Шаблоны проектирования

Определение

Шаблон проектирования (паттерн) — повторимая архитектурная конструкция, представляющая собой решение проблемы проектирования в рамках некоторого часто возникающего контекста.

Типы шаблонов

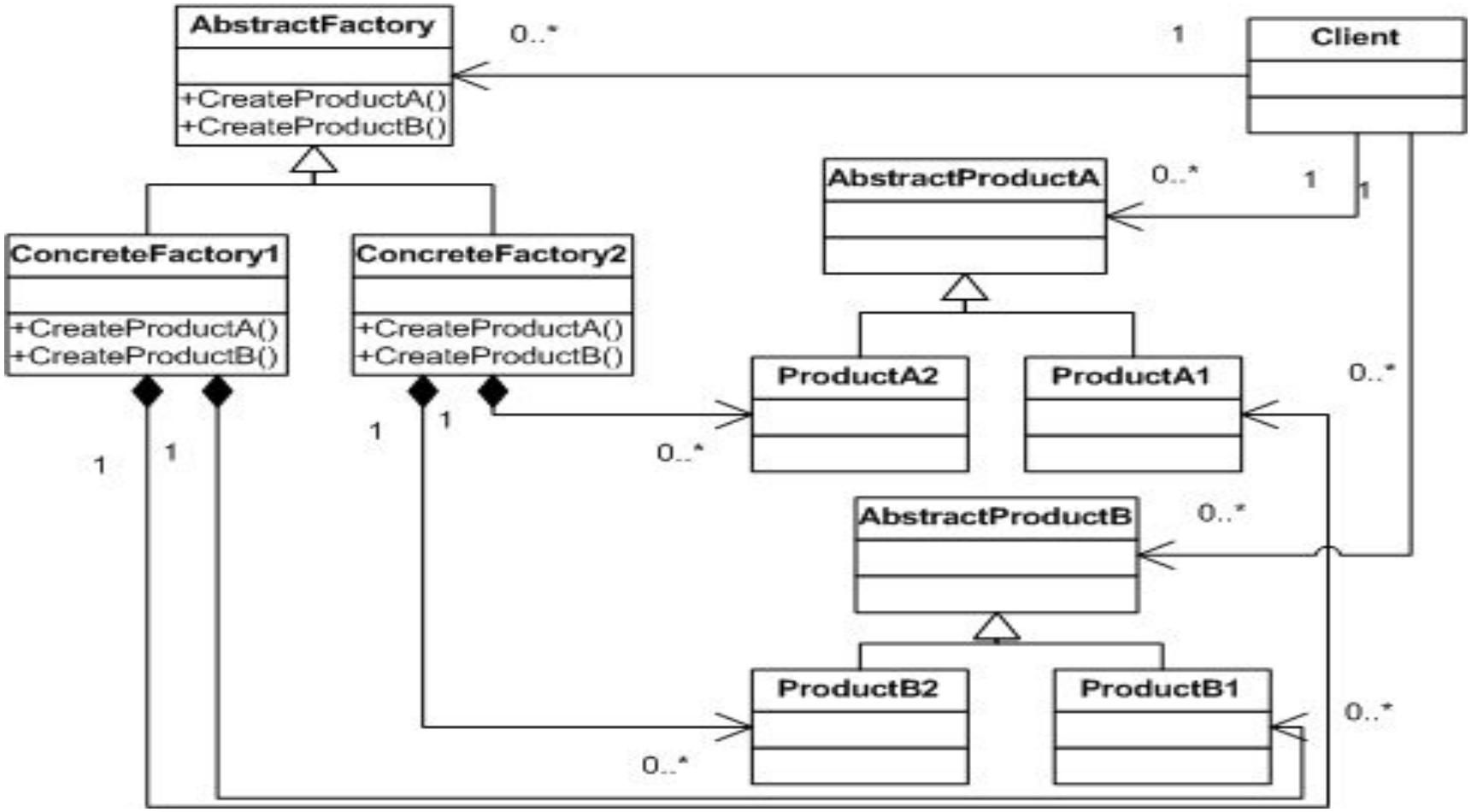
Порождающие	Структурные	Поведенческие
предназначены для создания объектов	для управления статическими, структурными связями между объектами.	обеспечивают координацию функционального взаимодействия между объектами

Порождающие шаблоны

Абстрактная фабрика (abstract factory) - предоставление интерфейса для создания семейств связанных между собой или зависимых друг от друга объектов без указания их конкретных классов

Фабрика должна иметь операции, которые создают новые объекты, объекты, называют продуктами

Абстрактная фабрика



Шаблон Строитель (Builder)

Упрощает создание сложных объектов путем определения класса,

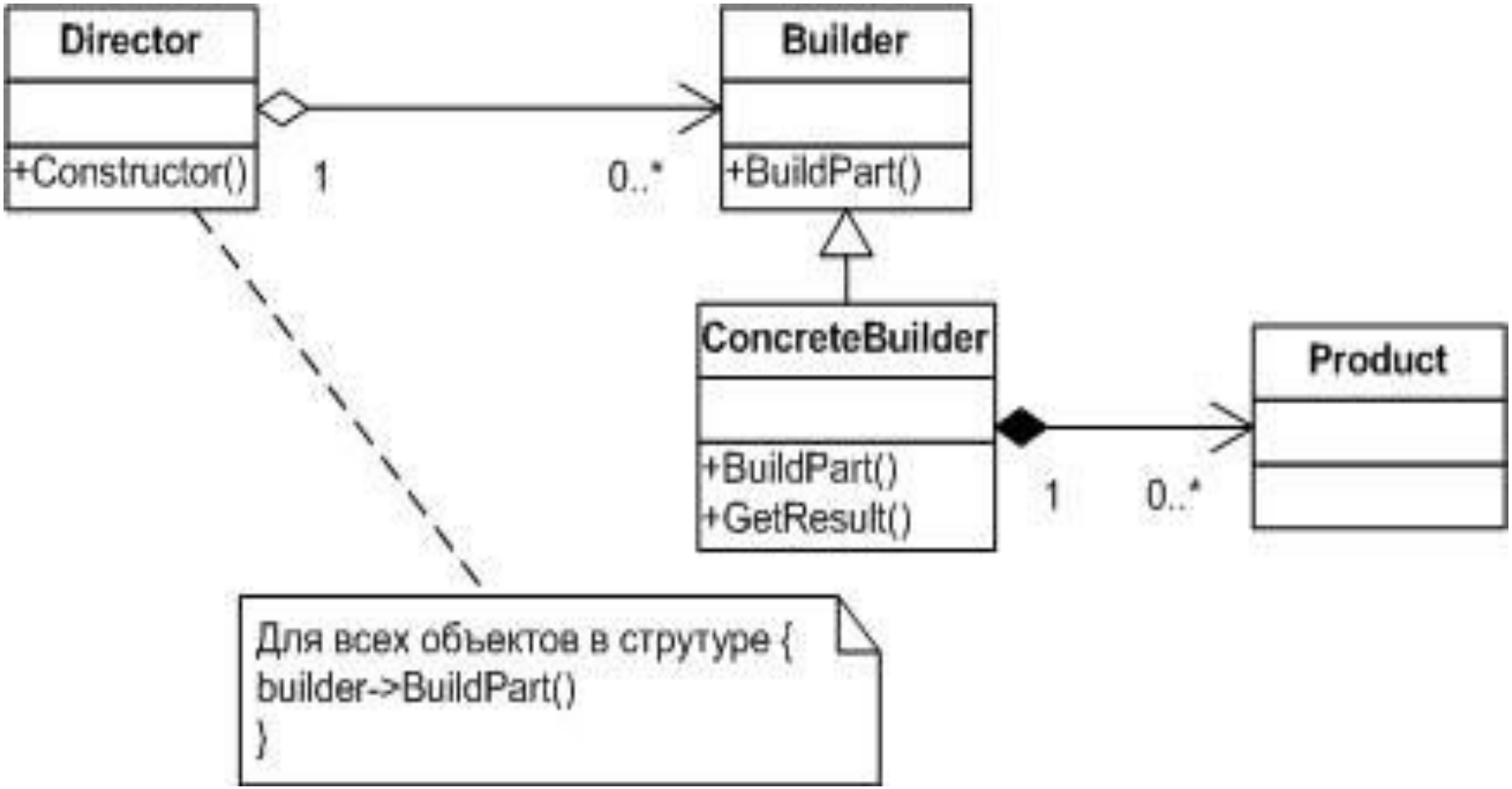
предназначенного для построения экземпляров другого класса. Шаблон Builder

генерирует только одну сущность. Хотя эта сущность в свою очередь может

содержать более одного класса, но один из полученных классов всегда является

главным.

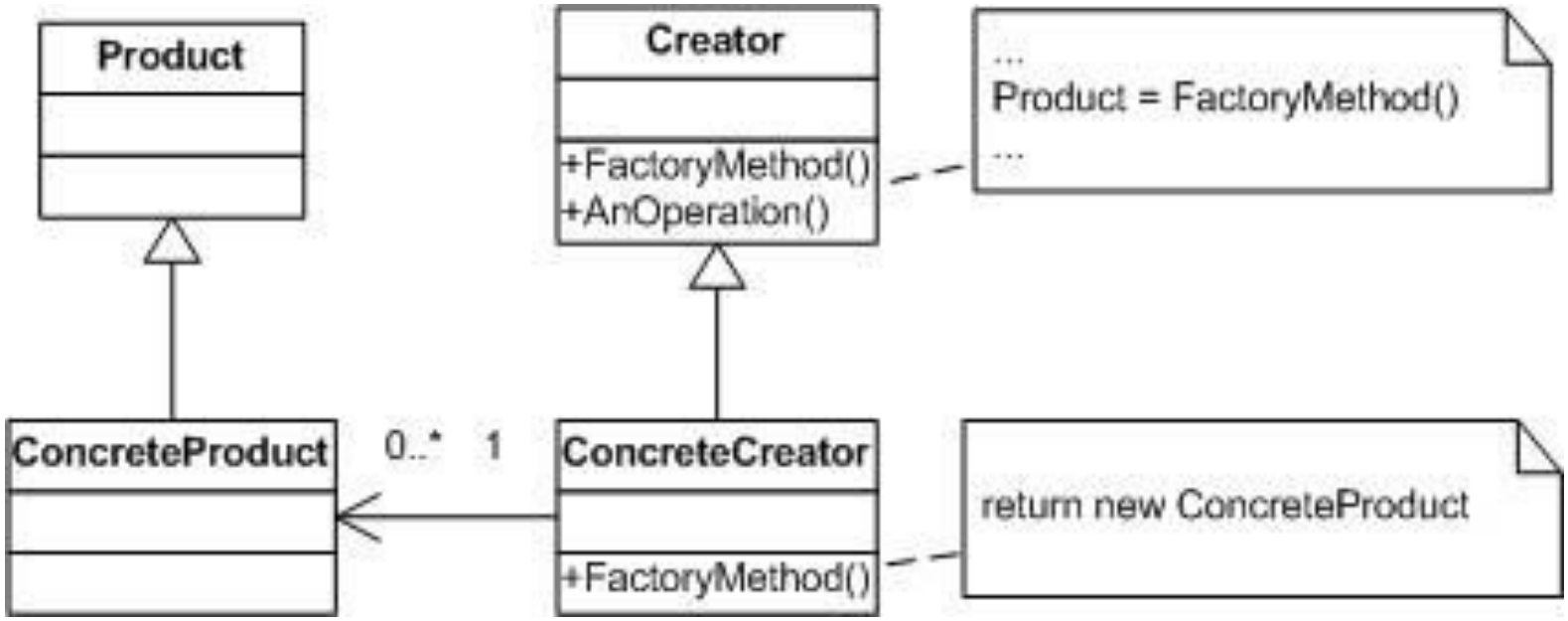
Шаблон Строитель (Builder)



Шаблон Фабричный метод (Factory Method)

Определяет стандартный метод создания объекта, не связанный с вызовом конструктора, оставляя решение о том, какой именно объект создавать, за подклассами.

Шаблон Фабричный метод (Factory Method)



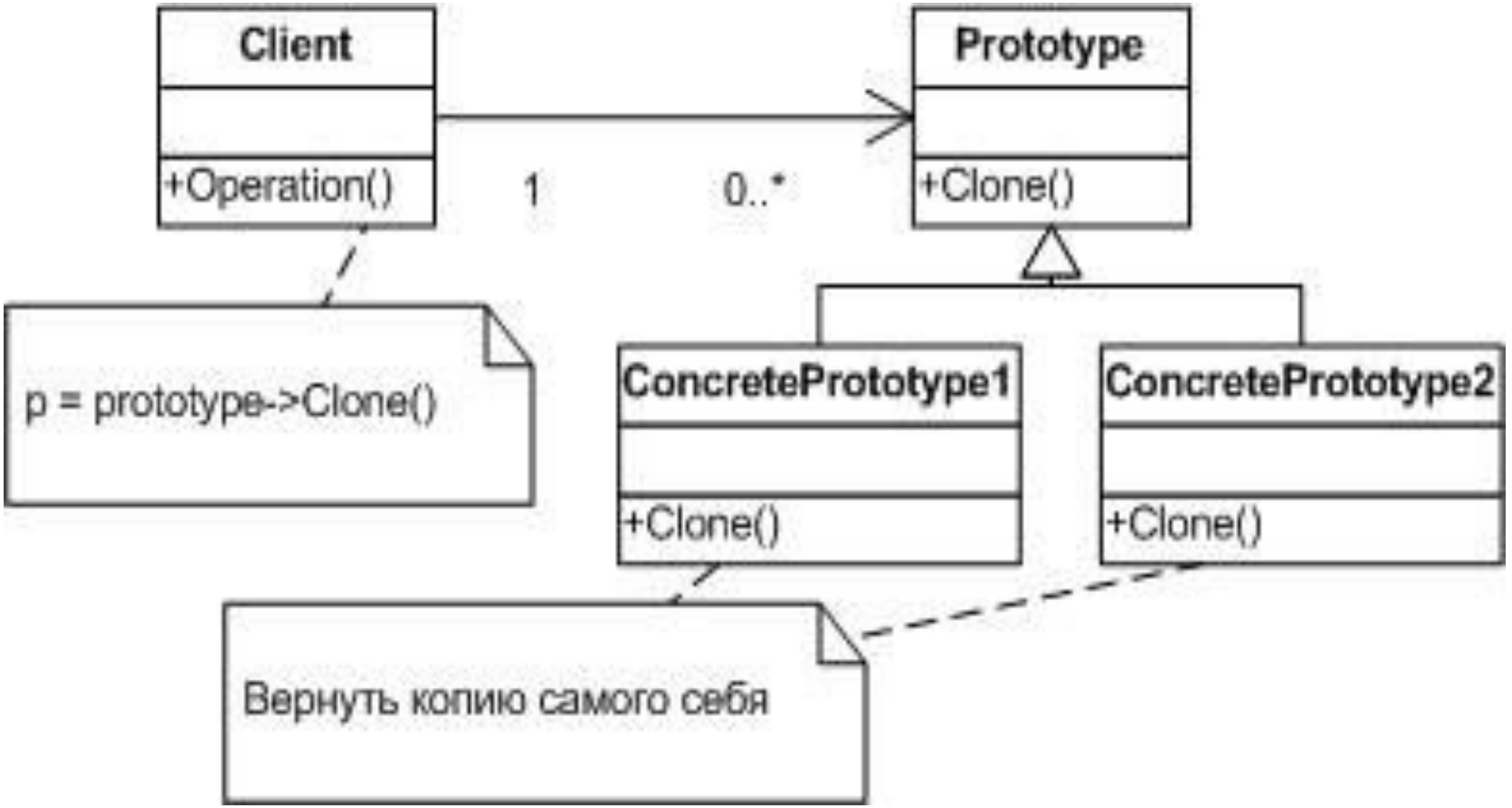
Шаблон Прототип (Prototype)

Облегчает динамическое создание путем определения классов, объекты которых могут создавать собственные дубликаты.

Задаёт виды создаваемых объектов с помощью экземпляра-прототипа и создаёт новые объекты путём копирования этого прототипа.

Это паттерн создания объекта через клонирование другого объекта вместо создания через конструктор.

Шаблон Прототип (Prototype)



Шаблон Одиночка (Singleton)

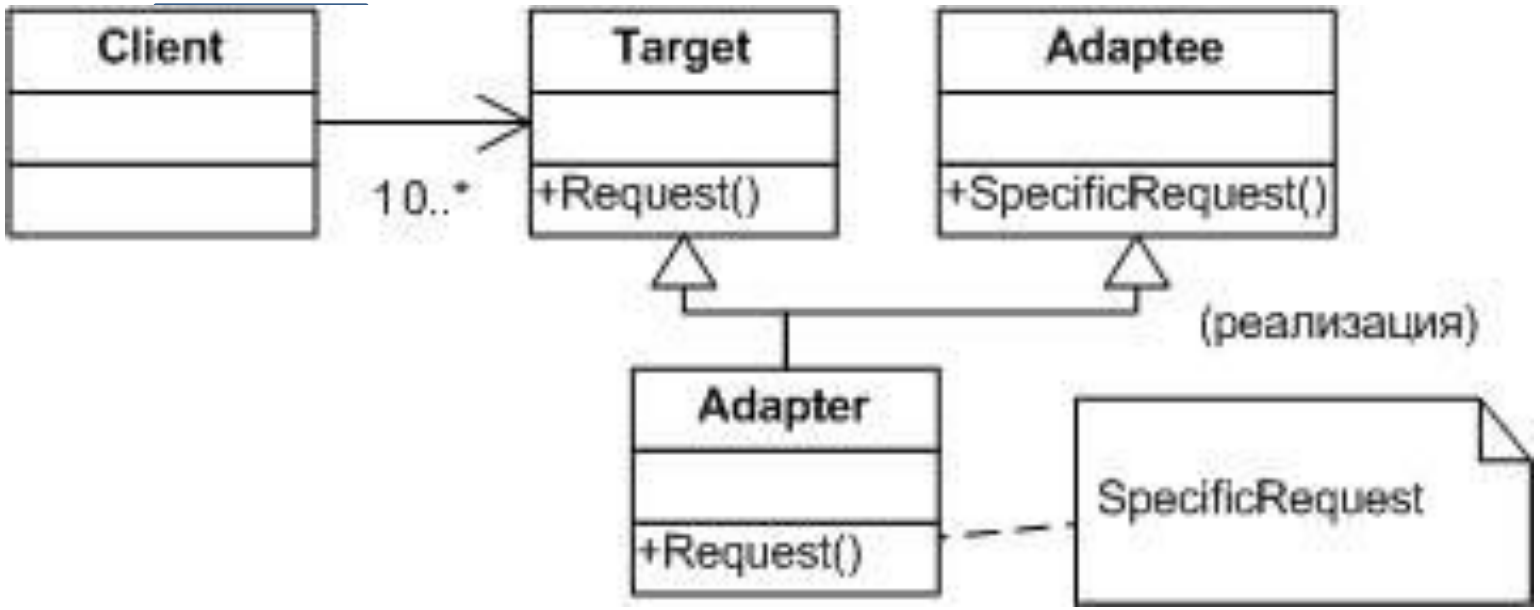
Обеспечивает наличие в системе только одного экземпляра заданного класса, позволяя другим классам получать доступ к этому экземпляру.

Структурные шаблоны

Адаптер (adapter) Обеспечение взаимодействия двух классов путем преобразования интерфейса одного из них таким образом, чтобы им мог пользоваться другой класс.

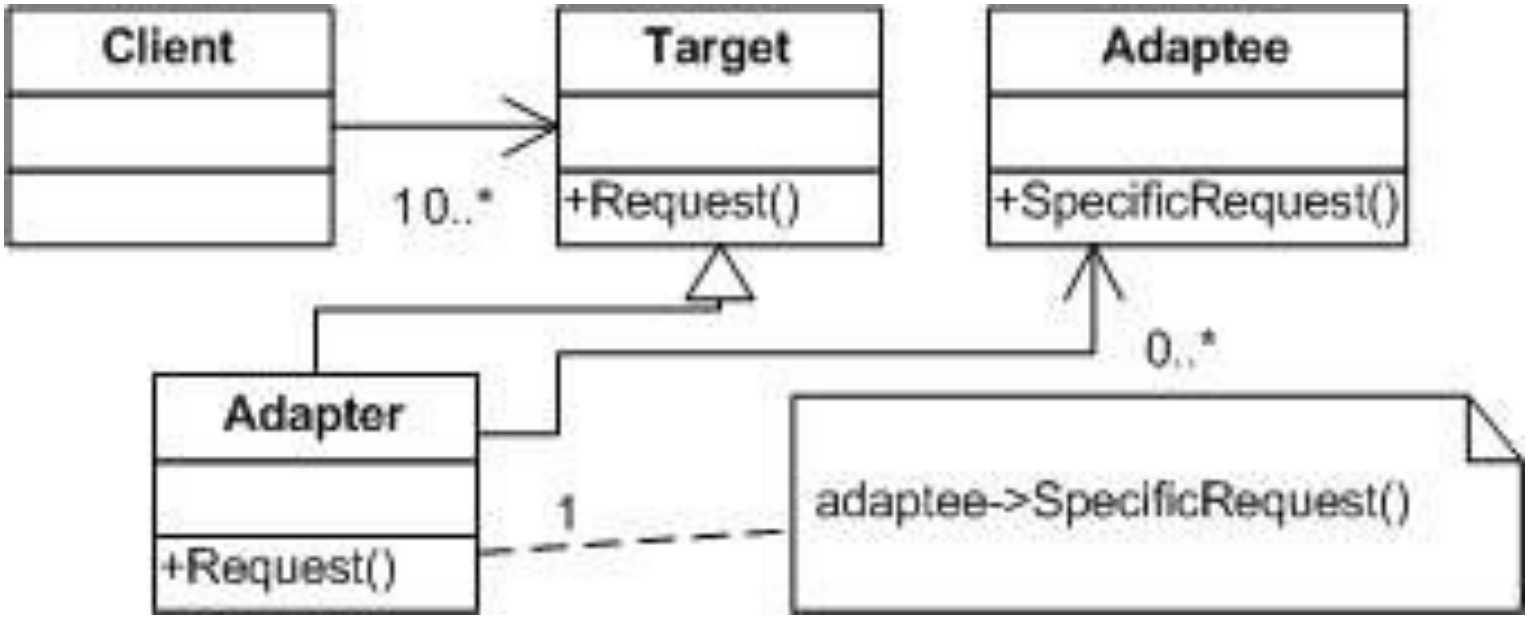
Шаблон адаптер (adapter)

На основе наследования:



Шаблон адаптер (adapter)

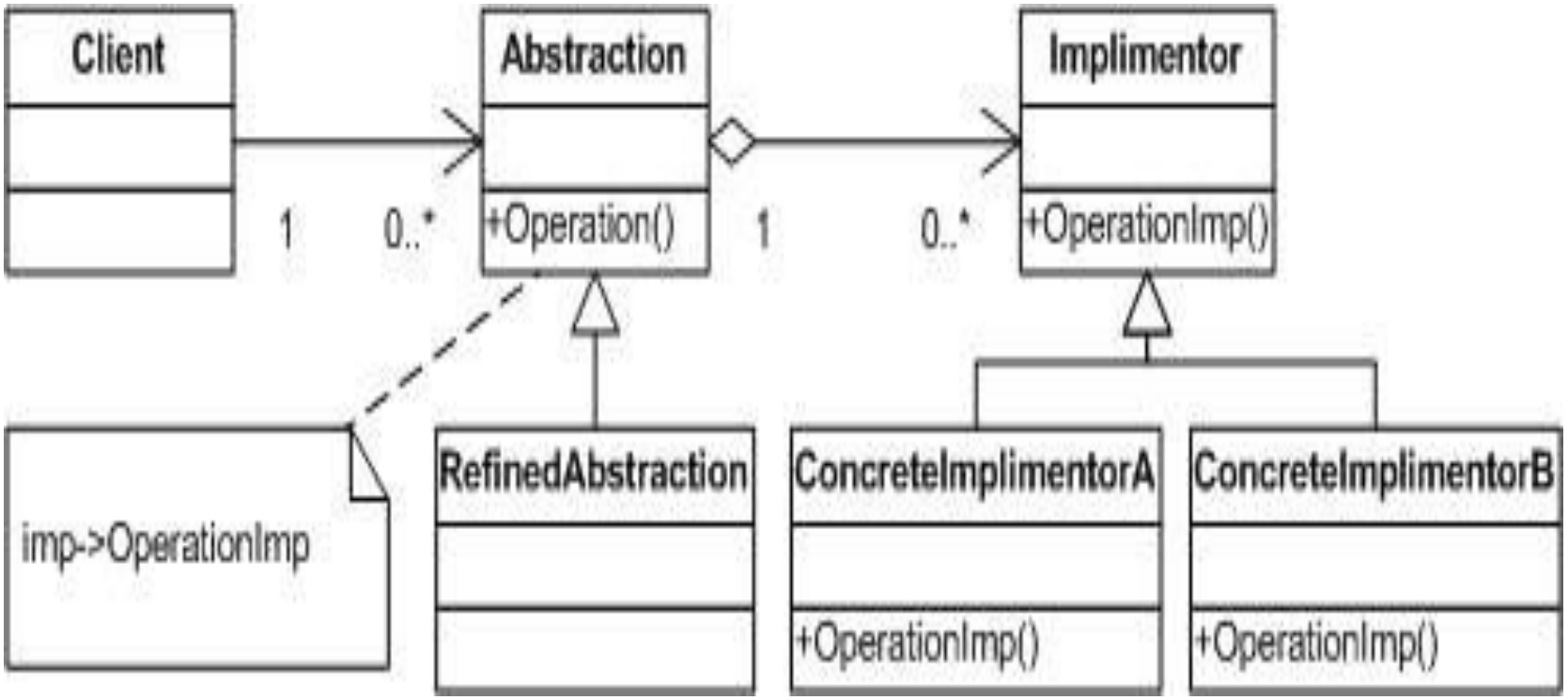
На основе композиции объектов:



Шаблон Мост (Bridge)

Разделение сложного компонента на две независимые, но взаимосвязанные иерархические структуры: функциональную абстракцию и внутреннюю реализацию.

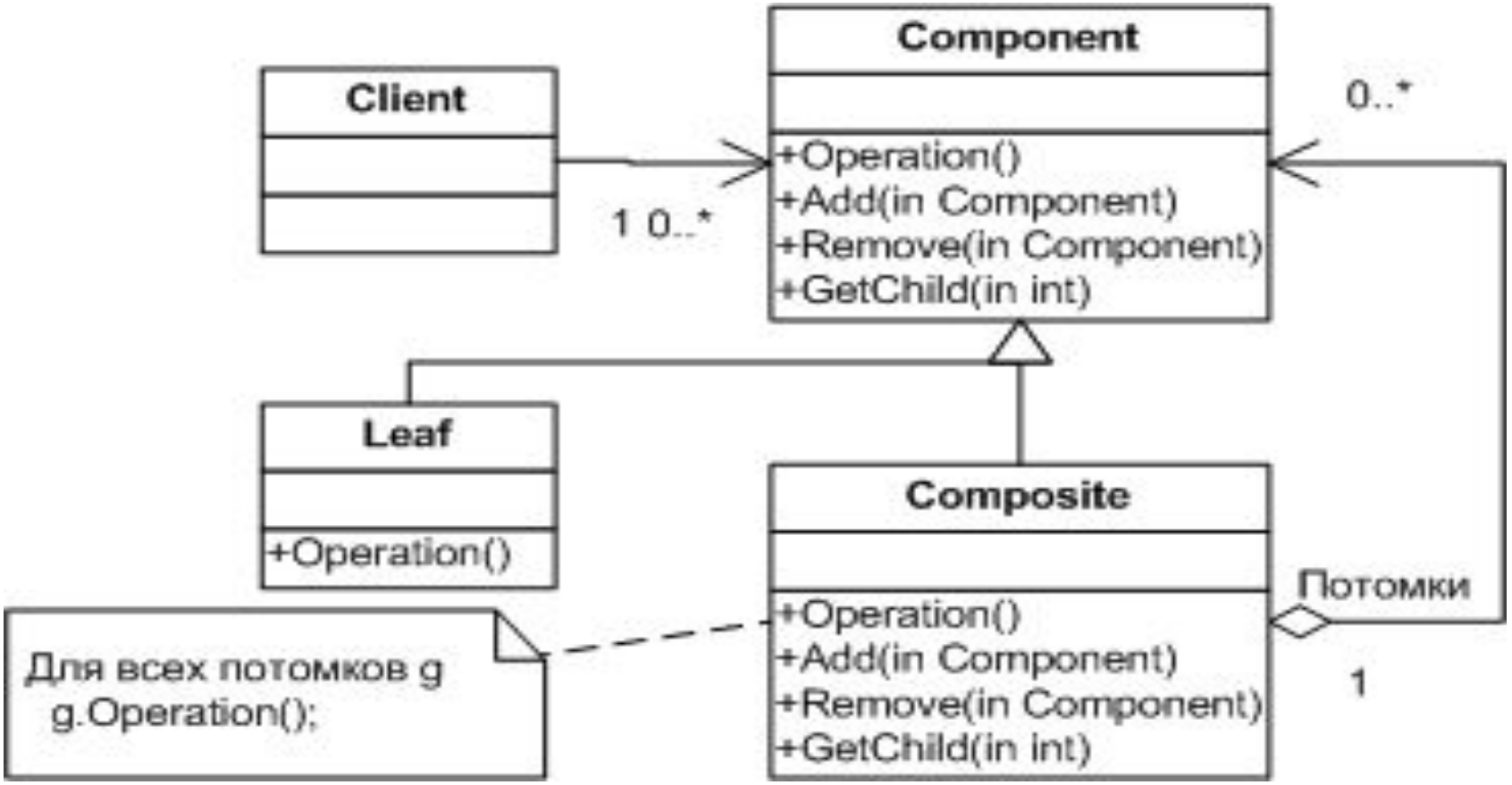
Шаблон Мост (Bridge)



Шаблон Компоновщик (Composite)

Предоставление гибкого механизма для создания иерархических древовидных структур произвольной сложности, элементы которых могут свободно взаимодействовать с единым интерфейсом

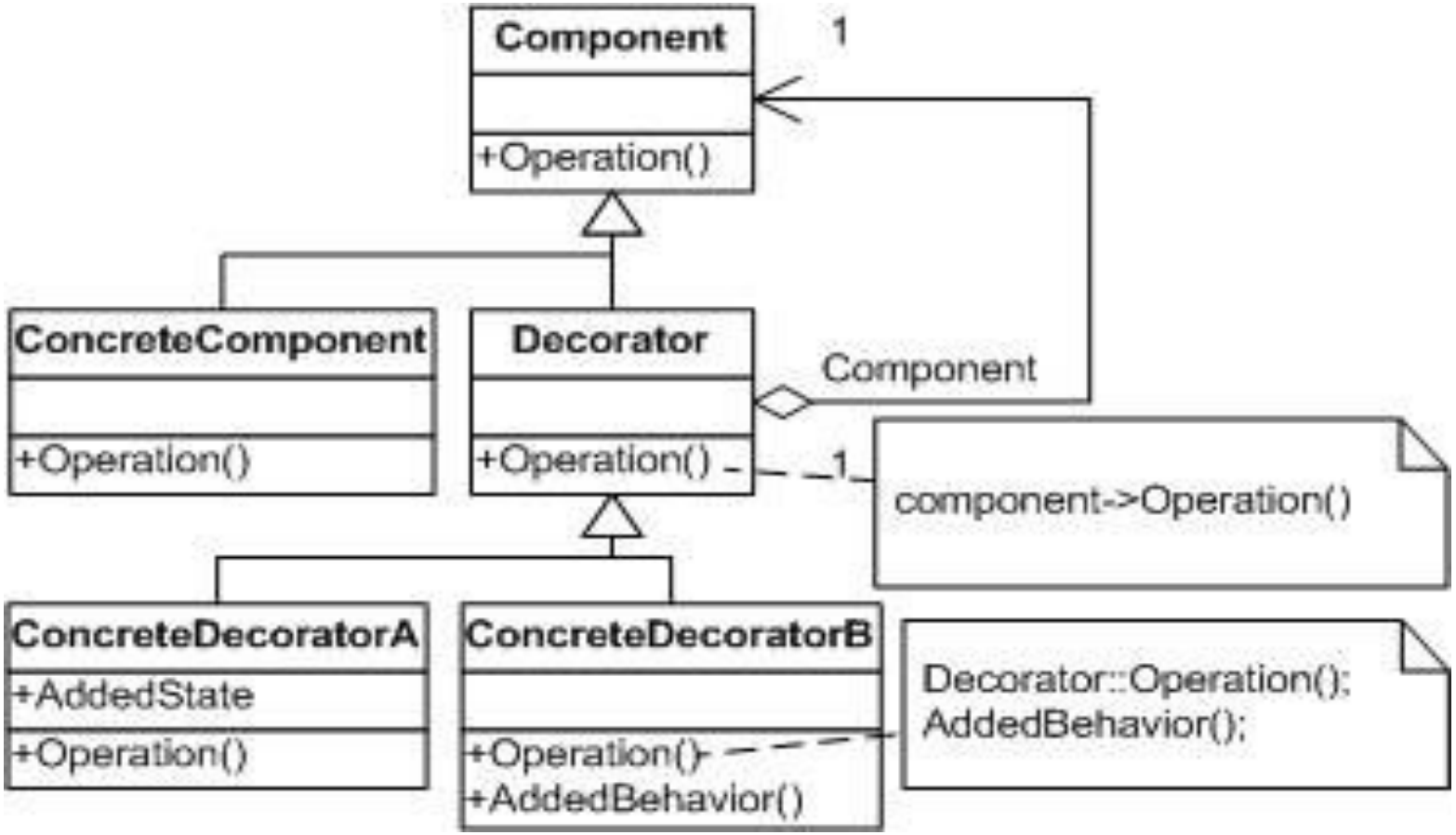
Шаблон Компоновщик (Composite)



Шаблон Декоратор (Decorator)

Предоставление механизма для добавления или удаления функциональности компонентов без изменения их внешнего представления или функций.

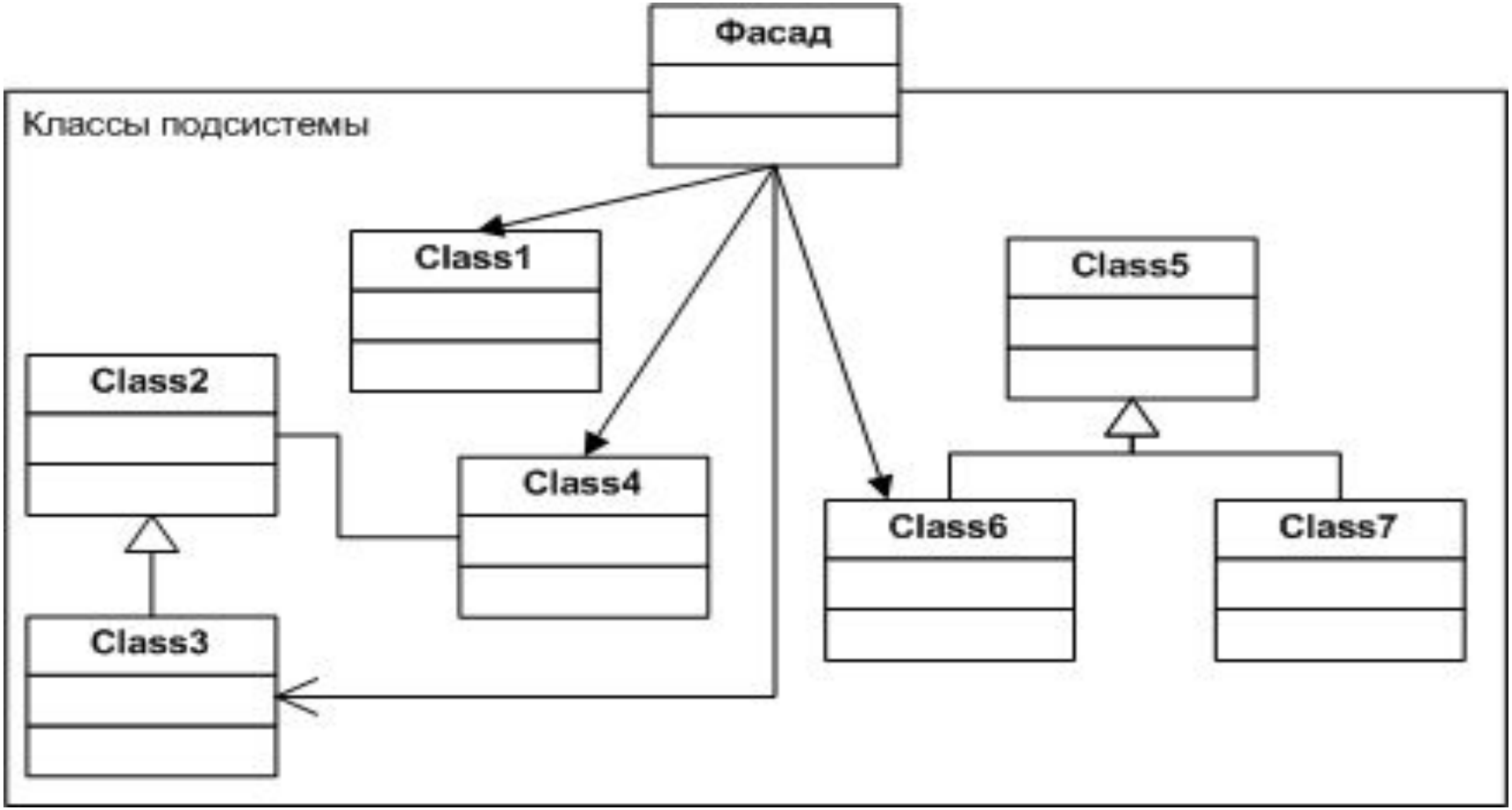
Шаблон Декоратор (Decorator)



Шаблон Фасад (Facade)

Создание упрощенного интерфейса для группы подсистем или сложной подсистемы.

Шаблон Фасад (Facade)



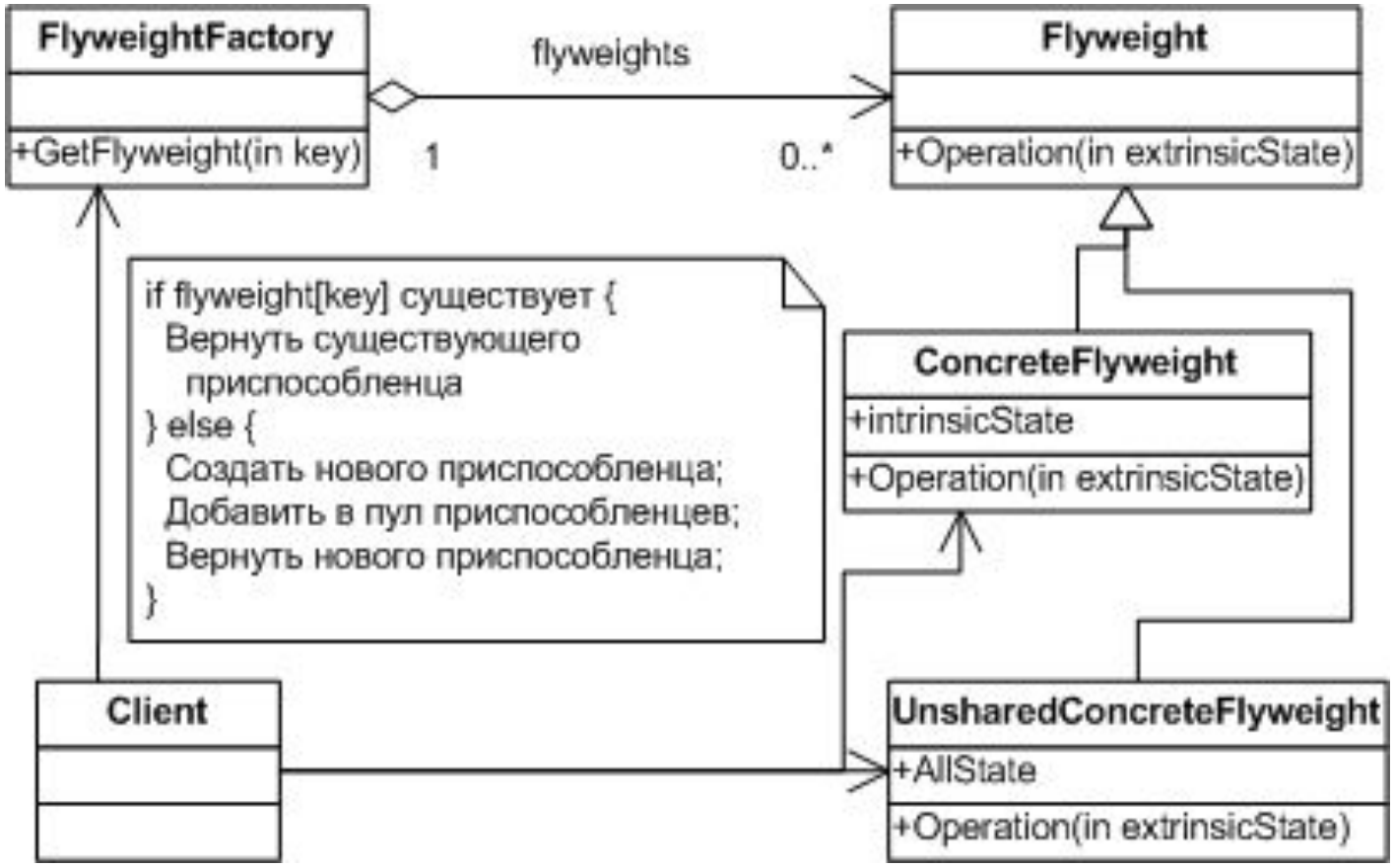
Шаблон Приспособленец (Flyweight)

Уменьшение количества объектов системы с многочисленными низкоуровневыми особенностями путем совместного использования подобных объектов.

Использует фабрику.

Использует разделение для эффективной поддержки множества мелких объектов.

Шаблон Приспособленец (Flyweight)



Шаблон Прокси (Proxy)

предоставляет объект, который контролирует доступ к другому объекту, перехватывая все вызовы

Функции:

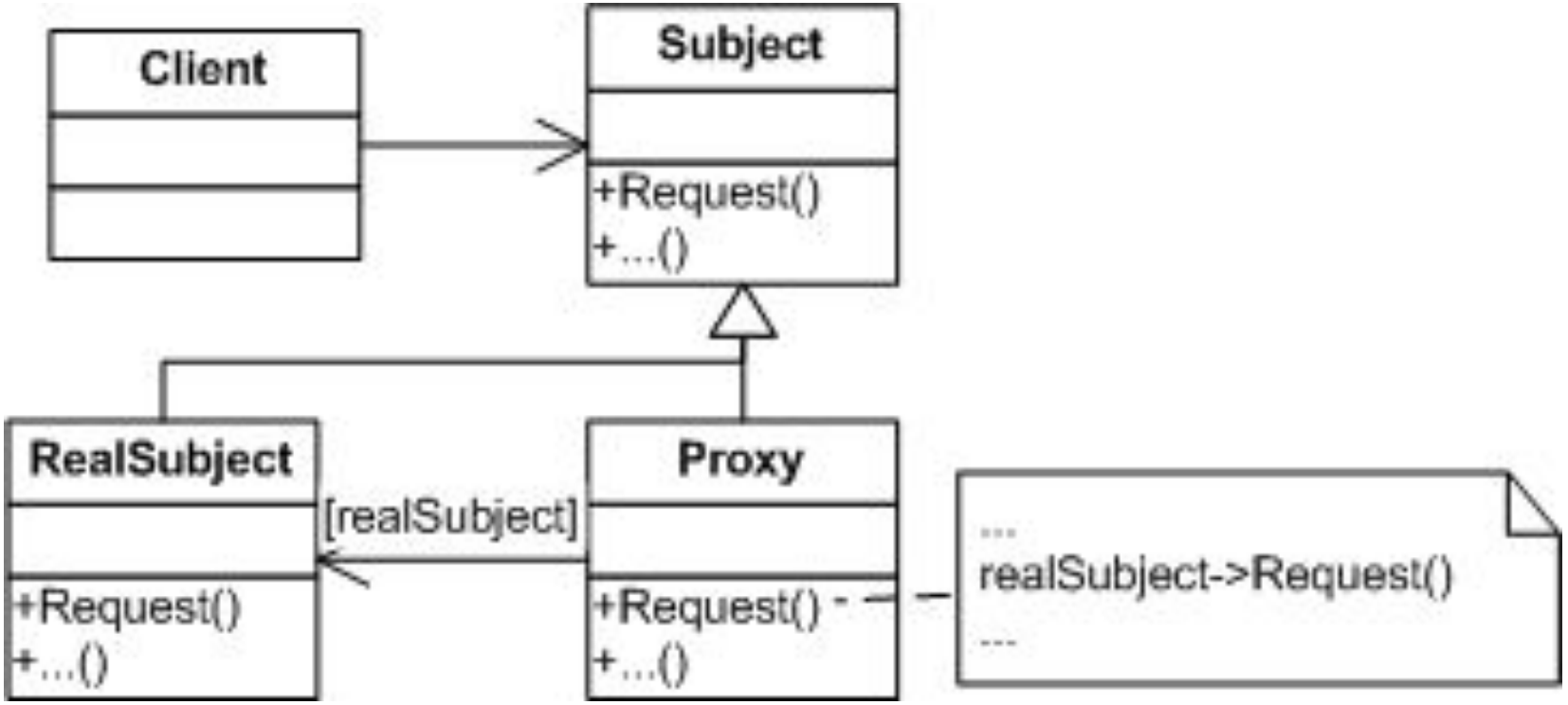
-логирование

-экранирование функций

-синхронизация (многопоточность)

-кеширование

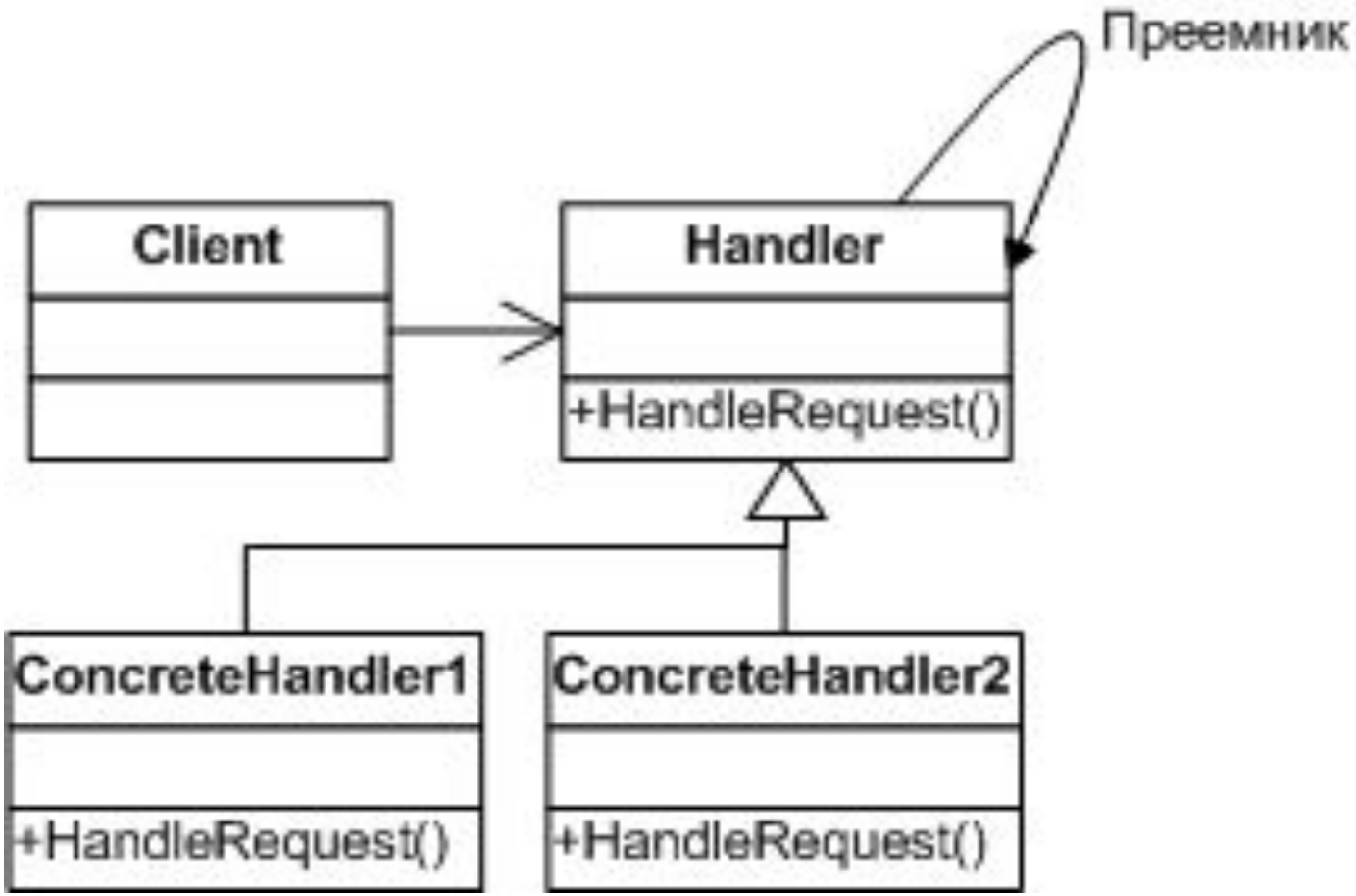
Шаблон Прокси (Proxy)



Поведенческие шаблоны

Цепочка ответственности (Chain of Responsibility). Предназначен для организации в системе уровней ответственности. Использование этого шаблона позволяет установить, должно ли сообщение обрабатываться на том уровне, где оно было получено, или же оно должно передаваться для обработки другому объекту.

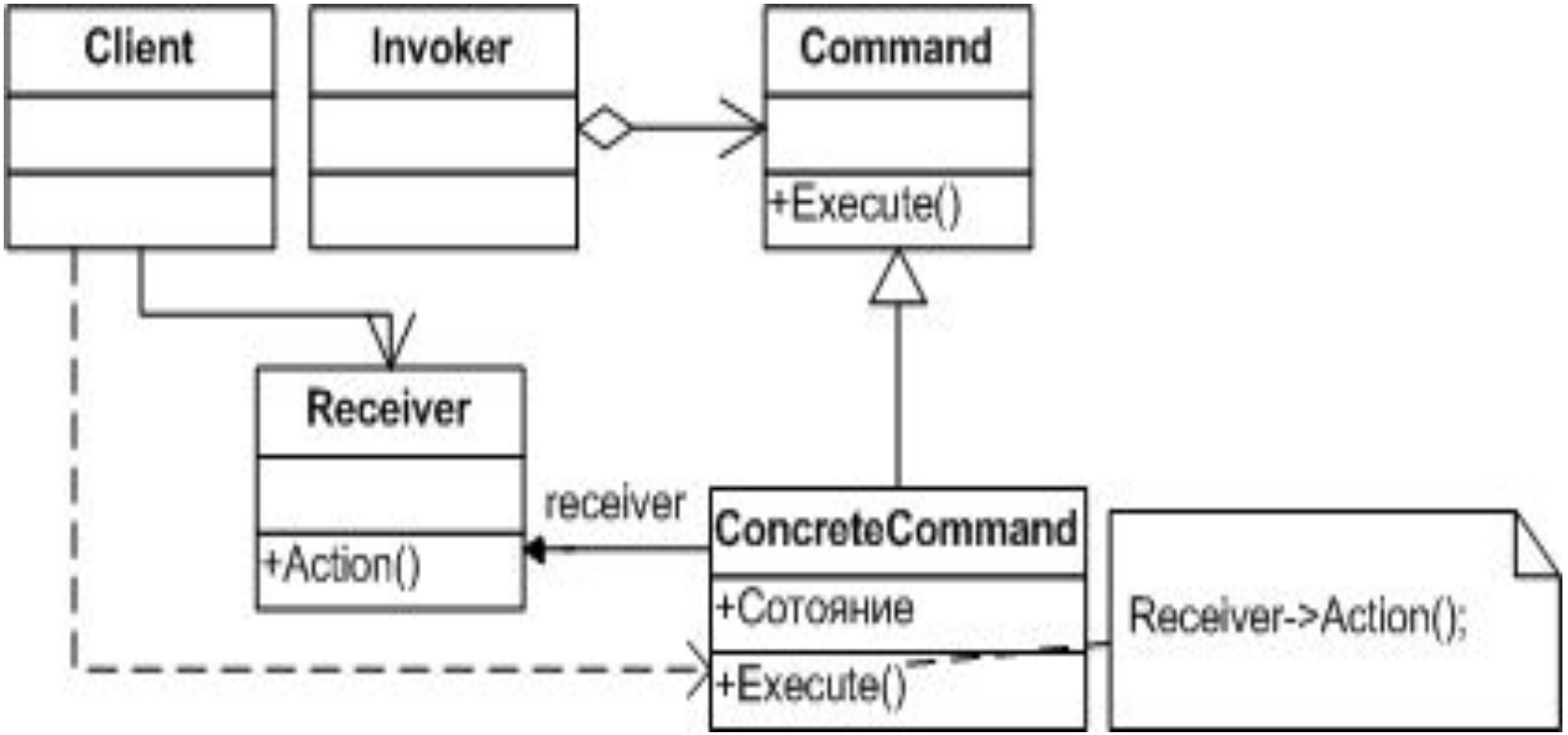
Шаблон цепочка ответственности (Chain of Responsibility)



Шаблон Команда (Command)

Обеспечивает обработку команды в виде объекта, что позволяет сохранять ее, передавать в качестве параметра методам, а также возвращать ее в виде результата, как и любой другой объект.

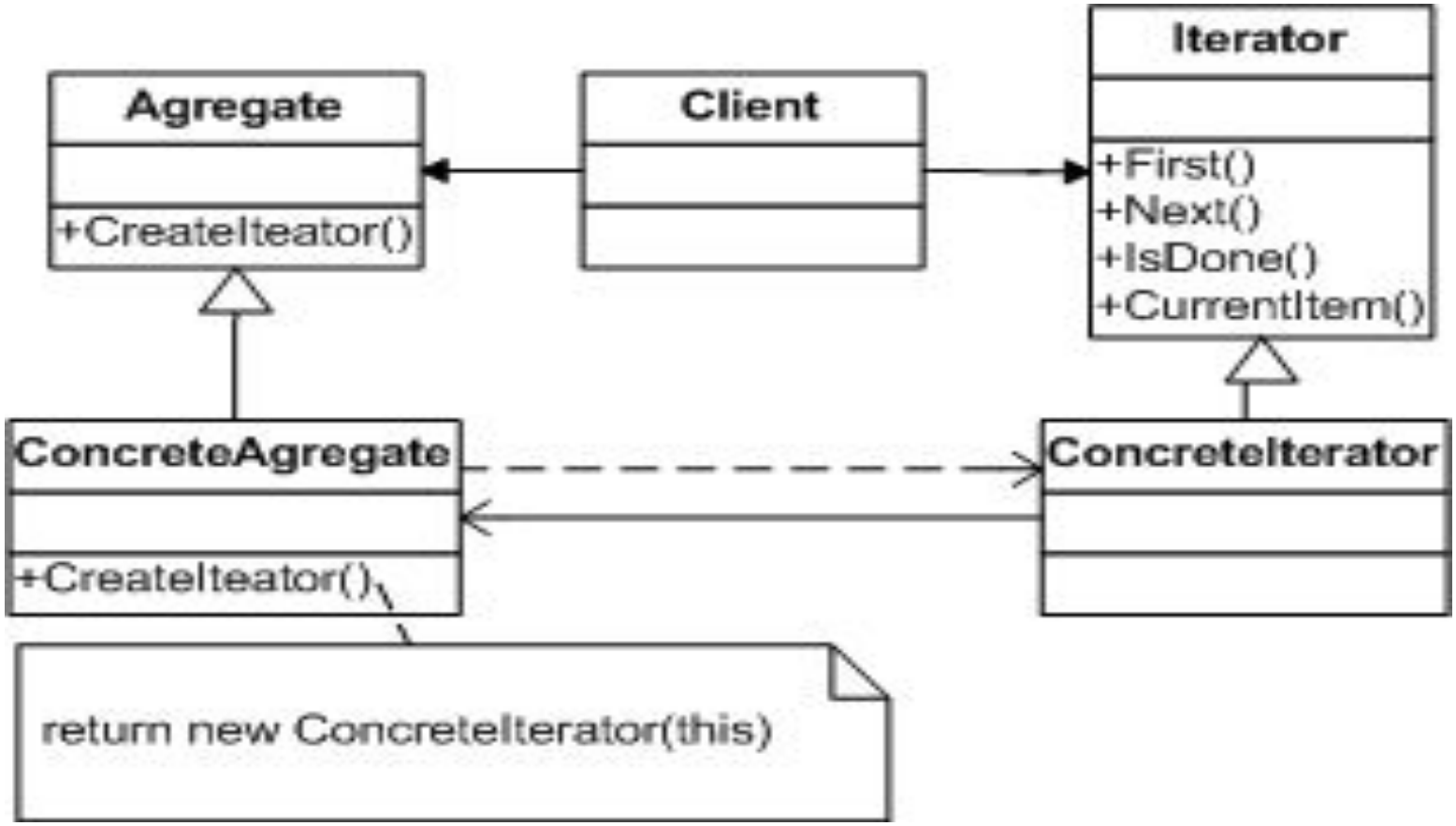
Шаблон Команда (Command)



Шаблон Итератор (Iterator)

Предоставляет единый метод последовательного доступа к элементам коллекции, не зависящий от самой коллекции и никак с ней не связанный.

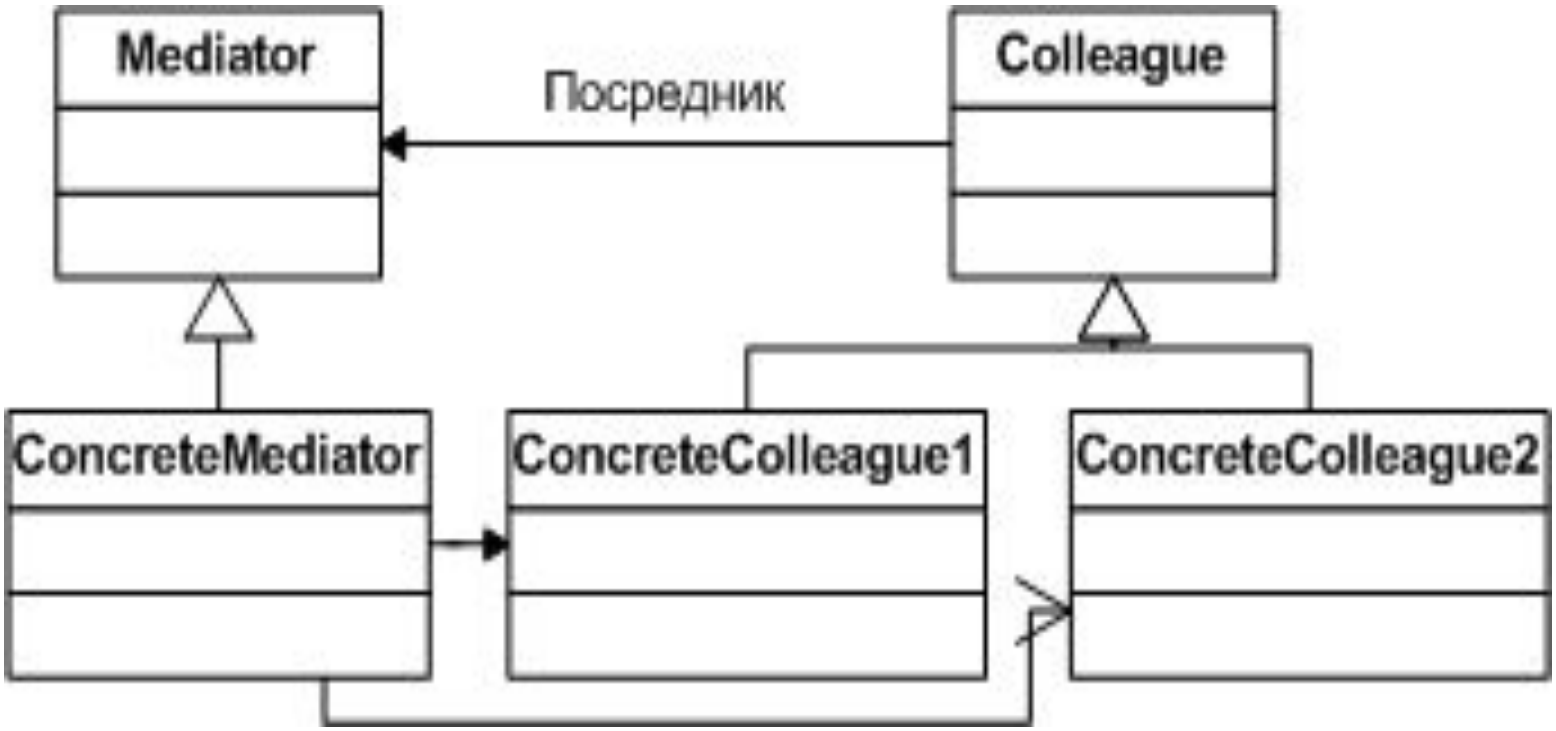
Шаблон Итератор (Iterator)



Шаблон Посредник (Mediator)

Предназначен для упрощения взаимодействия объектов системы путем создания специального объекта, который управляет распределением сообщений между остальными объектами.

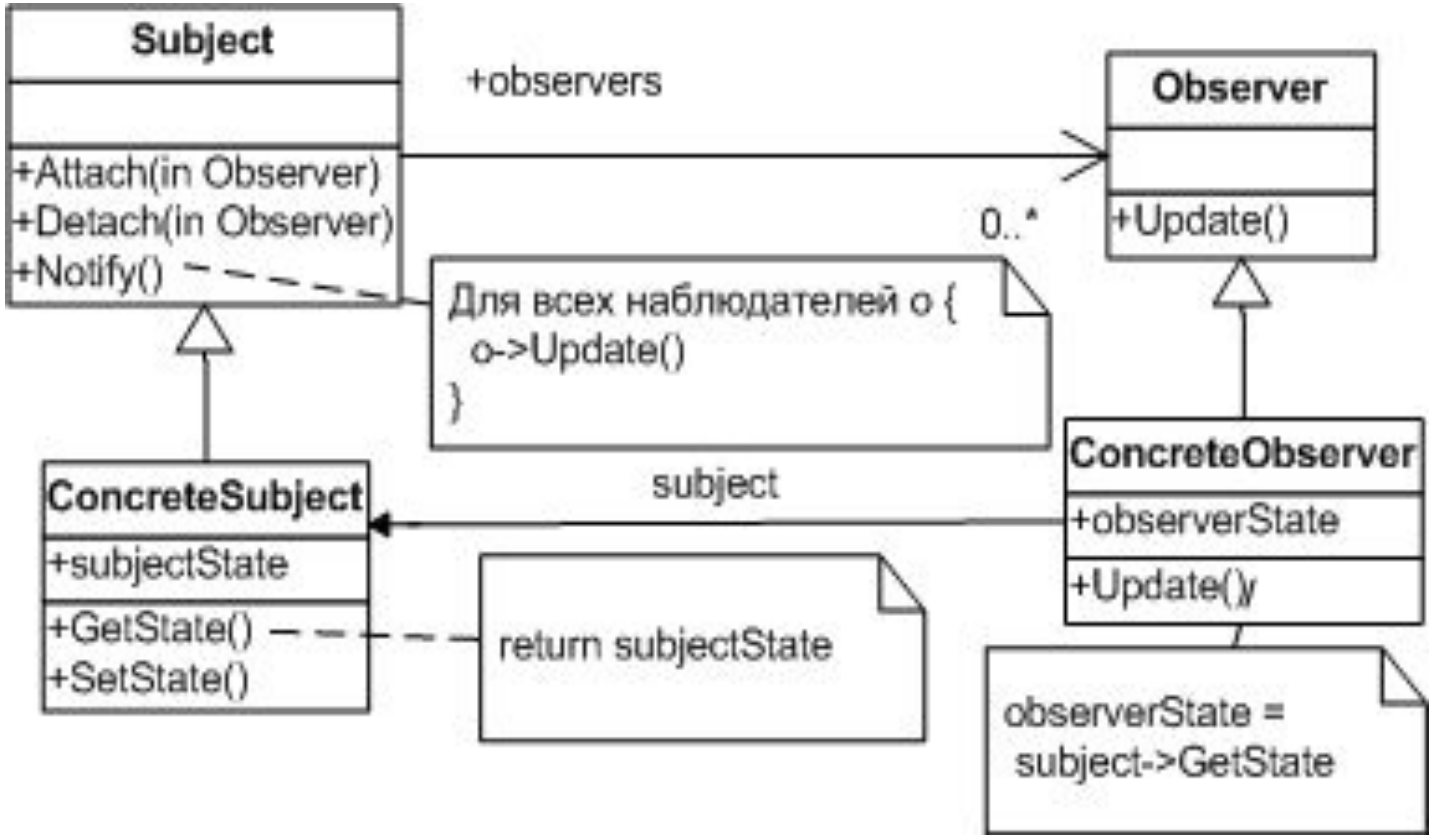
Шаблон Посредник (Mediator)



Шаблон Наблюдатель (Observer)

Предоставляет компоненту возможность гибкой рассылки сообщений интересующим его получателям.

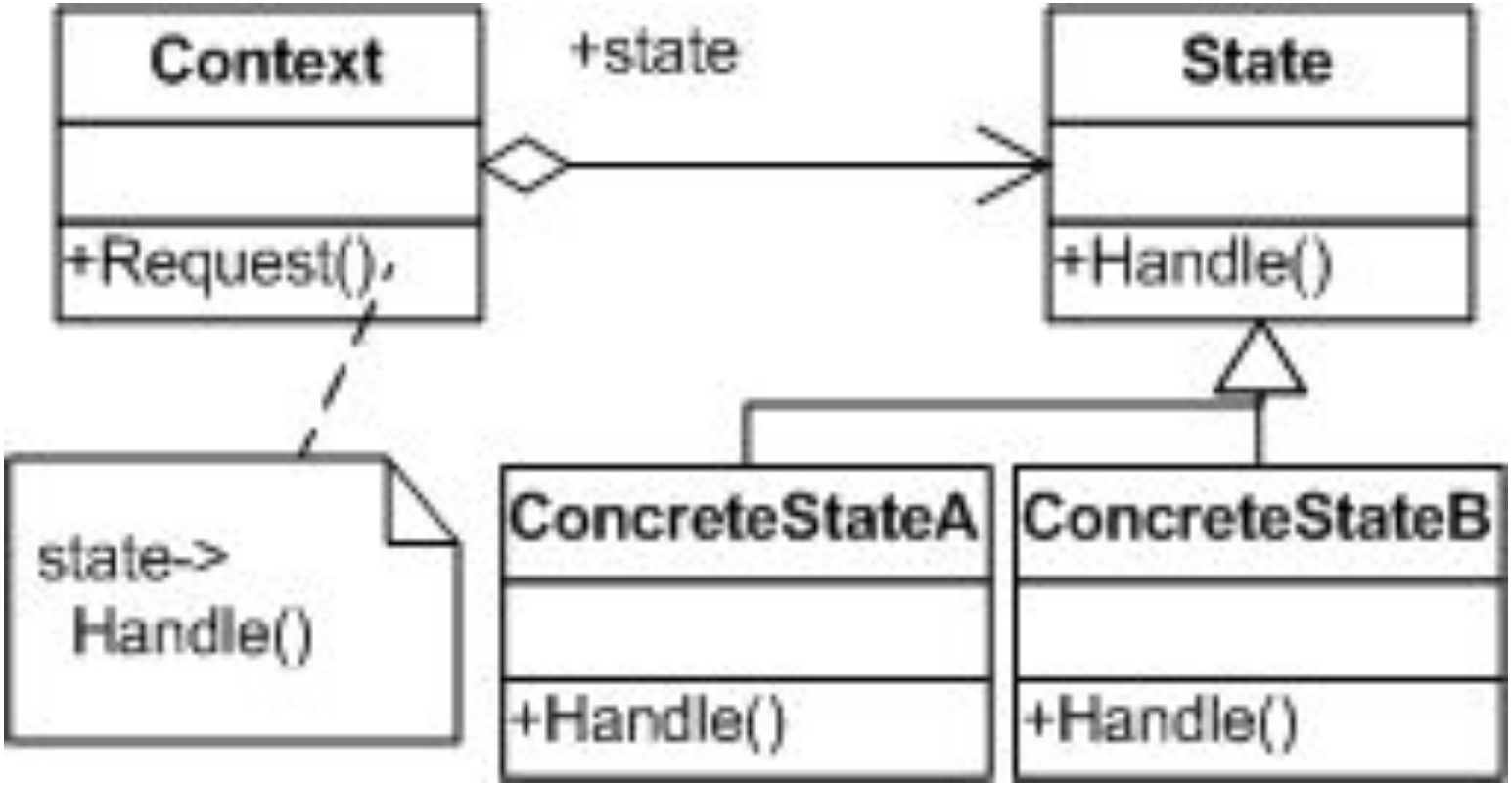
Шаблон Наблюдатель (Observer)



Шаблон Состояние (State)

Обеспечивает изменение поведения объекта во время выполнения программы.

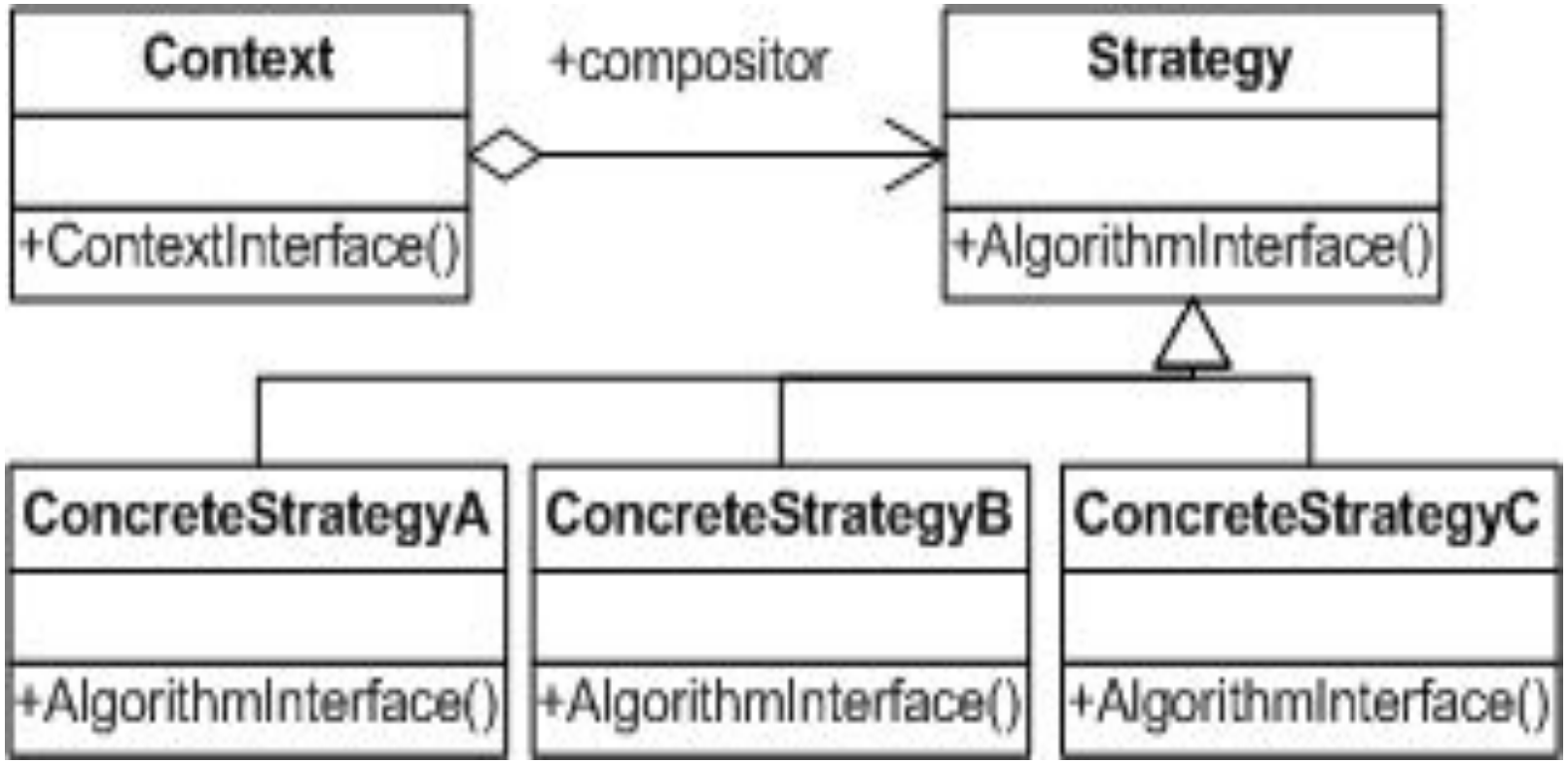
Шаблон Состояние (State)



Шаблон Стратегия (Strategy)

Предназначен для определения группы классов, которые представляют собой набор возможных вариантов поведения. Это дает возможность гибко подключать те или иные наборы вариантов поведения во время работы приложения, меняя его функциональность "на ходу".

Шаблон Стратегия (Strategy)



Шаблон Шаблонный метод (Template Method)

Предоставляет метод, который позволяет подклассам перекрывать части метода, не прибегая к их переписыванию.

Шаблон Шаблонный метод (Template Method)