Тема 1. Введение в паттерны

Понятие паттерна проектирования

При создании программных систем перед разработчиками часто встает проблема выбора тех или иных проектных решений. В этих случаях на помощь приходят паттерны. Дело в том, что почти наверняка подобные задачи уже решались ранее и уже существуют хорошо продуманные элегантные решения, составленные экспертами. Если эти решения описать и систематизировать в каталоги, то они станут доступными менее опытным разработчикам, которые после изучения смогут использовать их как шаблоны или образцы для решения задач подобного класса. Паттерны как раз описывают решения таких повторяющихся задач.

Концепция создания программного обеспечения с использованием паттернов, несомненно, очень важная, но относительно молодая, быть может, поэтому до сих пор нет четкого определения, что же такое паттерн. Об этом свидетельствуют непрекращающиеся дискуссии в популярной литературе и на соответствующих форумах в сети.

Например, следует ли считать алгоритмы и структуры данных паттернами? По этому вопросу существуют противоположные мнения. Согласно одному из них, алгоритмы являются вычислительными паттернами, а хорошо известная фундаментальная монография Дональда Кнута "Искусство программирования" по сути, представляет собой каталог таких паттернов. Согласно другому мнению, алгоритмы не являются паттернами, так как решаемые ими проблемы слишком малы (оперируют такими понятиями как вычислительная сложность и потребление ресурсов), а область решения хорошо очерчена. Паттерны же решают проблемы большего масштаба, при этом паттерн дает не конкретное решение, а некий путь к решению, причем, выбор правильного паттерна - задача нетривиальная, предполагающая от архитектора наличие интуиции, опыта, определенного творчества.

Впервые, в конце 1970-х годов Кристофером Александром был разработан каталог паттернов, предназначенных для проектирования зданий и городов [1]. В конце 1980-х годов Кент Бек и Вард Каннингем попытались перенести идеи Александра в область разработки ПО, составив 5 небольших паттернов для проектирования пользовательских интерфейсов на языке Smalltalk. В 1989 Джеймс Коплиен в целях обучения C++ внутри компании AT&T составил каталог идиом C++ (разновидность паттернов, специфичных

для языка программирования), а в 1991 на его основе вышла в свет книга "Advanced C++ Programming Styles and Idioms" (имеется русский перевод [7]).

Однако по настоящему популярным применение паттернов в индустрии разработки программного обеспечения стало после того, как в 1994 был опубликован каталог, включающий 23 паттерна объектно-ориентированного проектирования [5]. Этот каталог настолько популярен, что часто упоминается как паттерны GoF ("Gang of Four" или "банда четырех" по числу авторов).

В настоящее время паттерны продолжают непрерывно развиваться. Появляются новые паттерны, категории и методы их описания.

Описание паттернов проектирования

В силу популярности каталога GoF часто под паттернами проектирования подразумевают все виды паттернов программной индустрии, что является не совсем корректным. В области разработки программных систем существует множество паттернов, которые отличаются областью применения, масштабом, содержимым, стилем описания. Например, в зависимости от сферы применения существуют такие паттерны как паттерны анализа, проектирования, тестирования, документирования, организации процесса разработки, планирования проектов и другие.

В настоящее время наиболее популярными паттернами являются паттерны проектирования. Одной из распространенных классификаций таких паттернов является классификация по степени детализации и уровню абстракции рассматриваемых систем. Согласно [2], паттерны проектирования программных систем делятся на следующие категории:

- Архитектурные паттерны
- Паттерны проектирования
- Идиомы

Архитектурные паттерны, являясь наиболее высокоуровневыми паттернами, описывают структурную схему программной системы в целом. В данной схеме указываются отдельные функциональные составляющие системы, называемые подсистемами, а также взаимоотношения между ними. Примером архитектурного паттерна является хорошо известная программная парадигма "модель-представление-контроллер" (model-view-controller - MVC). В свою очередь, подсистемы могут состоять из архитектурных единиц уровнем ниже. **Паттерны проектирования** описывают схемы детализации программных подсистем и отношений между ними, при этом они не влияют

на структуру программной системы в целом и сохраняют независимость от реализации языка программирования. Паттерны GoF относятся именно к этой категории. Согласно [5], под паттернами проектирования объектно-ориентированных систем понимается описание взаимодействия объектов и классов, адаптированных для решения общей задачи проектирования в конкретном контексте.

> В русскоязычной литературе обычно встречаются несколько вариантов перевода оригинального названия design patterns - **паттерны проектирования**, **шаблоны проектирования**, **образцы**. Здесь в основном используется первый вариант, иногда второй.

Идиомы, являясь низкоуровневыми паттернами, имеют дело с вопросами реализации какой-либо проблемы с учетом особенностей данного языка программирования. При этом часто одни и те же идиомы для разных языков программирования выглядят по-разному или не имеют смысла вовсе. Например, в C++ для устранения возможных утечек памяти могут использоваться интеллектуальные указатели. Интеллектуальный указатель содержит указатель на участок динамически выделенной памяти, который будет автоматически освобожден при выходе из зоны видимости. В среде Java такой проблемы просто не существует, так как там используется автоматическая сборка мусора. Обычно, для использования идиом нужно глубоко знать особенности применяемого языка программирования.

Следует отметить, что в программной области существуют и другие виды паттернов, не относящиеся к проектированию вообще, например, паттерны анализа, тестирования, документирования и др.

Задача каждого паттерна - дать четкое описание проблемы и ее решения в соответствующей области. Для этого могут использоваться разные форматы описаний от художественно-описательного [1] до строгого, академического [5]. В общем случае описание паттерна всегда содержит следующие элементы:

1. Название паттерна. Представляет собой уникальное смысловое имя, однозначно определяющее данную задачу или проблему и ее решение.
2. Решаемая задача. Здесь дается понимание того, почему решаемая проблема действительно является таковой, четко описывает ее границы.
3. Решение. Здесь указывается, как именно данное решение связано с проблемой, приводится пути ее решения.
4. Результаты использования паттерна. Обычно приводятся достоинства, недостатки и компромиссы.

Результаты применения паттернов:

Один из соавторов GoF, Джон Влиссидес [4] приводит следующие преимущества применения паттернов проектирования:

1. Они (паттерны) позволяют суммировать опыт экспертов и сделать его доступным рядовым разработчикам.
2. Имена паттернов образуют своего рода словарь, который позволяет разработчикам лучше понимать друг друга.
3. Если в документации системы указано, какие паттерны в ней используются, это позволяет читателю быстрее понять систему.
4. Паттерны упрощают реструктуризацию системы независимо от того, использовались ли паттерны при ее проектировании.

Правильно выбранные паттерны проектирования позволяют сделать программную систему более гибкой, ее легче поддерживать и модифицировать, а код такой системы в большей степени соответствует концепции повторного использования.