + \text{Пересылка сообщения}$$

Дэвид Вест в книге [4] описывает объектное мышление, необходимое разработчику программного приложения, и которое, как считается, присуще разработчикам из компании Microsoft. Дэвид Вест полагает, что именно объектное мышление делает программиста хорошим профессионалом, а не методы и инструментарий. На примерах он показывает, что лучшие программисты опираются на анализ и концептуальность в мышлении больше, чем на формальные процессы и методы.

В монографии [5] рассматриваются вопросы подготовки инженеров-программистов; представлен базисный корпус знаний ВОК [5, 31-40], который необходимо учитывать при составлении соответствующих программ подготовки. По нашему мнению, одной из важнейших составляющих подготовки инженеров-программистов является формирование у студентов объектного мышления, объектного стиля программирования, а также формирование навыков применения методологий объектно-ориентированной разработки.

В современном программировании широко используется объектно-ориентированный подход, который позволяет повышать качество программ, производительность работы программиста, эффективность командной работы. Кроме того, профессиональные среды программирования содержат средства для поддержки объектно-ориентированного программирования. Поэтому при подготовке программистов в университете важно включение в учебные планы

дисциплин, связанных с объектно-ориентированным программированием и проектированием. Однако в процессе преподавания объектно-ориентированного программирования (ООП) приходится сталкиваться с некоторыми трудностями, перечисленными ниже:

1. Понимание базовых понятий.

У студентов вызывают затруднения понимание базовых понятий ООП, таких как класс, объект, интерфейс, абстракция, инкапсуляция, наследование и полиморфизм.

Для решения этой проблемы в классической монографии [6], названной «Процесс объектно-ориентированной мысли», представлены, в понятной для студентов форме, основы объектно-ориентированных концепций и объяснено, как можно использовать различные объектные технологии в практическом программировании. Автор монографии [6] М. Вейсфелд знакомит читателя с объектно-ориентированными концепциями, абстракциями, классами (public и private), повторными кодами и средами разработки. Учитывая современное состояние и развитие информационных технологий в [6] уделено большое внимание вопросам, связанным с построением объектов, работающих с XML, базами данных и распределенными системами (включая EJBs, .NET, Web-сервисы и т.п.). Для иллюстраций и построения соответствующих диаграмм используется UML, стандартный язык моделирования объектов, представлены иллюстрации и примеры каждой концепции.

В статье [7] авторы полагают, что для формирования понимания, как выполняется объектно-ориентированная программа, студентам в качестве заданий можно предложить нарисовать диаграмму состояния программы в определенный момент времени. Преподаватель проводит инструктаж, а также дает студентам минимальные указания относительно того, что должны содержать их диаграммы, как на них должны быть изображены центральные концепции и взаимосвязи, появляющиеся при выполнении программы. Такой подход позволяет студентам легко осваивать ОО-понятия и формирует понимание, как выполняются программы.

2. Выбор языка программирования.

Сложно выбрать язык программирования, наиболее подходящий для преподавания ООП. Чаще всего используется один из следующих языков программирования – C++, C#, Java, Object Pascal, Python, Objective-C. Возможно, в некоторых случаях разумнее использовать псевдокод. В статье [8] приведен рейтинг ТЮВЕ языков программирования, а также даны рекомендации о том, какие из них и почему можно использовать в учебном процессе при подготовке инженеров-программистов.

При выборе языка программирования для преподавания ООП необходимо учитывать кроссплатформенность, наличие множества свободно распространяемых, а также коммерческих сред разработки, возможность решения широкого круга задач. Например, Java используется для создания настольных, серверных, мобильных приложений, JavaScript – при разработке приложения в web-дизайне. Язык программирования Java во всех этих случаях один, различие в том, что используются разные библиотеки классов. Отметим, что в каждом языке программирования имеются свои особенности и возможности, которые не являются прихотью разработчиков, а являются осознанной необходимостью, позволяющей решать те или иные задачи проще и эффективнее. Например, в C++ при работе с объектами имеется функционал, которого нет в других языках программирования.



Рис. 1.

Компоненты методологии объектно-ориентированной разработки

Для преодоления трудностей и проблем в обучении студентов объектно-ориентированному проектированию и программированию мы предлагаем разбить учебный материал, связанный с изучением методологий объектно-ориентированной разработки, на четыре составляющие (см. рис. 1):

- **Модуль 1** – Введение в объектно-ориентированный подход. Целями модуля являются ознакомление студентов с основами объектно-ориентированной парадигмы; с различными объектно-ориентированными языками программирования; с типичным жизненным циклом объектно-ориентированной разработки; а также разъяснение различий между этими языками; формирование у студентов понимания различий между парадигмами программирования; понимания преимущества объектно-ориентированной парадигмы.

Модуль 1 – Введение в объектно-ориентированный подход
Цели: <ul style="list-style-type: none"> – Знакомство с основами объектно-ориентированной парадигмы; – Понимание разницы между различными парадигмами программирования; – Понимание преимущества объектно-ориентированной парадигмы;

<ul style="list-style-type: none"> – Знакомство с различными объектно-ориентированными языками программирования, а также разъяснение различия между этими языками; – Знакомство с типичным жизненным циклом объектно-ориентированной разработки. 	
<p>Содержание:</p> <ul style="list-style-type: none"> – Базовая философия объектно-ориентированной парадигмы; – Основные компоненты объектно-ориентированного моделирования; – Обзор основных концепций объектно-ориентированного программирования; – Объектно-ориентированные языки программирования; – Диаграммы классов; – Моделирование взаимосвязей; – Известные классы взаимосвязей; – Абстракция; – Инкапсуляция / сокрытие информации; – Наследование; – Полиморфизм; – Обзор класса идентификации и спецификации; <p style="text-align: center;">Модель объектно-ориентированного процесса.</p>	<p>Результаты:</p> <p>После окончания модуля 1 студенты должны будут</p> <ul style="list-style-type: none"> – Определять ключевые концепции объектно-ориентированной парадигмы; – Различать объектно-ориентированную и другие парадигмы программирования; – Понимать преимущества и ограничения объектно-ориентированной парадигмы; – Анализировать проблему и выявлять ее объекты и их взаимоотношения; – Уметь объяснять типичный процесс объектно-ориентированной разработки; – Сравнить и сопоставлять различные объектно-ориентированные языки программирования.

- **Модуль 2** – Универсальный язык моделирования UML. Целью модуля является знакомство с концепциями моделирования программного обеспечения; с универсальным языком моделирования UML; различными артефактами UML, условными обозначениями (нотациями), семантикой и типичным использованием; а также понимание выгод и ограничений UML.

Модуль 2 – Универсальный язык моделирования UML
<p>Цели:</p> <ul style="list-style-type: none"> – Знакомство с концепциями моделирования программного обеспечения; – Знакомство с универсальным языком моделирования UML; – Понимание выгод и ограничений UML;

– Знакомство с различными артефактами UML, условными обозначениями (нотациями), семантикой и типичным использованием.	
<p>Содержание:</p> <ul style="list-style-type: none"> – Обзор основных объектно-ориентированных концепций; – Концепции модели, моделирования и языка моделирования; – Введение в универсальный язык моделирования UML; – UML диаграммы; – Диаграмма классов; – Объектная диаграмма; – Диаграмма вариантов использования; – Диаграмма последовательности; – Диаграмма взаимодействия; – Диаграмма состояния; – Диаграмма деятельности; – Диаграмма компонентов; – Диаграмма развертывания; – UML отношения; – Моделирование динамических представлений. 	<p>Результаты:</p> <p>После окончания модуля 2 студенты должны будут</p> <ul style="list-style-type: none"> – Разрабатывать модели с использованием UML; – Разрабатывать диаграммы классов, модели объектов и диаграммы последовательности для небольших и средних проблем разработки программного обеспечения; – Понимать отличия между различными моделями UML, а также понимать в каких случаях эти модели необходимо использовать при разработке программных продуктов; – Понимать отличия статистических и динамических представлений программного обеспечения.

- **Модуль 3** – Объектно-ориентированные системы анализа и проектирования. Цель – формирование понимания необходимости получения максимальных выгод из объектной технологии; необходимости, места и целей систем анализа и проектирования; способов использования универсального языка моделирования UML в анализе и проектировании; моделей и формализмов, которые должны быть развернуты; моделей, необходимых для проектирования и представления процесса. Кроме того, в рамках этого модуля необходимо научить студентов применять объектно-ориентированные методы для определения спецификаций, проектирования, реализации проекта.

Модуль 3 – Объектно-ориентированные системы анализа и проектирования
Цели:

<ul style="list-style-type: none"> – Понимание необходимости получения максимальных выгод из объектной технологии; – Понимание необходимости, места и целей систем анализа и проектирования; – Понимание способов использования универсального языка моделирования UML в анализе и проектировании; – Понимание моделей и формализмов, которые должны быть развернуты; – Понимание какими моделями процесс проектируется и представляется; – Научить применять объектно-ориентированные методы должным образом для производства спецификаций и проектов. 	
<p>Содержание:</p> <ul style="list-style-type: none"> – Введение – Объектно-ориентированный процесс и проект – Сопоставление объектно-ориентированного анализа и объектно-ориентированного проектирования; – Требования инжиниринга; – Требования выявления и анализа; – Анализ модели с использованием UML; – Моделирование вариантов использования – Система диаграммы последовательности (SSD) – Доменные (концептуальные) модели – Добавление ассоциаций – Добавление атрибутов – Операция контрактов – Примеры моделей анализа и моделирования анализа. – Пример Интернет-покупок. – Моделирование деятельности – Моделирование диаграмм состояния – Архитектура программного обеспечения – Объектно-ориентированное проектирование – Обзор 	<p>Результаты:</p> <p>После окончания модуля 3 студенты должны будут</p> <ul style="list-style-type: none"> – Определять место и цели моделирования и проектирования проекта; – Применять объектно-ориентированные методы для определения спецификаций, проектирования, реализации проекта; – Анализировать модели с использованием UML; – Проводить анализ требований, выявлять риски; – Моделировать диаграммы вариантов использования для анализа требования, диаграммы деятельности, состояния; – Выявлять архитектуру программного продукта; – Использовать шаблонное проектирование; – Создавать диаграммы взаимодействия, диаграммы классов.

<ul style="list-style-type: none"> – Диаграмма взаимодействия – RDD (Responsibility-driven design) с шаблонами GRASP – Создание диаграммы классов проектирования. 	
--	--

- **Модуль 4** – Объектно-ориентированная реализация. Модуль рассчитан на студентов, которые знакомы, по крайней мере, с одним из объектно-ориентированных языков программирования. В модуле используется С++ или Java, поскольку это наиболее популярные среди программистов языки. Главная цель модуля заключается в демонстрации того, каким образом объектно-ориентированные модели превращаются в конструкции С++ или Java с использованием ОО-парадигмы.

Модуль 4 – Объектно-ориентированная реализация	
<p>Данный курс представляет методики и способы преобразования объектно-ориентированного проектирования артефактов в программный код. Модуль рассчитан на студентов, которые знакомы, по крайней мере, с одним из объектно-ориентированных языков программирования, а также желающих приобрести опыт кодирования для объектно-ориентированного проекта. В модуле используется С++ или Java, поскольку это наиболее популярные среди программистов языки. В модуле будет продемонстрировано, каким образом объектно-ориентированные модели превращаются в конструкции С++ с использованием ОО парадигмы. Использование С++ или Java не подразумевает специального разрешения.</p> <p>С #, Smalltalk, Python и многие другие объектно-ориентированные языки допускают принципы объектного проектирования и отображения представленного в этом модуле кода.</p> <p>Цели:</p> <ul style="list-style-type: none"> – Ознакомить с методиками и способами преобразования объектно-ориентированного проектирования артефактов в программный код; – Программная реализация объектно-ориентированного проекта. 	
<p>Содержание:</p> <ul style="list-style-type: none"> – Введение – Реализация классов – Реализация статического поведения – Реализация динамического поведения 	<p>Результаты:</p> <p>После изучения этого модуля студент сможет:</p> <ul style="list-style-type: none"> – Выявлять требования реализации в языке объектно-ориентированного программирования в целом.

<ul style="list-style-type: none"> – Создание экземпляров и удаление объектов – Наследование – Реализация агрегации – Реализация нескольких связей – Примеры – Основы тестирования программного обеспечения. 	<ul style="list-style-type: none"> – Выявлять информацию в UML моделях проектирования, необходимых для реализации. – Реализовывать основные строительные блоки класса. – Реализовывать спецификации поведения, разработанные в ходе объектно-ориентированного проектирования. – Реализовать динамические характеристики, разработанные в ходе объектно-ориентированного проектирования. – Реализовывать обобщения/специализации, агрегации и другие сложные отношения. – Выполнять отображение процесса объектно-ориентированного проектирования, разработанного с использованием UML для реализации через практические примеры (case study), который соединяют все части в одно целое.
--	---

Предложенная выше разбивка на модули не обязательно предполагает изучение ОО-методологий в пределах одной учебной дисциплины. В рамках концепции «Базисного корпуса знаний» возможно включение разных модулей в учебные программы разных дисциплин, например, «Программирование», «Программирование для начинающих», «Объектно-ориентированное программирование», «Объектно-ориентированный анализ и проектирование», «Технологии проектирования». Принципиально – сохранение последовательности изучения модулей. Освоение каждого модуля является пререквизитом допуска к изучению последующих модулей.

Литература

1. Элиенс А. Принципы объектно-ориентированной разработки программ. 2-е издание. – М.: Вильямс, 2002. – 496 с.

2. Eliens A. Principles of Object-Oriented Software Development. 2nd edition [Электронный ресурс] – Режим доступа: <http://www.cs.vu.nl/~eliens/online/oo/>
3. Eliens A. Principles of Object-Oriented Software Development. New version [Электронный ресурс] – Режим доступа: <http://www.cs.vu.nl/~eliens/poosd/index.html>
4. West D. Microsoft Object Thinking – Redmond (Washington): Microsoft Press, 2004. – 368 p.
5. Сейдаметова З.С. Подготовка инженеров-программистов по специальности «Информатика»: [монография] / Зарема Сейдалиевна Сейдаметова. – Симферополь: Крымучпедгиз, 2007. – 480, [1] с.
6. Weisfeld M. Object-Oriented Thought Process / Matt Weisfeld. – Sams Publishing, 2003. – 304 p. – ISBN 0-672-32611-6.
7. Sajaniemi J. A Study of the Development of Students' Visualizations of Program State during an Elementary Object-Oriented Programming Course / Jorma Sajaniemi, Marja Kuittinen, Taina Tikansalo // The Third International Computing Education Research Workshop (ICER'07). – September 15–16, 2007. – Atlanta, Georgia, USA – p. 1–15.
8. Манжос Л.О. Мови програмування в навчанні майбутніх програмістів / Л.О. Манжос, З.С. Сейдаметова // Науковий часопис НПУ ім. М.П.Драгоманова. Серія №2. Комп'ютерно-орієнтовані системи навчання. Зб. наукових праць / Редрада. – К.: НПУ ім. Драгоманова, 2010. – № 8 (15). – С. 35-41.

References

9. Jeliens A. Principy ob#ektno-orientirovannoj razrabotki programm. 2-e izdanie. – М.: Vil'jams, 2002. – 496 s. (*In Russian*)
10. Eliens A. Principles of Object-Oriented Software Development. 2nd edition [Jelektronnyj resurs] – Rezhim dostupa: <http://www.cs.vu.nl/~eliens/online/oo/>
11. Eliens A. Principles of Object-Oriented Software Development. New version [Jelektronnyj resurs] – Rezhim dostupa: <http://www.cs.vu.nl/~eliens/poosd/index.html>

12. West D. Microsoft Object Thinking – Redmond (Washington): Microsoft Press, 2004. – 368 p.
13. Sejdametova Z.S. Podgotovka inzhenerov-programmistov po special'nosti «Informatika» : [monografija] / Zarema Sejdaliyeva Sejdametova. – Simferopol': Krymchpedgiz, 2007. – 480, [1] s. (*In Russian*)
14. Weisfeld M. Object-Oriented Thought Process / Matt Weisfeld. – Sams Publishing, 2003. – 304 p. – ISBN 0-672-32611-6.
15. Sajaniemi J. A Study of the Development of Students' Visualizations of Program State during an Elementary Object-Oriented Programming Course / Jorma Sajaniemi, Marja Kuittinen, Taina Tikansalo // The Third International Computing Education Research Workshop (ICER'07). – September 15–16, 2007. – Atlanta, Georgia, USA – p. 1–15.
16. Manzhos L.O. Movi programuvannja v navchanni majbutnih programistiv / L.O. Manzhos, Z.S. Sejdametova // Naukovij chasopis NPU im. M.P.Dragomanova. Serija №2. Komp'juterno-orientovani sistemi navchannja. Zb. naukovih prac' / Redrada. – K.: NPU im. Dragomanova, 2010. – № 8 (15). – S. 35-41. (*In Ukrainian*)