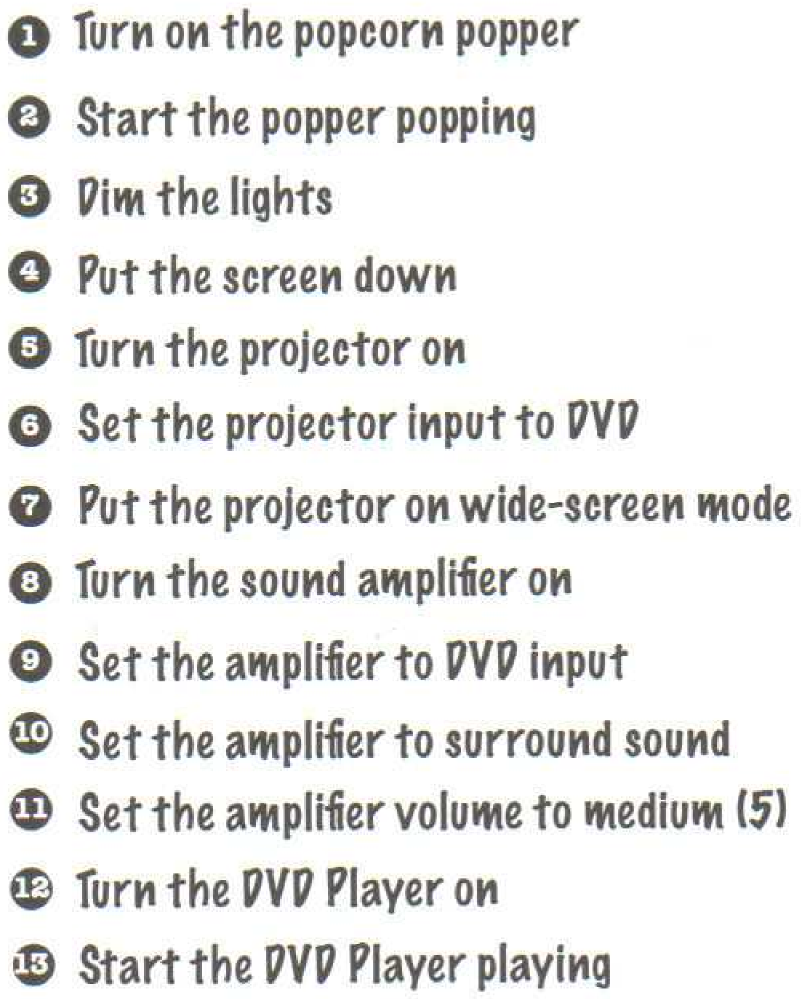
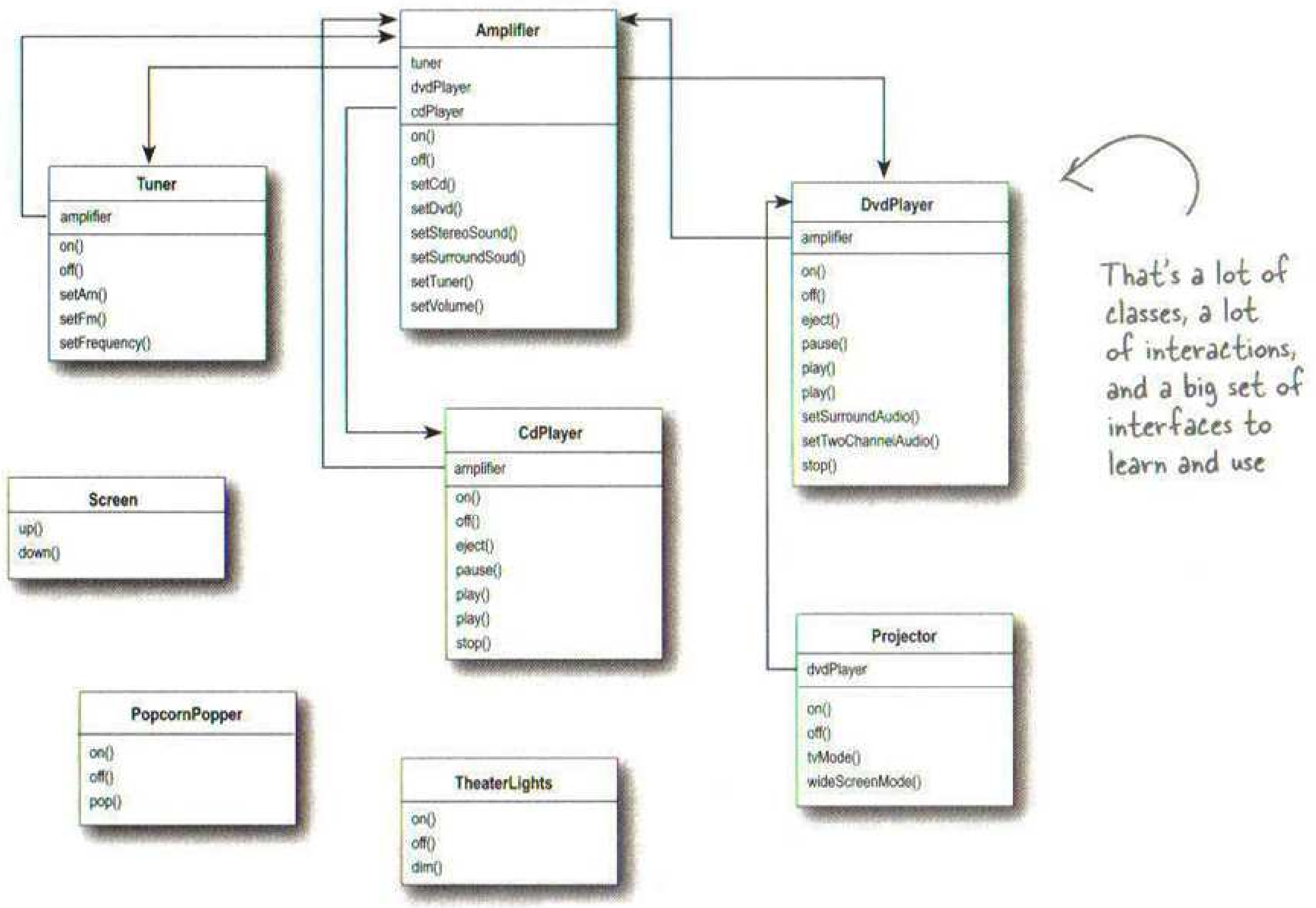
Tecniche di Progettazione: Design Patterns

GoF: Façade

Watching the movie the hard way….

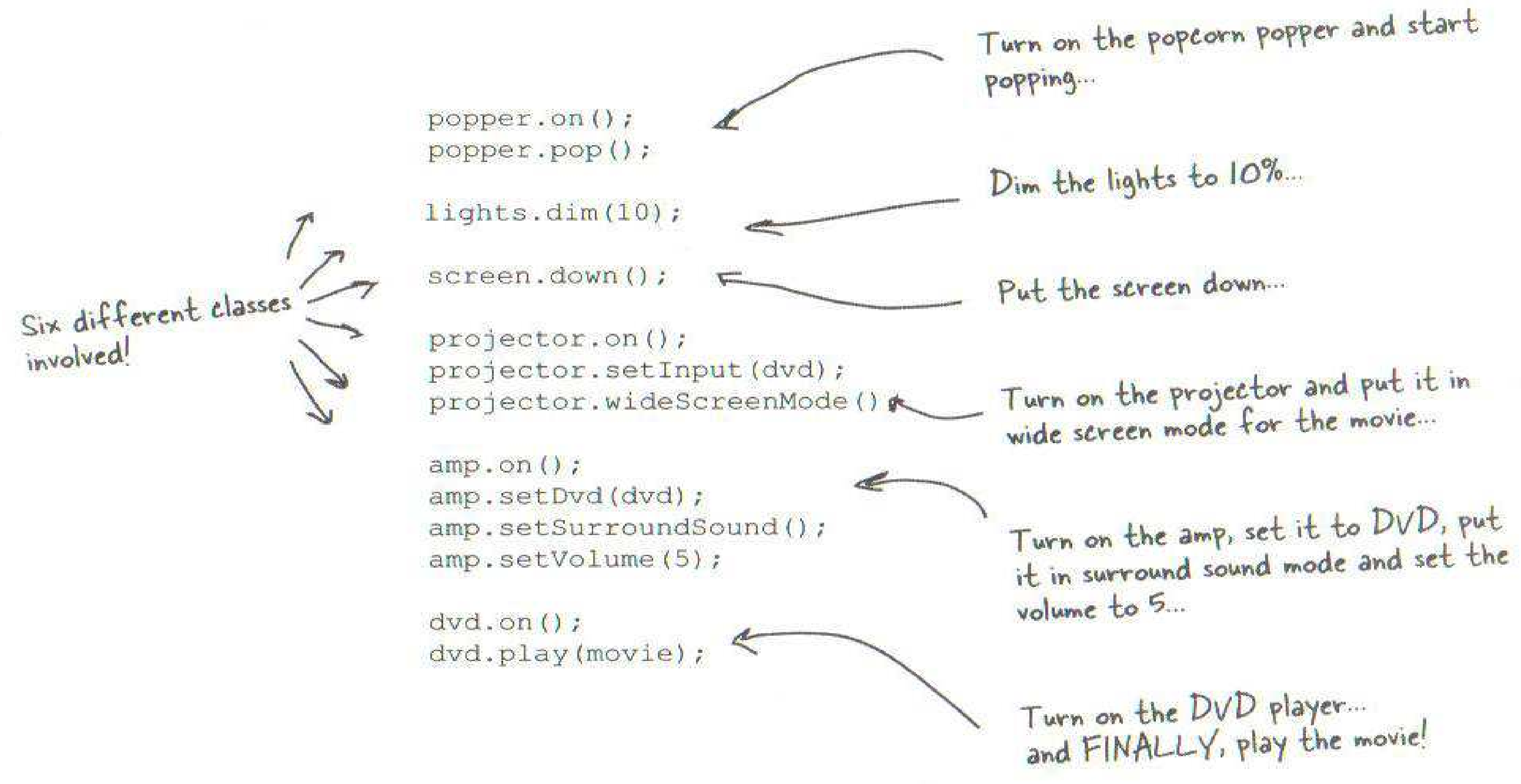


1. Turn on the popcorn popper
2. Start the popper popping
3. Dim the lights
4. Put the screen down
5. Turn the projector on
6. Set the projector input to DVD
7. Put the projector on wide-screen mode
8. Turn the sound amplifier on
9. Set the amplifier to DVD input
10. Set the amplifier to surround sound
11. Set the amplifier volume to medium (5)
12. Turn the DVD Player on
13. Start the DVD Player playing



Amplifier
tuner
dvdPlayer
cdPlayer
on()
off()
setCd()
setDvd()
setStereoSound()
setSurroundSoud()
setTuner()
setVolume()
Tuner
amplifier
on()
off()
setAm()
setFm()
setFrequency()
DvdPlayer
amplifier
on()
off()
eject()
pause()
play()
play()
setSurroundAudio()
setTwoChannelAudio()
stop()
CdPlayer
amplifier
on()
off()
eject()
pause()
play()
play()
stop()
Screen
up()
down()
Projector
dvdPlayer
on()
off()
tvMode()
wideScreenMode()
PopcornPopper
on()
off()
pop()
TheaterLights
on()
off()
dim()
That's a lot of classes, a lot of interactions, and a big set of interfaces to learn and use

What needs to be done to watch a movie….



Turn on the popcorn popper and start popping...

Dim the lights to 10%...

Put the screen down...

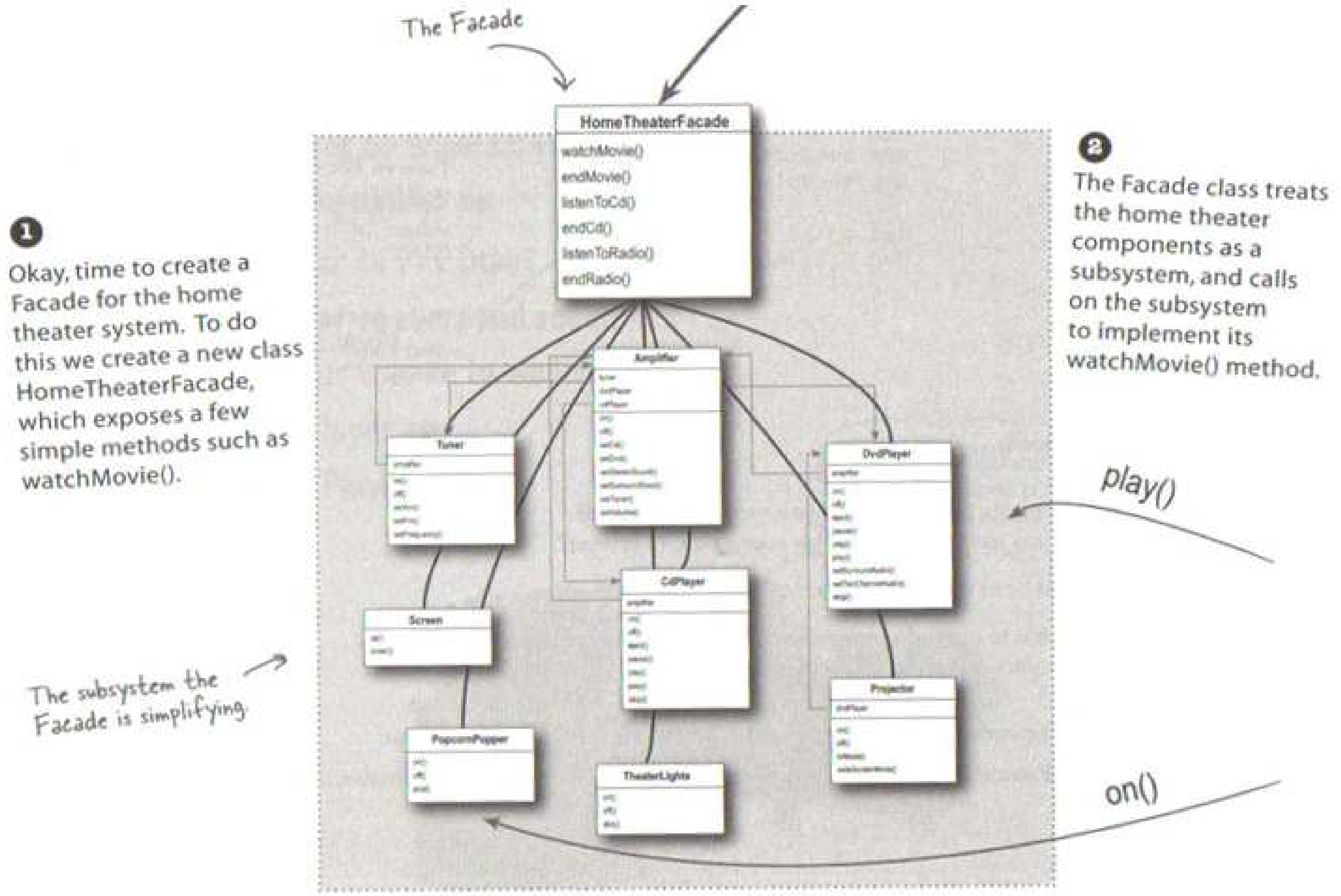
Turn on the projector and put it in wide screen mode for the movie...

Turn on the amp, set it to DVD, put it in surround sound mode and set the volume to 5...

Turn on the DVD player... and FINALLY, play the movie!

Six different classes involved!

```
popper.on();
popper.pop();

lights.dim(10);

screen.down();

projector.on();
projector.setInput(dvd);
projector.wideScreenMode();

amp.on();
amp.setDvd(dvd);
amp.setSurroundSound();
amp.setVolume(5);

dvd.on();
dvd.play(movie);
```



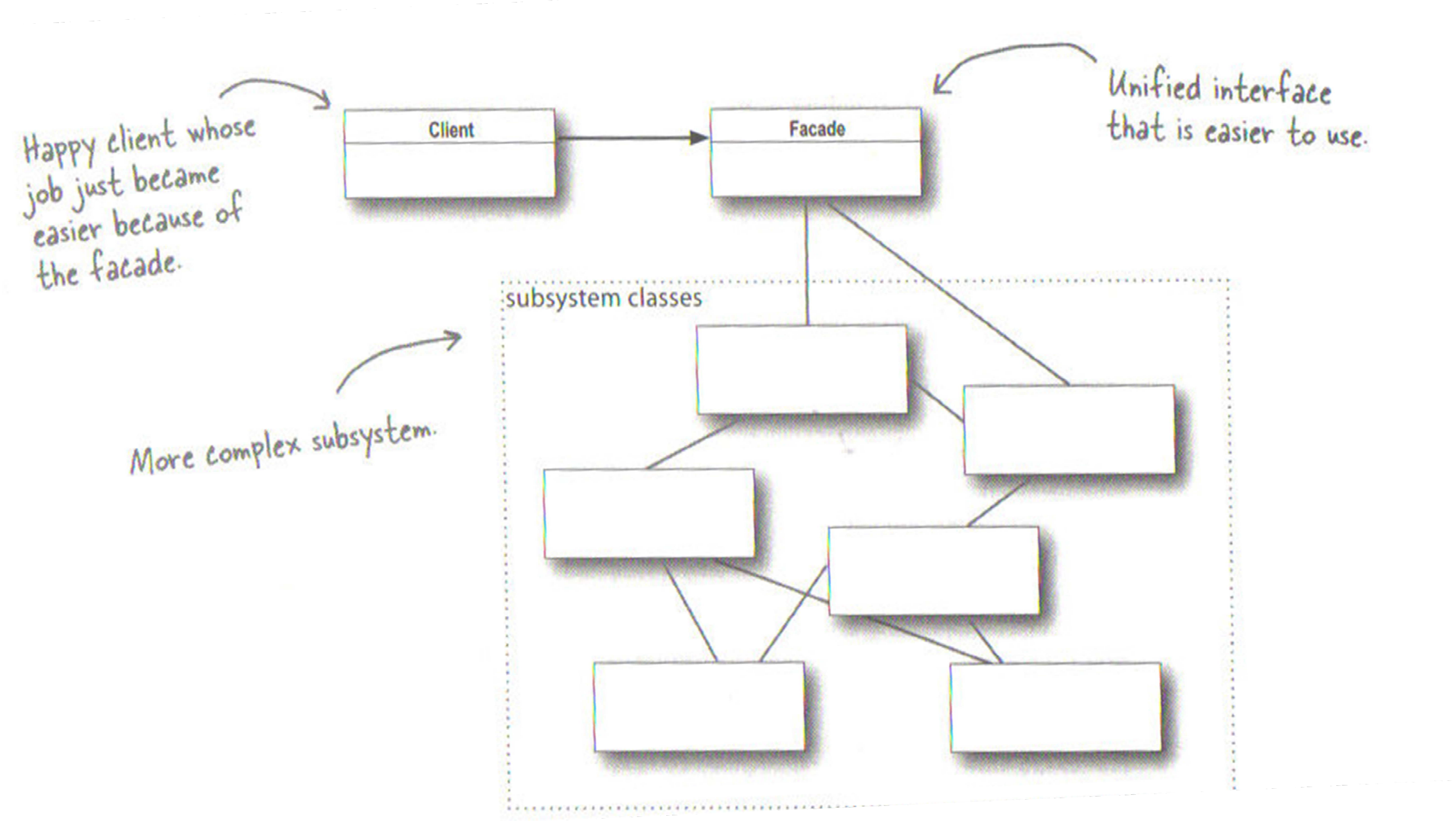
1 Okay, time to create a Facade for the home theater system. To do this we create a new class HomeTheaterFacade, which exposes a few simple methods such as watchMovie().

2 The Facade class treats the home theater components as a subsystem, and calls on the subsystem to implement its watchMovie() method.

Façade Pattern defined

The Façade Pattern provides a unified interface to a set of interfaces in a subsystem. Façade defines a higher level interface that makes the subsystem easier to use.

Façade pattern – Class Diagram

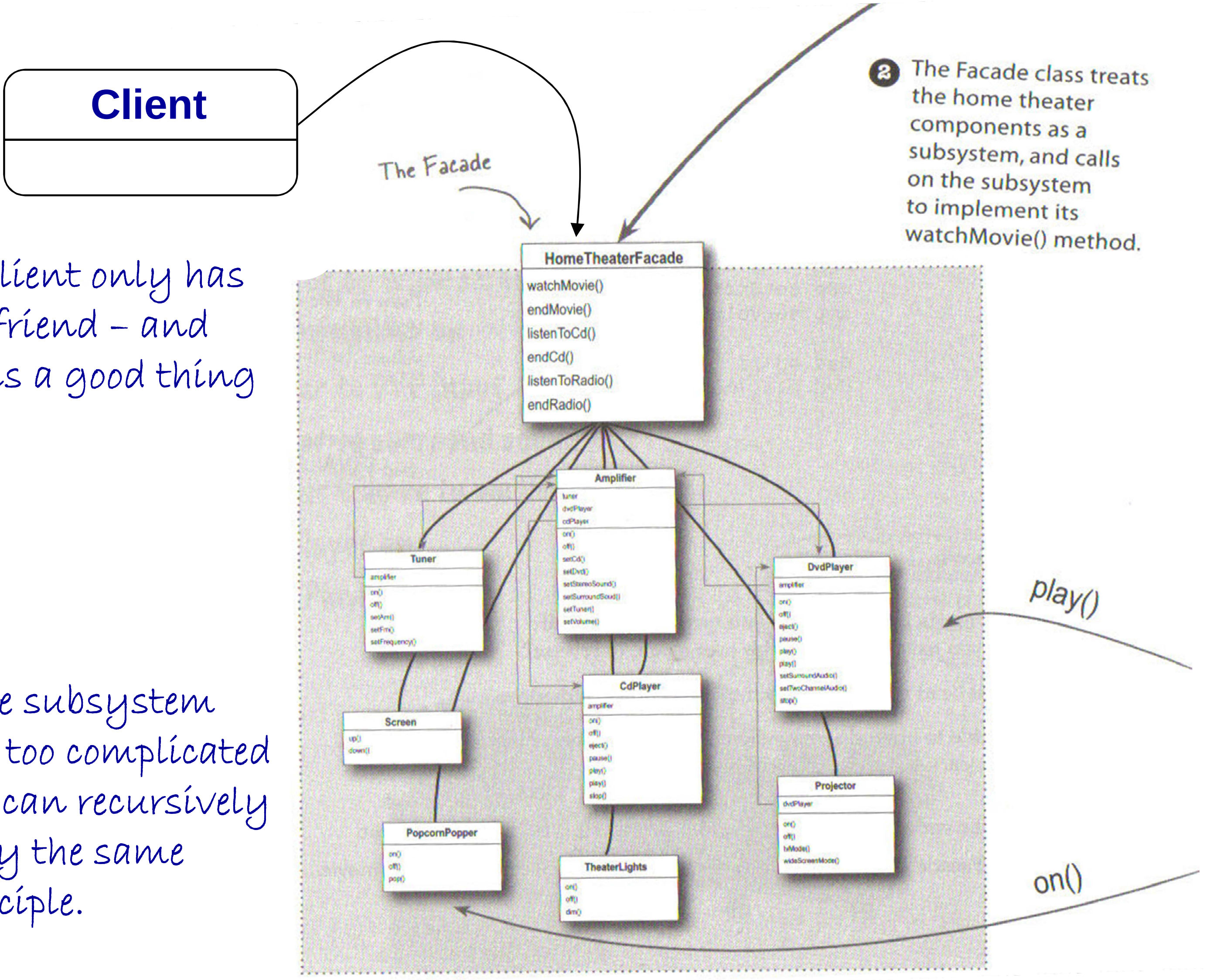

Happy client whose job just became easier because of the facade.
Client
Facade
Unified interface that is easier to use.
subsystem classes
More complex subsystem.

Design Principle

Principle of Least Knowledge

talk only to your immediate friends

Basically this says minimize your dependencies

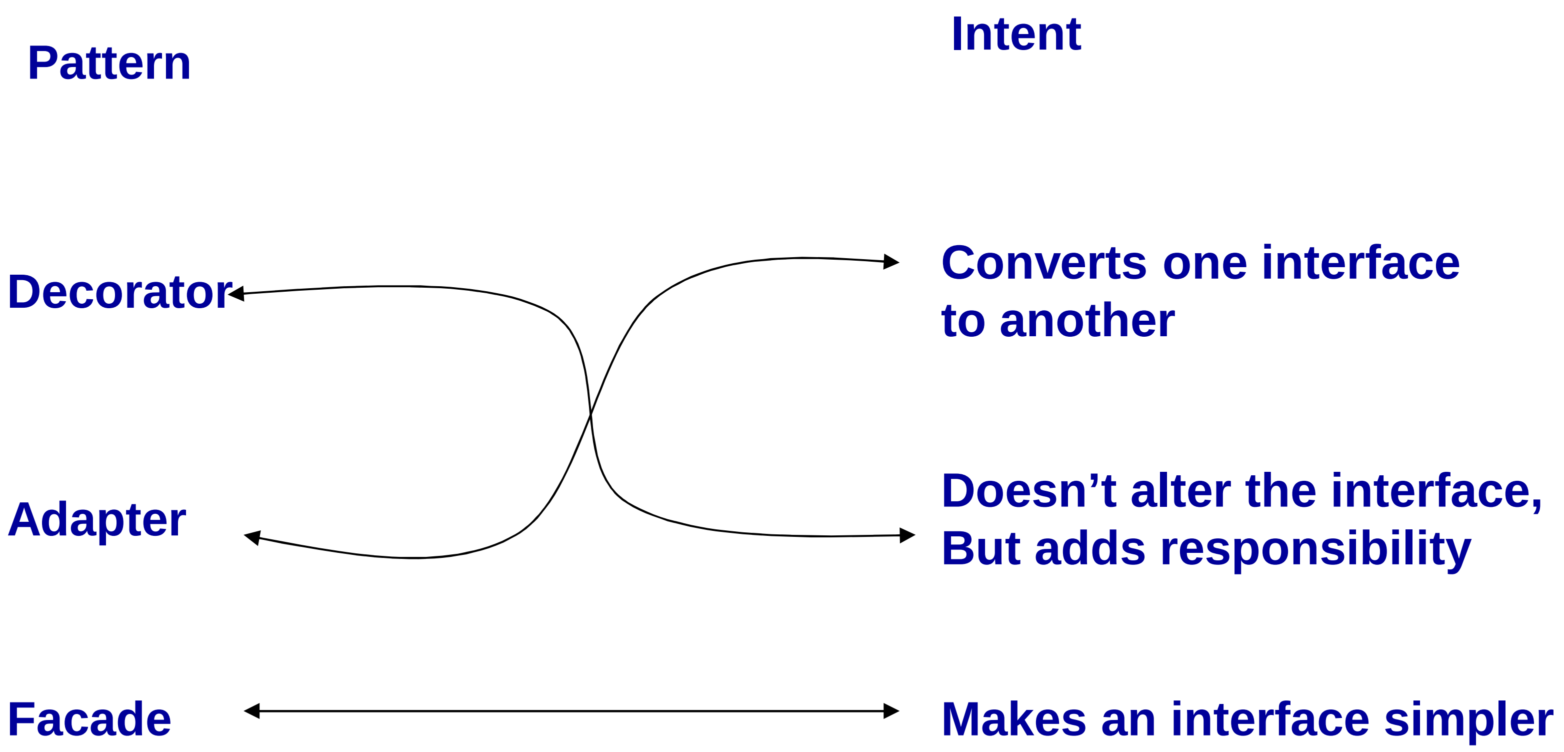


❷ The Facade class treats the home theater components as a subsystem, and calls on the subsystem to implement its watchMovie() method.

The client only has one friend – and that is a good thing

If the subsystem gets too complicated you can recursively apply the same principle.

A little comparison


Pattern
Intent
Decorator
Converts one interface to another
Adapter
Doesn’t alter the interface, But adds responsibility
Facade
Makes an interface simpler