ВОПРОСЫ МОДЕЛИРОВАНИЯ ПРОГРАММНЫХ СРЕДСТВ В КУРСЕ «ПРОГРАММИРОВАНИЕ»

Л. Ф. Дробушевич, В. В. Конах

Белорусский государственный университет
Минск, Беларусь
e-mail: droblf@bsu.by; konakh@bsu.by

Рассматриваются вопросы использования средств моделирования при обучении программированию на младших курсах.

Ключевые слова: программирование; моделирование; интегрированная среда разработки; UML; Microsoft Visual Studio; IntelliJ IDEA; Eclipse.

QUESTIONS OF MODELING SOFTWARE IN THE COURSE «PROGRAMMING»

L. F. Drobushevich, V. V. Konakh

Belarusian State University
Minsk, Belarus

The problems of the use of modeling tools for teaching programming in junior courses.

Keywords: programming; modeling; integrated development environment; UML; Microsoft Visual Studio; IntelliJ IDEA; Eclipse.

О ценности языка UML (унифицированный язык моделирования) и об особенностях его использования в программных проектах сегодня написано достаточно много [1, 2]. Разработчикам программных систем (ПС) в современных условиях не чужды задачи проектирования, моделирования и понимания чужих программных продуктов и процессов, что очень важно. С помощью элементов и категорий UML можно влиять на процесс разработки, принимать решения и решать те или иные задачи. Уровень понимания UML и умение его использовать совместно с языками программирования позволяют повысить качество разрабатываемого продукта, что является основной целью современных ИСО-стандартов.

В рамках курса «Программирование» не так просто найти место для изучения UML и его использования для моделирования учебных программ из-за ограниченного количества учебных часов и большого объема изучаемого на 1−2 курсах материала. Однако использование UML позволит упростить создание проектов с помощью объектно-ориентированных языков программирования. Следовательно, можно дать основы UML на этапе изучения вопросов проектирования алгоритмов. Создание любого проекта начинается с обдумывания структуры будущего приложения и его поведения. Почему бы не использовать для этого UML? К созданной диаграмме всегда можно вернуться и без особого труда разобраться, что же делает программа, чего не скажешь

про исходный код спустя время. Более того, обучать университетских студентов программированию невозможно без учета задач и целей всех, кто вовлечен в процесс разработки программного обеспечения (ПО): заказчики, аналитики, архитекторы, дизайнеры, программисты и тестировщики.

Одна из сложных задач – выбрать уровень изложения UML и способов моделирования в рамках курса «Программирование». Целесообразно на первое место при решении этой непростой задачи поставить краткость и понятность изложения. Трудные и сомнительные места нотации UML на младших курсах вполне можно обойти стороной. Это позволит всем, кто «с нуля» изучает UML, грамотное, осмысленное и, главное, эффективное вхождение в его нотацию.

Одним из назначений UML является создание таких моделей, для которых возможна автоматическая генерация программного кода приложения. Более того, природа моделей такова, что возможен и обратный процесс: автоматическое получение модели по исходному коду приложения. При изучении языков программирования можно выделить два варианта использования UML:

– Вариант рисования (визуализации) диаграмм подразумевает использование диаграмм UML с целью обдумывания, обмена информацией между людьми, документирования и т. п. В этом варианте использования нотации поддерживающий инструмент не очень важен и нужен − простую диаграмму можно нарисовать и от руки.

– Вариант разработки приложений подразумевает детальное моделирование, реализацию и тестирование приложения в терминах UML. В этом варианте основной результат – работающее приложение, которое может быть скомпилировано в язык, поддерживаемый определенной системой программирования.

Сегодня процесс разработки ПО осуществляется в интегрированных средах разработки, которые используются также и в процессе обучения программированию. Рассмотрим краткий обзор возможностей моделирования ПО с помощью UML в трех широко используемых современных интегрированных средах разработки: Microsoft Visual Studio, IntelliJ IDEA и Eclipse [3].

Microsoft Visual Studio – это интегрированная среда разработки, которая содержит многочисленный набор инструментов для создания ПО: от планирования до разработки пользовательского интерфейса, написания кода, тестирования, отладки, анализа качества кода и производительности, развертывания в средах клиентов, которые предназначены для максимально эффективной совместной работы [3]. Visual Studio обеспечивает поддержку C#, C и C++, JavaScript, F# и VisualBasic. Поэтому такой мощный инструмент разработки программного обеспечения немыслим без средств моделирования ПО.

Для создания моделей необходима версия Visual Studio Ultimate, так как только она содержит в своем меню пункт «Архитектура». В этой версии, например, для описания и передачи пользовательских требований можно использовать диаграммы вариантов использования, деятельности, классов и последовательностей. Для описания функциональных возможностей можно использовать диаграммы компонентов, классов, деятельности и последовательностей.

Для этих диаграмм поддерживаются все отношения между элементами согласно стандарту UML 2. Код в этой версии среды генерируется на основе диаграмм классов. Получить модель в виде диаграммы классов возможно по коду программы.

В Visual Studio Ultimate диаграмма последовательностей показывает взаимодействие, которое представляет последовательность сообщений между экземплярами

классов, компонентами и подсистемами. Существует два способа получения диаграмм последовательностей:

– Диаграммы последовательностей образуют часть модели и существуют только в пределах UML-проекта.

– Диаграммы последовательностей, основанные на реализации методов в коде, могут быть созданы из кода программы на C# и помещены в любую модель проекта.

IntelliJ IDEA – интегрированная среда разработки ПО на многих языках программирования, в частности Java, JavaScript, Python [4]. Начиная с версии 9.0, IntelliJ IDEA доступна в двух версиях: Community Edition и Ultimate Edition. В версии Ultimate Edition реализована поддержка Java EE, UML-диаграмм, подсчет покрытия кода, а также поддержка других систем управления версиями, языков и фреймворков.

Поддержка UML-моделирования в среде IntelliJ IDEA включает следующие аспекты:

– Reverseengineering (обратная разработка) – подразумевает создание модели в виде диаграммы классов на базе существующего кода. Такая модель позволяет получить обзор классов и пакетов, отношения между ними, изучить библиотеки и просмотреть зависимости.

– Forwardengineering – позволяет создавать модель в виде диаграммы классов, устанавливать отношения между элементами (классами и пакетами). В среде IntelliJ IDEA автоматически генерируется исходный код по созданной модели. Поддерживается синхронизация модели и кода.

Eclipse – свободная интегрированная среда разработки модульных кроссплатформенных приложений [5]. В состав Eclipse в числе прочих входят проекты для реализации метамодели UML 2.0 для поддержки разработки инструментов моделирования и средство для создания моделей и генерации кода Modeling Framework.

БИБЛИОГРАФИЧЕСКИЕ ССЫЛКИ

1. Фаулер М., Скотт К. UML в кратком изложении. Применение стандартного языка объектного моделирования : пер. с англ. М. : Мир, 1999.
2. Буч Г., Рамбо Д., Джекобсон А. Язык UML. Руководство пользователя : пер. с англ. М. : ДМК, 2000.
3. Сайт разработчика Microsoft [Электронный ресурс]. URL: https://msdn.microsoft.com
4. Сайт разработчика Intellij IDEA [Электронный ресурс]. URL: https://www.jetbrains.com/help/idea/2016.1/working-with-diagrams.html
5. Сайт разработчика Eclipse [Электронный ресурс]. URL: https://eclipse.org/modeling/emf