

PYTHON Преподаватель: Жуков Николай Николаевич
Работу выполнила студентка ИКНиТО ИВТ: Шибаета Мария Дмитриевна

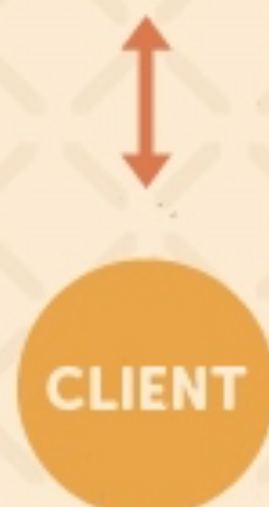
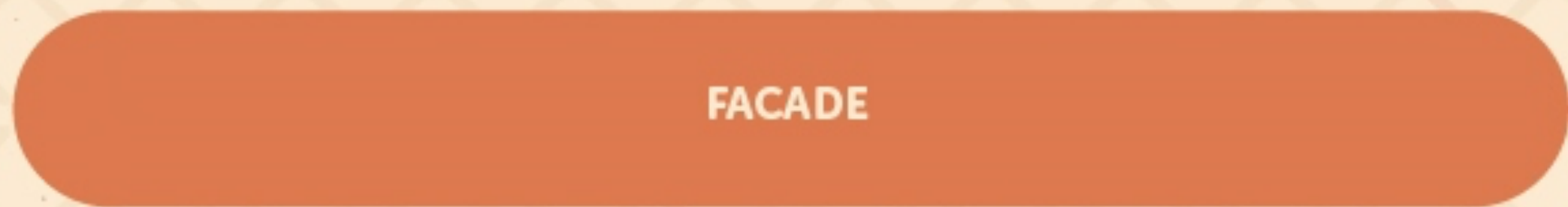
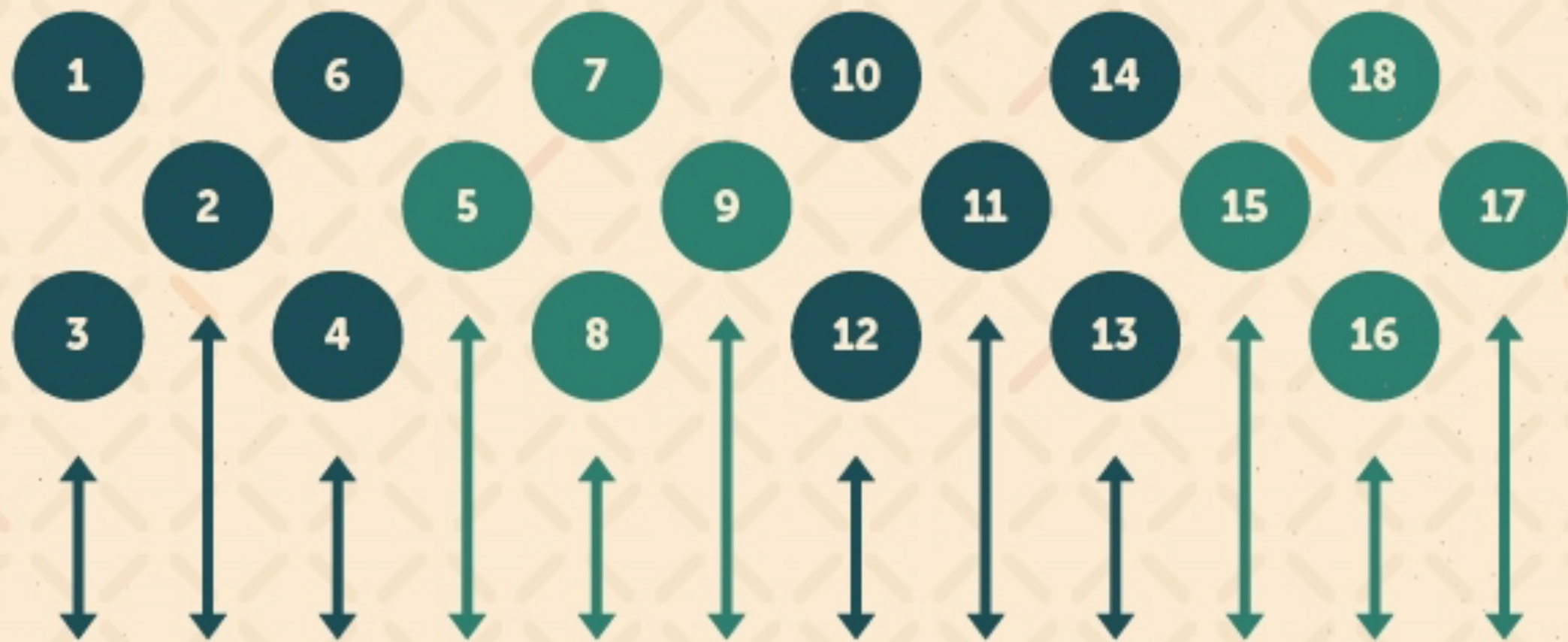
ШАБЛОН ФАСАД

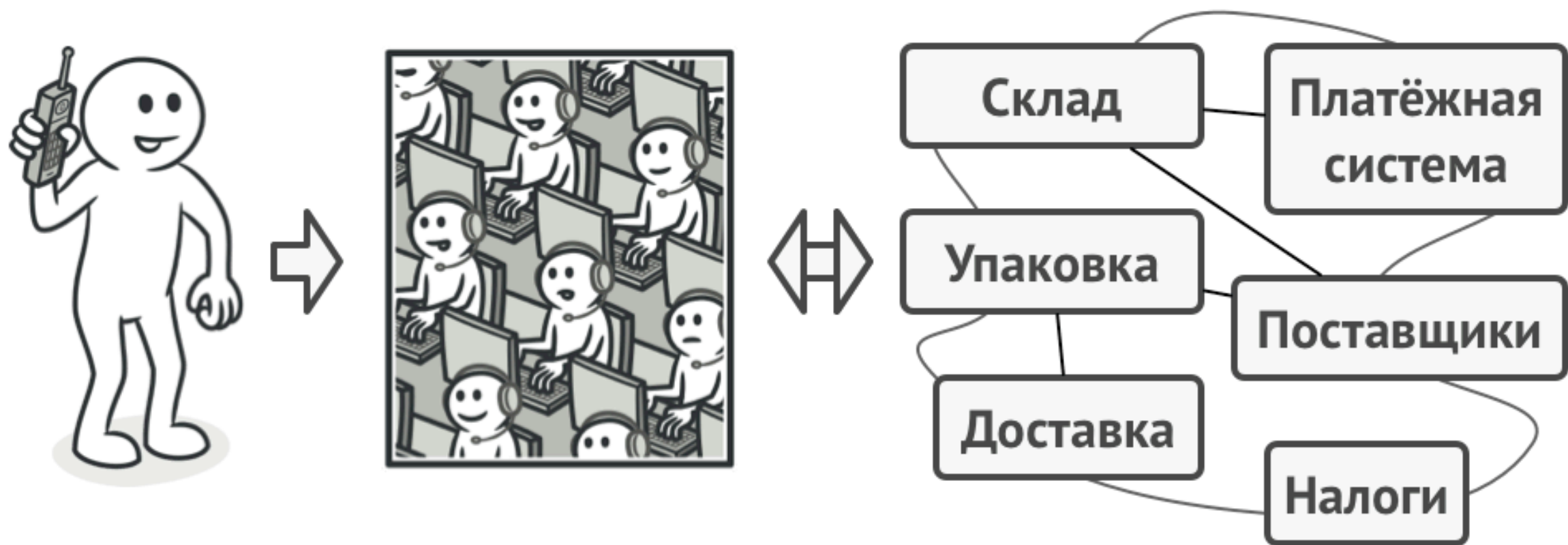
2019 ГОД

**ШАБЛОНЫ ПРОЕКТИРОВАНИЯ — ЭТО РУКОВОДСТВА
ПО РЕШЕНИЮ ПОВТОРЯЮЩИХСЯ ПРОБЛЕМ.**

**ФАСАД — СТРУКТУРНЫЙ ШАБЛОН ПРОЕКТИРОВАНИЯ, ПОЗВОЛЯЮЩИЙ
СКРЫТЬ СЛОЖНОСТЬ СИСТЕМЫ ПУТЁМ СВЕДЕНИЯ ВСЕХ ВОЗМОЖНЫХ
ВНЕШНИХ ВЫЗОВОВ К ОДНОМУ ОБЪЕКТУ, ДЕЛЕГИРУЮЩЕМУ ИХ
СООТВЕТСТВУЮЩИМ ОБЪЕКТАМ СИСТЕМЫ.**

*Структурные шаблоны – шаблоны проектирования, в которых рассматривается вопрос о том, как из классов и объектов образуются более крупные структуры. Т.е. занимаются объединением отдельных классов и объектов в сложные группы.





Пример телефонного заказа.

ПРИМЕНИМОСТЬ

- ▶ Паттерн часто встречается в клиентских приложениях, написанных на Python, которые используют классы-фасады для упрощения работы со сложными библиотеки или API.

ПРИЗНАКИ ПРИМЕНЕНИЯ ПАТТЕРНА

- ▶ Фасад угадывается в классе, который имеет простой интерфейс, но делегирует основную часть работы другим классам. Чаще всего, фасады сами следят за жизненным циклом объектов сложной системы.

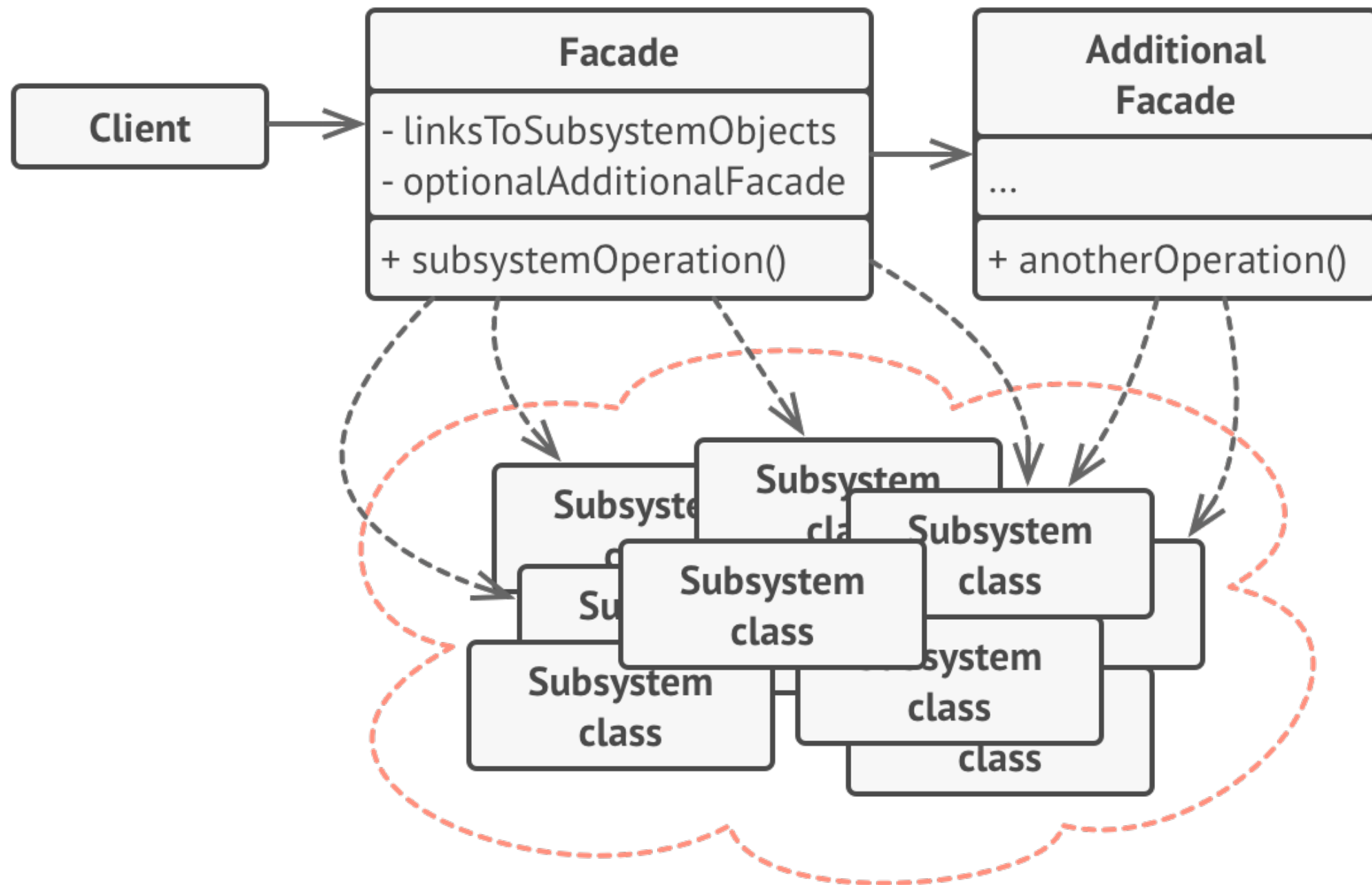
ПРОБЛЕМА

- ▶ Вашему коду приходится работать с большим количеством объектов некой сложной библиотеки или фреймворка. Вы должны самостоятельно инициализировать эти объекты, следить за правильным порядком зависимостей и так далее.
- ▶ В результате бизнес-логика ваших классов тесно переплетается с деталями реализации сторонних классов. Такой код довольно сложно понимать и поддерживать.

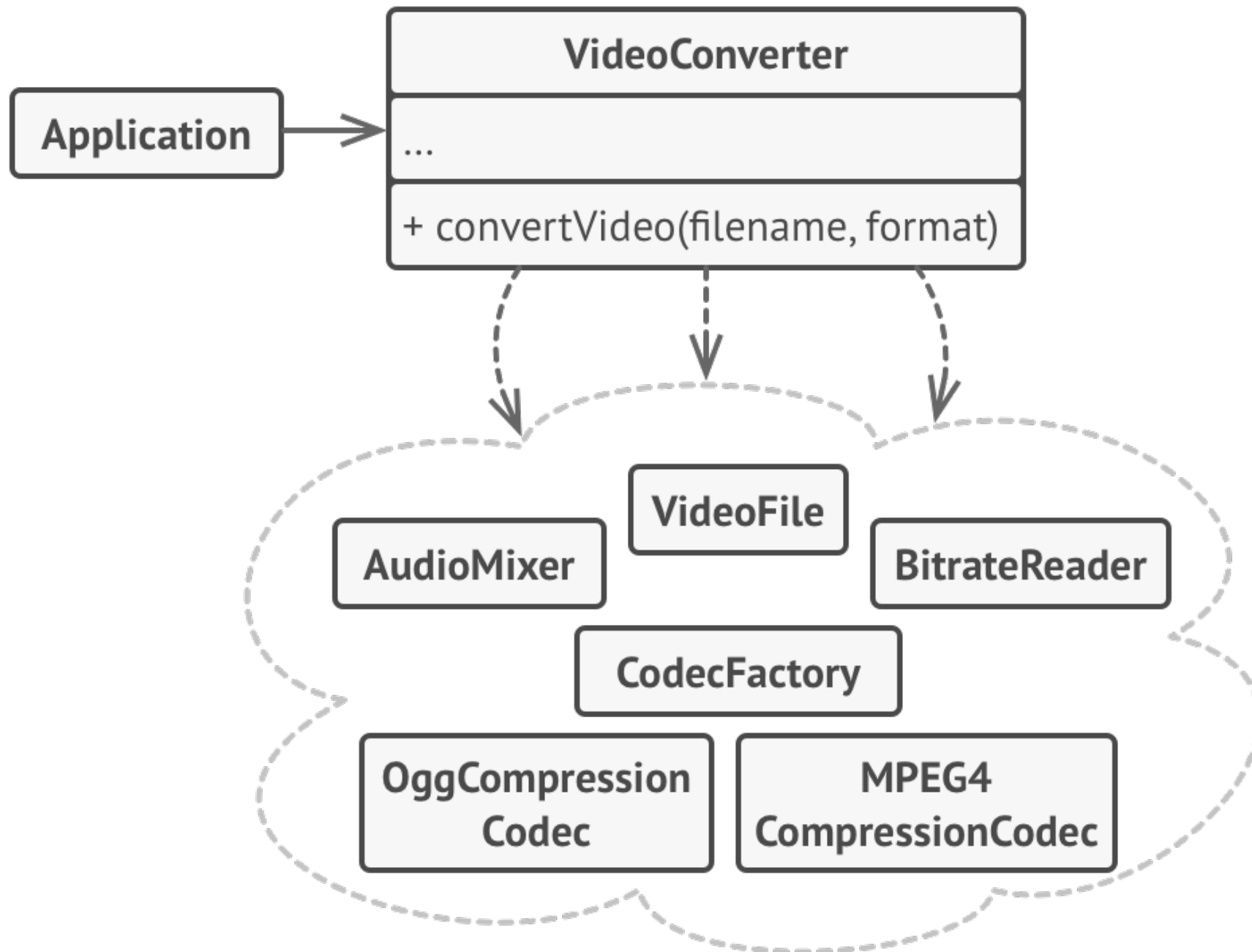
РЕШЕНИЕ

- ▶ Фасад – это простой интерфейс для работы со сложной подсистемой, содержащей множество классов. Фасад может иметь урезанный интерфейс, не имеющий 100% функциональности, которой можно достичь, используя сложную подсистему напрямую.
- ▶ Фасад полезен, если вы используете какую-то сложную библиотеку со множеством подвижных частей, но вам нужна только часть её возможностей.
- ▶ К примеру, программа, заливающая видео котиков в социальные сети, может использовать профессиональную библиотеку сжатия видео. Но все, что нужно клиентскому коду этой программы – простой метод `#encode(filename, format)#`. Создав класс с таким методом, вы реализуете свой первый фасад.

СТРУКТУРА



ПРИМЕР



```
1  class VideoFile
2  // ...
3  class OggCompressionCodec
4  // ...
5  class MPEG4CompressionCodec
6  // ...
7  class CodecFactory
8  // ...
9  class BitrateReader
10 // ...
11 class AudioMixer
12 // ...
13
14 class VideoConverter is
15     method convert(filename, format):File is
16         file = new VideoFile(filename)
17         sourceCodec = new CodecFactory.extract(file)
18         if (format == "mp4")
19             destinationCodec = new MPEG4CompressionCodec()
20         else
21             destinationCodec = new OggCompressionCodec()
22         buffer = BitrateReader.read(filename, sourceCodec)
23         result = BitrateReader.convert(buffer, destinationCodec)
24         result = (new AudioMixer()).fix(result)
25         return new File(result)
26
27 class Application is
28     method main() is
29         convertor = new VideoConverter()
30         mp4 = convertor.convert("youtubevideo.ogg", "mp4")
31         mp4.save()
32
```

ОТНОШЕНИЯ С ДРУГИМИ ПАТТЕРНАМИ

- ▶ **Фасад** задаёт новый интерфейс, тогда как **Адаптер** повторно использует старый. *Адаптер* оборачивает только один класс, а *Фасад* оборачивает целую подсистему. #Кроме того, *Адаптер* позволяет двум существующим интерфейсам работать сообща, вместо того, чтобы задать полностью новый.
- ▶ **Абстрактная фабрика** может быть использована вместо **Фасада** для того, чтобы скрыть платформно-зависимые классы.
- ▶ **Посредник** и **Фасад** похожи тем, что пытаются организовать работу множества существующих классов.
- ▶ **Фасад** можно сделать **Одиночкой**, так как обычно нужен только один объект-фасад.
- ▶ **Фасад** похож на **Заместитель** тем, что замещает сложную подсистему и может сам её инициализировать. Но в отличие от *Фасада*, *Заместитель* имеет тот же интерфейс, что его служебный объект, благодаря чему их можно взаимозаменять.

СПАСИБО ЗА ВНИМАНИЕ!