1. Представление о прототипах в проектировании

За несколько десятилетий сложилось общепринятое представление о **прототипах в проектировании,** причём оно родилось вне программирования и, следовательно, не ограничено рамками проектирования программного обеспечения.

Родоначальником идеологии использования прототипов в проектировании как средства передачи знаний стал в 70-е годы Кристофер Александер[1]. Он написал несколько книг, в которых задокументировал ряд образцовых решений в строительстве и архитектуре, с тем, чтобы их положительный опыт мог быть использован другими. Позже эту идею стало активно перенимать сообщество разработчиков программного обеспечения, хотя, конечно же, в неназванном, или неоформленном, виде она присутствовала всегда.

Главным импульсом к распространению методов проектирования программного обеспечения с использованием прототипов стала книга «Прототипы проектирования, элементы повторно используемого объектно-ориентированного программного обеспечения», авторство которой принадлежит, как их часто называют, «банде четырех» (Gang of Four, GoF) в составе **Эрика Гаммы, Ричарда Хелма, Ральфа Джонсона и Джона Влисидеса**. Они проделали весьма нужную и актуальную работу. Не изобретая ничего самостоятельно, члены «банды» собрали удачные проектные решения и документально их оформили в книге **«Приёмы объектно-ориентированного проектирования. Паттерны проектирования»** . С тех пор вышел еще целый ряд книг по технологии программирования с использованием прототипов; они различаются областями приложения и функциональным назначением, но в отношении того, что такое прототип, все авторы солидарны.

Остановимся немного на книге. Она разделена на две части, причём первые две главы отведены для изучения возможности и ловушек объектно-ориентированного программирования, а остальные главы связаны с описанием классических шаблонов проектирования программного обеспечения. Книга включает в себя примеры в C ++ и Smalltalk. Она оказала влияние на области программной инженерии и рассматривается как важный источник для объектно-ориентированной теории и практики проектирования.

Существует несколько близких определений того, что такое прототип. Самому Кристоферу Александеру принадлежит следующее определение:

«Каждый прототип представляет собой правило, состоящее из трёх

[1] **Кристофер Вольфганг Александер** (англ. *Christopher Wolfgang Alexander*; род. 4 октября 1936, Вена, Австрия) — архитектор и дизайнер, создатель более 200 архитектурных проектов в Калифорнии, Японии, Мексике и в других частях мира. (Википедия)

составляющих, которые выражают отношения контекста, проблемы и решения».

Есть более общее определение, по Мартину Фоулеру:

«Прототип есть идея, которая однажды была практически реализована и, вероятно, может быть использована в будущем».

1.1. Общие характеристики прототипов по Джону Крупи

1. Прототипы рожаются в процессе практической деятельности.
2. Для записи **прототипов обычно** используют некоторый структурированный формат.
3. Прототипы предотвращают от «изобретения колеса».
4. Прототипы существуют на различных уровнях абстракции.
5. Прототипы не являются чем-то застывшим, а предполагают непрерывное улучшение.
6. Прототип это всегда артефакт в том смысле, что это продукт, сделанный человеком, отличающийся от природного объекта, доступный для вторичного использования.
7. В прототипах объединяются проектирование и опыт.
8. Несколько прототипов могут быть использованы совместно для решения проблемы

В терминологии UML вместо слова «прототип» используется слово **«прецедент».** Понятие прецедента является очень важным в проектировании. Кратко определить прецедент можно следующим образом:

Прецедент – это последовательность действий, которые актор совершает над системой для достижения определённого результата.

Фактически он представляет с собой сценарий взаимодействия пользователя управляемого объекта с системой, имеющий самостоятельное значение.

Разбиение функциональности системы на отдельные прецеденты служит примерно той же цели, что и разбиение сложного алгоритма на подпрограммы.

- во-первых, это структурирование функциональности, поскольку сценарии отдельных функций легче прорабатывать, не отвлекаясь на посторонние детали.
- во-вторых, это избежание повторной работы над одним и тем же, подобно тому, как подпрограммы могут вызываться из разных мест программы.

Под *прецедентом* (case) понимается некая ситуация единичного или

множественных взаимодействий, происходящая между проектируемой системой и пользователем, либо другой системой, внешней по отношению к проектируемой.

Актором, участвующим в прецеденте, называется объект, взаимодействующий с проектируемой системой. В качестве актора могут выступать как пользователи, так и внешние программные или аппаратные системы. Таким образом, понятие ***прецедента*** можно определить следующим образом: ***прецедент*** (use case), — это последовательность взаимодействий между одним или несколькими акторами и проектируемой системой.

Прецеденты можно уподобить подпрограммам, на которые разделяется объёмная программа. Однако структура достаточно сложной программы редко бывает плоской, однородной. Обычно главная программа содержит вызов нескольких подпрограмм, те в свою очередь вызывают подпрограммы второго уровня и т.д.

Аналогичным образом и прецеденты могут образовывать иерархическую структуру. Однако в отличие от подпрограмм, которые обычно образуют простое отношение типа «A вызывает B» (за исключением не столь часто встречающихся взаимно рекурсивных подпрограмм), отношения между прецедентами несколько сложнее и разбиваются на 3 категории:

- ***включение;***
- ***расширение***;
- ***обобщение.***