они прилагают основные усилия к тому, чтобы найти аналогии текущей задаче в тех планах, алгоритмах, структурах данных и идиомах, которыми пользовались раньше или о которых было известно.

Специалисты в области информатики именуют и каталогизируют алгоритмы и структуры данных, но зачастую мы не заботимся о том, чтобы как-то назвать другие виды паттернов. Паттерны дают проектировщикам единую терминологию, которой можно пользоваться для общения, документирования и изучения возможных альтернатив. Система начинает выглядеть менее сложной, поскольку о ней становится возможным говорить на более высоком уровне абстракции, чем нотация языка проектирования или программирования.

Помощь при документировании и изучении

Знание описанных в книге паттернов проектирования помогает понимать существующие объектно-ориентированные системы, в большинстве которых паттерны применяются. Частенько слышатся сетования на то, что наследование в системах используется запутанно, а проследить поток управления очень трудно. По большей части такие жалобы связаны с непониманием использованных в системе паттернов. С другой стороны, если вы достаточно давно работаете с объектно-ориентированными системами, то, наверное, освоили описываемые здесь паттерны на собственном опыте. Отметим также, что знание паттернов поможет начинающему проектировщику работать так, как работает эксперт.

Понять систему, которая описана в терминах применяемых в ней паттернов проектирования, намного проще. В противном случае для выявления паттернов пришлось бы восстанавливать дизайн по исходным текстам. Наличие единого словаря означает, что вам нет нужды описывать паттерн целиком; достаточно просто назвать его, а читатель поймет, о чем идет речь. Проектировщик, незнакомый с паттернами, должен будет сначала справиться о них, но это все равно проще, чем обратное конструирование.

Мы постоянно применяем паттерны в своих проектах: используем их при выборе имен для классов, как основу для размышлений и обучения хорошему проектированию, а также для описания проектов [BJ94]. Все это достаточно простые случаи работы. Но легко представить себе и более изощренные способы применения паттернов, например основанные на них CASE-средства или гипертекстовые документы.

Дополнение существующих методов

С помощью объектно-ориентированного проектирования можно создать хороший дизайн, обучить начинающих проектировщиков правильным приемам работы и стандартизовать методики разработки хороших проектов. Обычно метод проектирования определяет символы (как правило, графические) для моделирования различных аспектов проекта, а также набор правил, диктующих, как и когда применять каждый символ; зачастую удается описать проблемы, с которыми пришлось столкнуться в ходе работы над проектом, способы их разрешения и способы оценки полученного результата. Но опыт квалифицированных разработчиков не исчерпывается одними методами проектирования, которые они используют.

Думается, что важным дополнением к методам объектно-ориентированного проектирования как раз и являются наши паттерны. Они показывают, как применять такие базовые приемы, как объекты, наследование и полиморфизм, демонстрируют, как можно параметризовать систему алгоритмом, поведением, состоянием или видом объектов, которые предполагается создавать. Паттерны проектирования позволяют не просто зафиксировать результаты решений, а ответить на многочисленные «почему», возникающие в ходе проектирования. Разделы «Применимость», «Результаты» и «Реализация» в описаниях паттернов помогут вам сориентироваться при принятии решения.

Паттерны проектирования особенно полезны тогда, когда нужно преобразовать аналитическую модель в модель реализации. Вопреки многочисленным заверениям о беспрепятственном переходе от объектно-ориентированного анализа к проектированию, на практике этот процесс никогда не происходит гладко. В гибком проекте, ориентированном на повторное использование, имеются объекты, которых нет в аналитической модели. На проект оказывают влияние выбранные язык программирования и библиотеки классов. Аналитические модели часто приходится пересматривать, чтобы обеспечить повторное использование. Многие паттерны, включенные в каталог, непосредственно связаны с такого рода вопросами, почему мы и называем их паттернами проектирования.

Полноценная методика проектирования основывается не только на паттернах проектирования. Тут могут быть паттерны и анализа, и пользовательского интерфейса, и оптимизации производительности. Но паттерны – очень важная составная часть методики проектирования, которая до сих пор отсутствовала.

Цель реорганизации

Повторно используемое программное обеспечение часто приходится реорганизовывать [OJ90]. Паттерны проектирования помогают вам решить, как именно реорганизовать проект, и могут уменьшить вероятность реорганизации в будущем.

По Брайану Футу (Brian Foote), жизненный цикл объектно-ориентированной программы состоит из трех фаз: *прототипирования*, *экстенсивного роста* и *консолидации* [Foo92].

Фаза прототипирования характеризуется высокой активностью проектировщиков, направленной на то, чтобы программа начала работать. При этом быстро создается прототип, который последовательно изменяется до тех пор, пока не будут решены первоначальные задачи. Обычно на данном этапе программа представляет собой набор иерархий классов, отражающих сущности предметной области. Основной вид повторного использования – это прозрачный ящик и наследование.

После ввода в эксплуатацию развитие программы определяется двумя противоречивыми факторами: программе необходимо отвечать все новым требованиям и одновременно должна повышаться степень ее повторной используемости. Новые требования диктуют необходимость добавления новых классов и операций, быть может, даже целых иерархий классов. Эти требования могут удовлетворяться, когда начинается фаза экстенсивного роста программы. Но данный период не может продолжаться долго. В конце концов, программа становится слишком негибкой и не поддается дальнейшим изменениям. Иерархии классов более не соответствуют