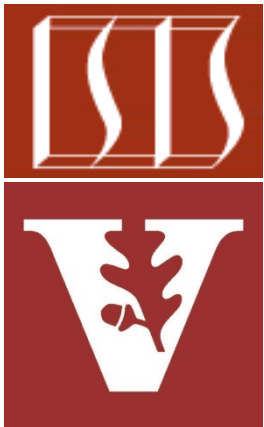


# Analyzing the Filter Class Hierarchy



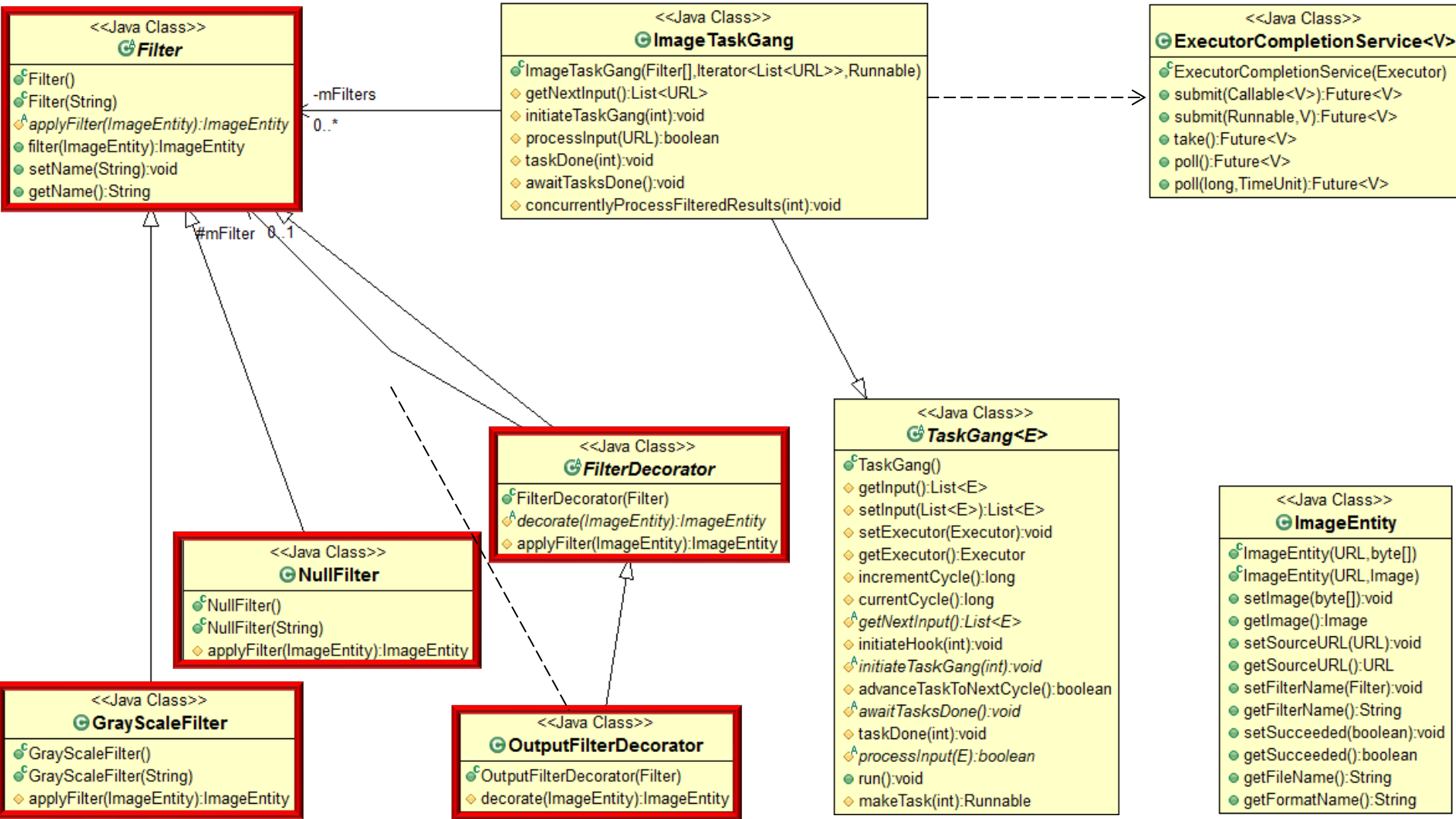
**Douglas C. Schmidt**  
**[d.schmidt@vanderbilt.edu](mailto:d.schmidt@vanderbilt.edu)**  
**[www.dre.vanderbilt.edu/~schmidt](http://www.dre.vanderbilt.edu/~schmidt)**

**Institute for Software  
Integrated Systems  
Vanderbilt University  
Nashville, Tennessee, USA**



# Learning Objectives in this Part of the Lesson

- Understand the pattern-oriented software implementation of the Filter class hierarchy



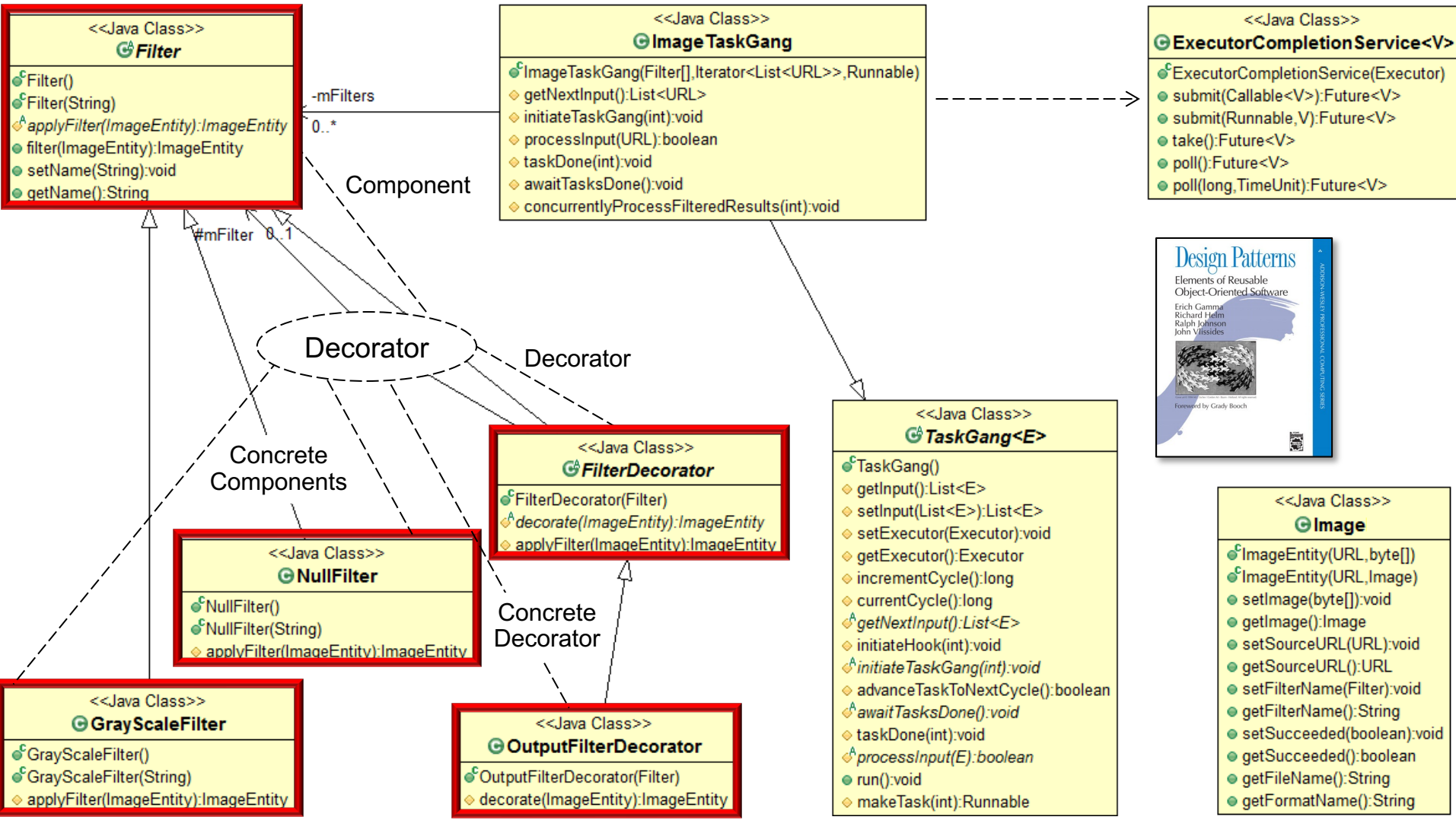
See <ImageTaskGangApplication/app/src/main/java/example/imagetaskgang/filters>

---

# Analysis of the Filter

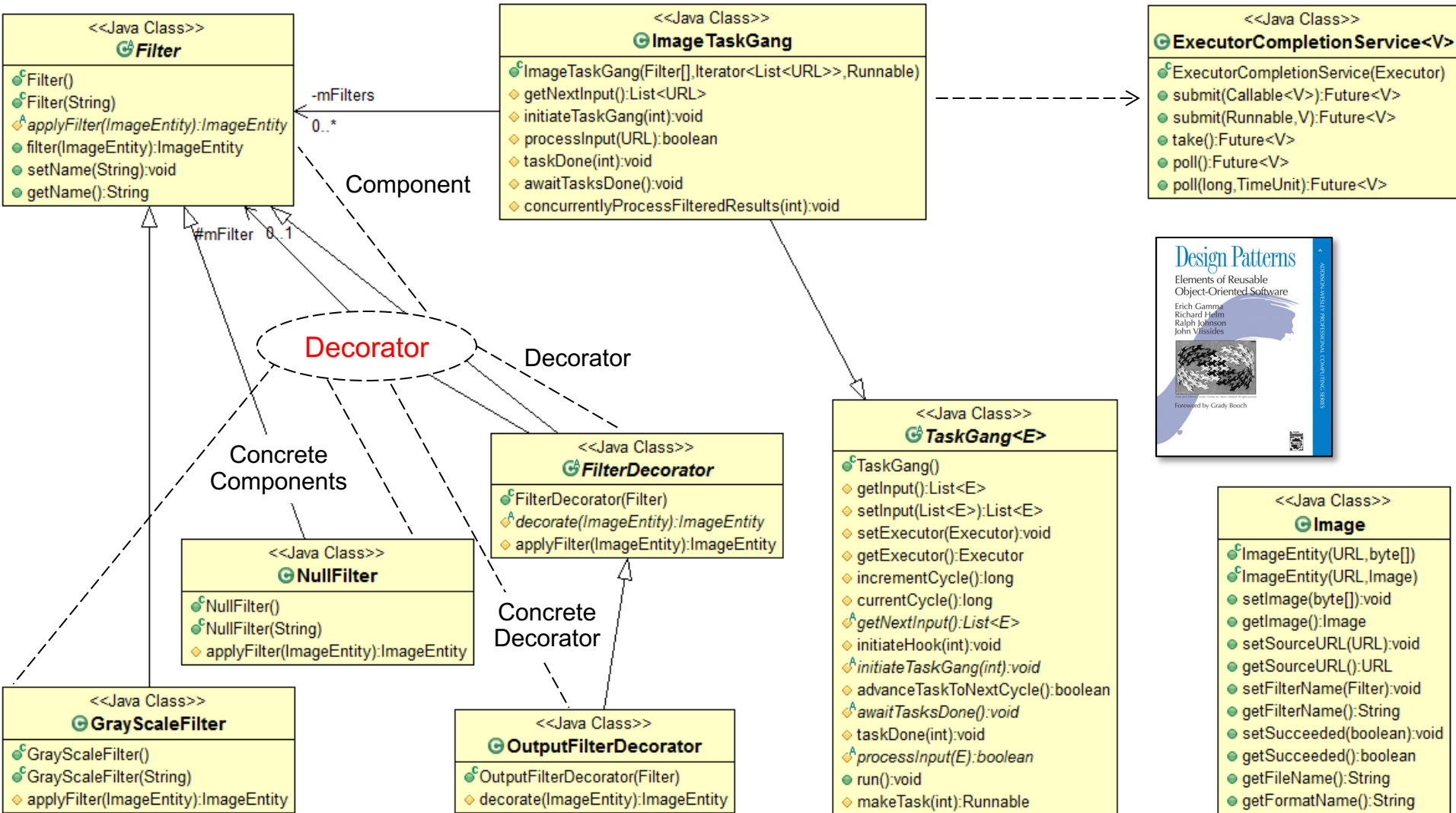
## Class Hierarchy Source Code

# Analysis of the Filter Hierarchy Classes



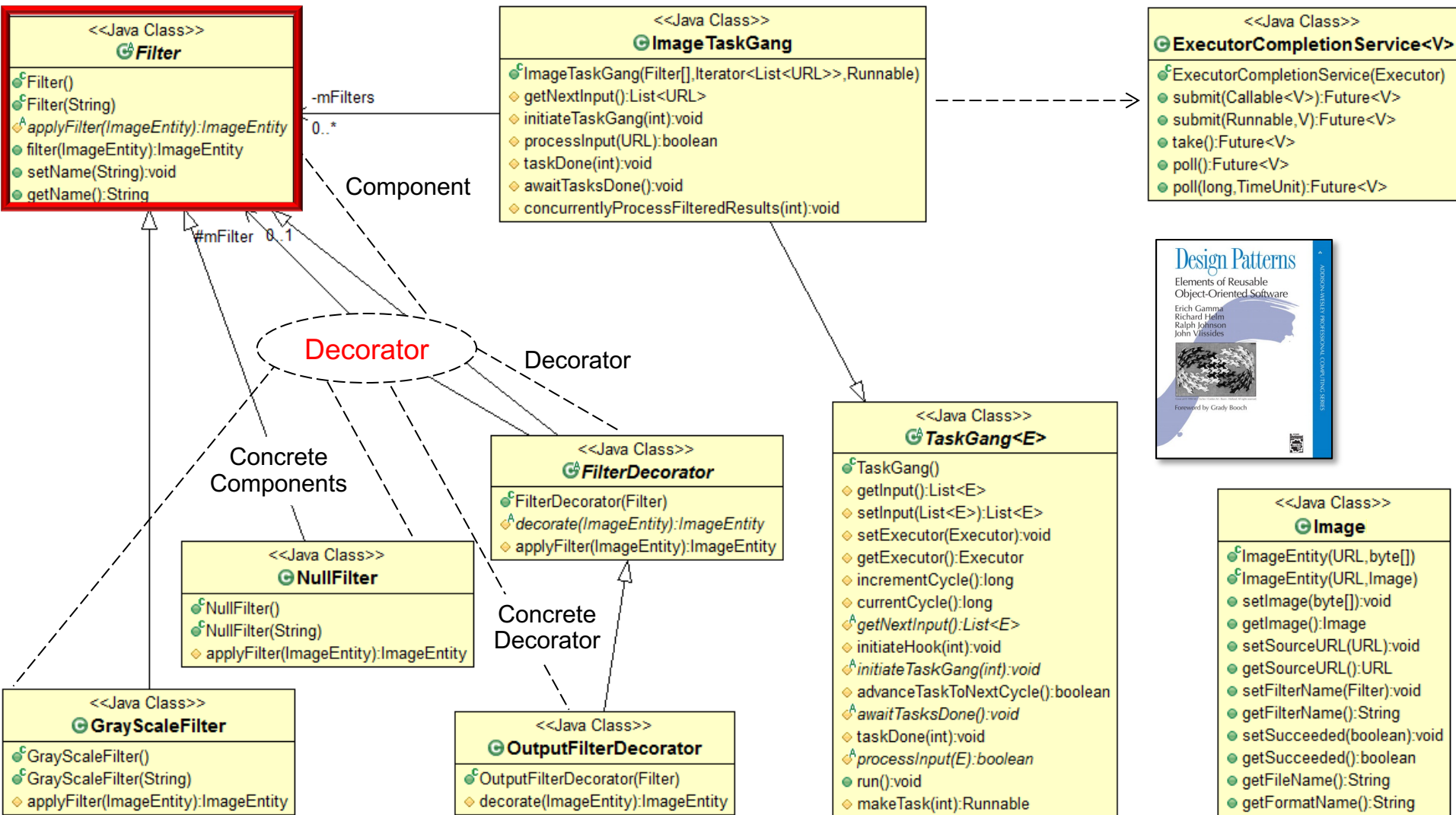
Provides the means to apply a series of filters to process & store each downloaded image

# Analysis of the Filter Hierarchy Classes

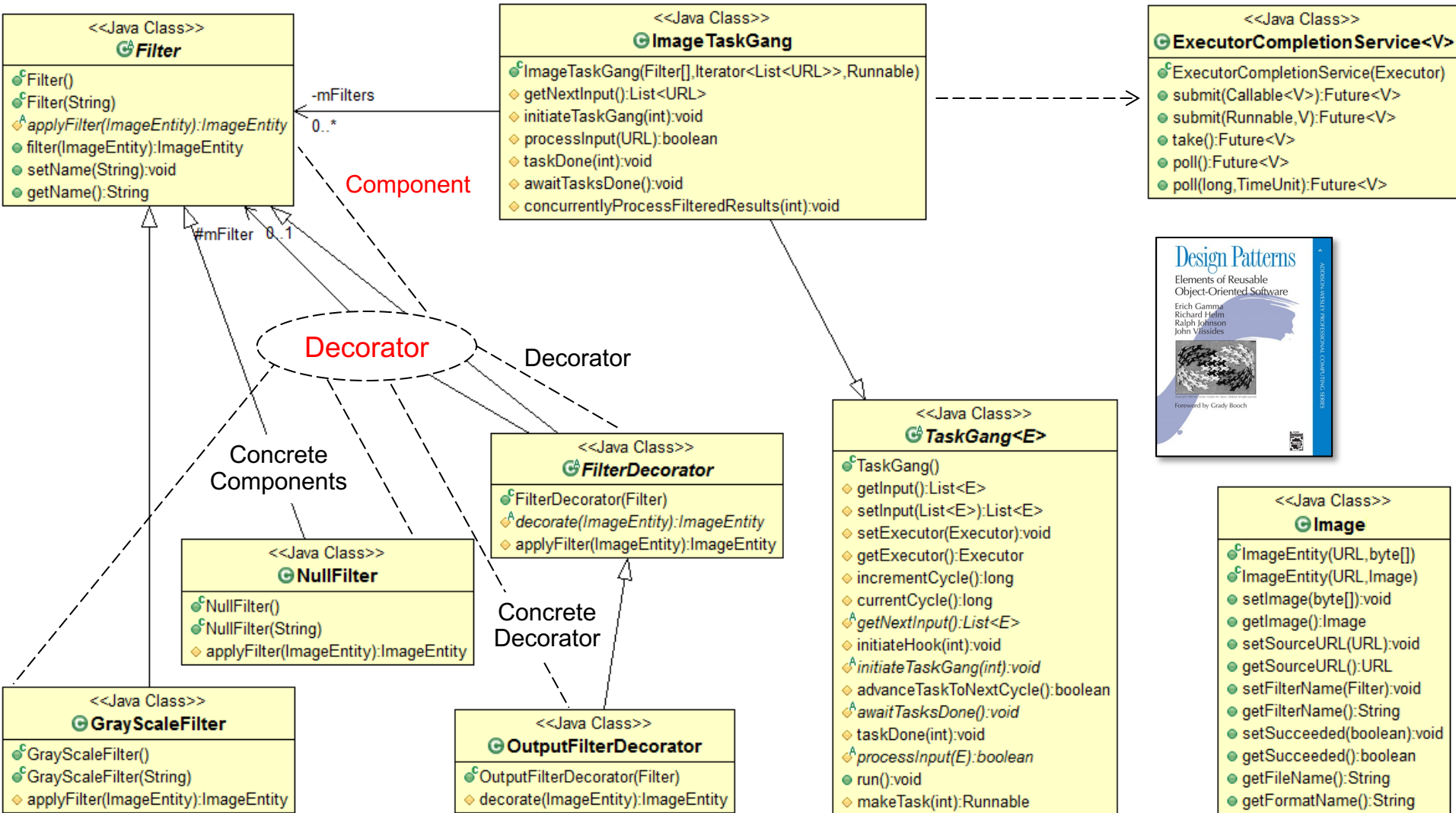


*Decorator* allows behavior to be added to an individual object transparently, without affecting the behavior of other object's from the same class

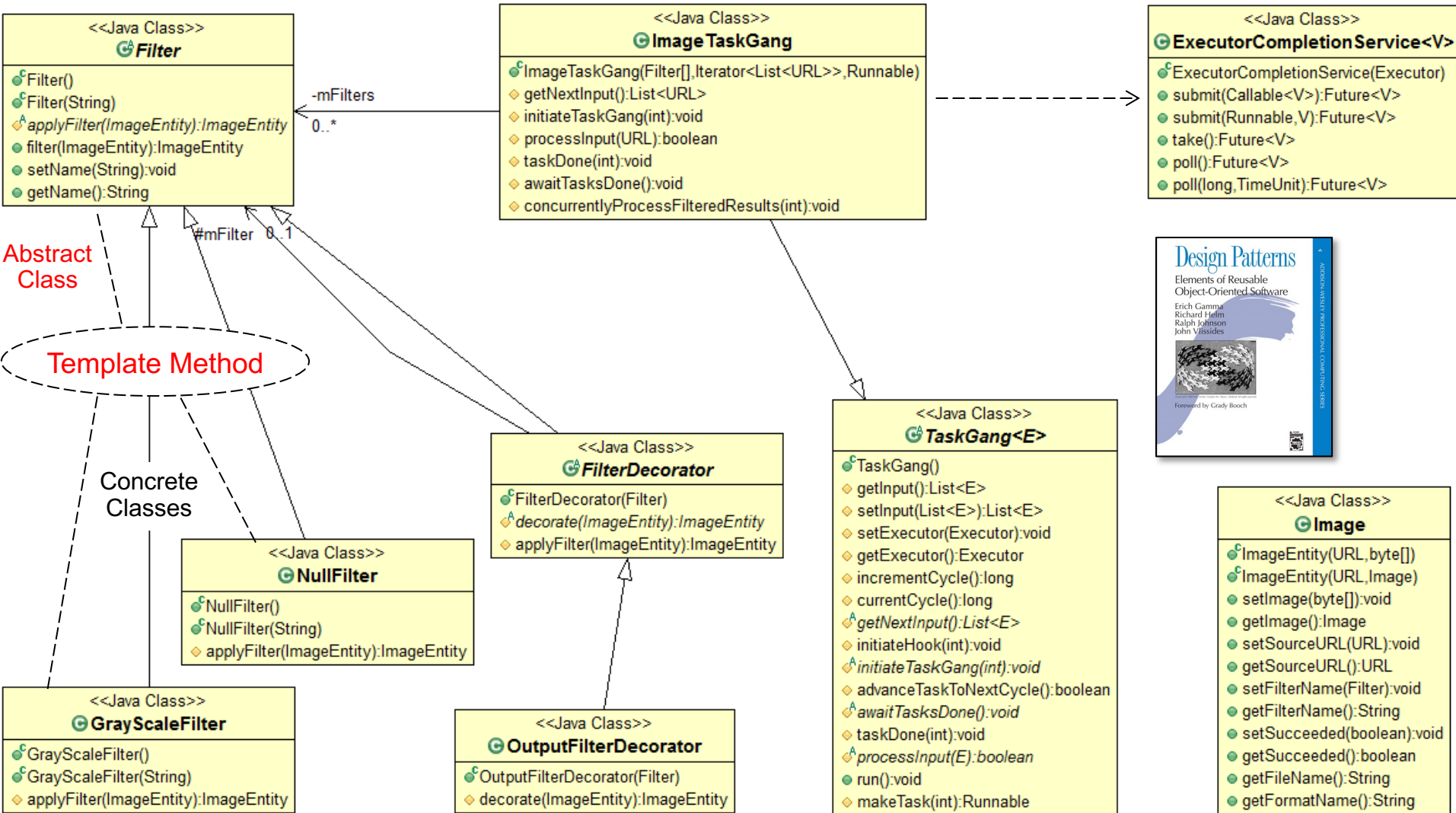
# Analysis of the Filter Class



# Analysis of the Filter Class



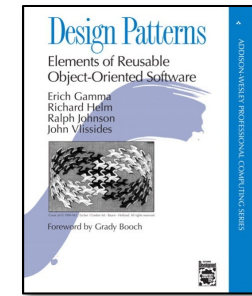
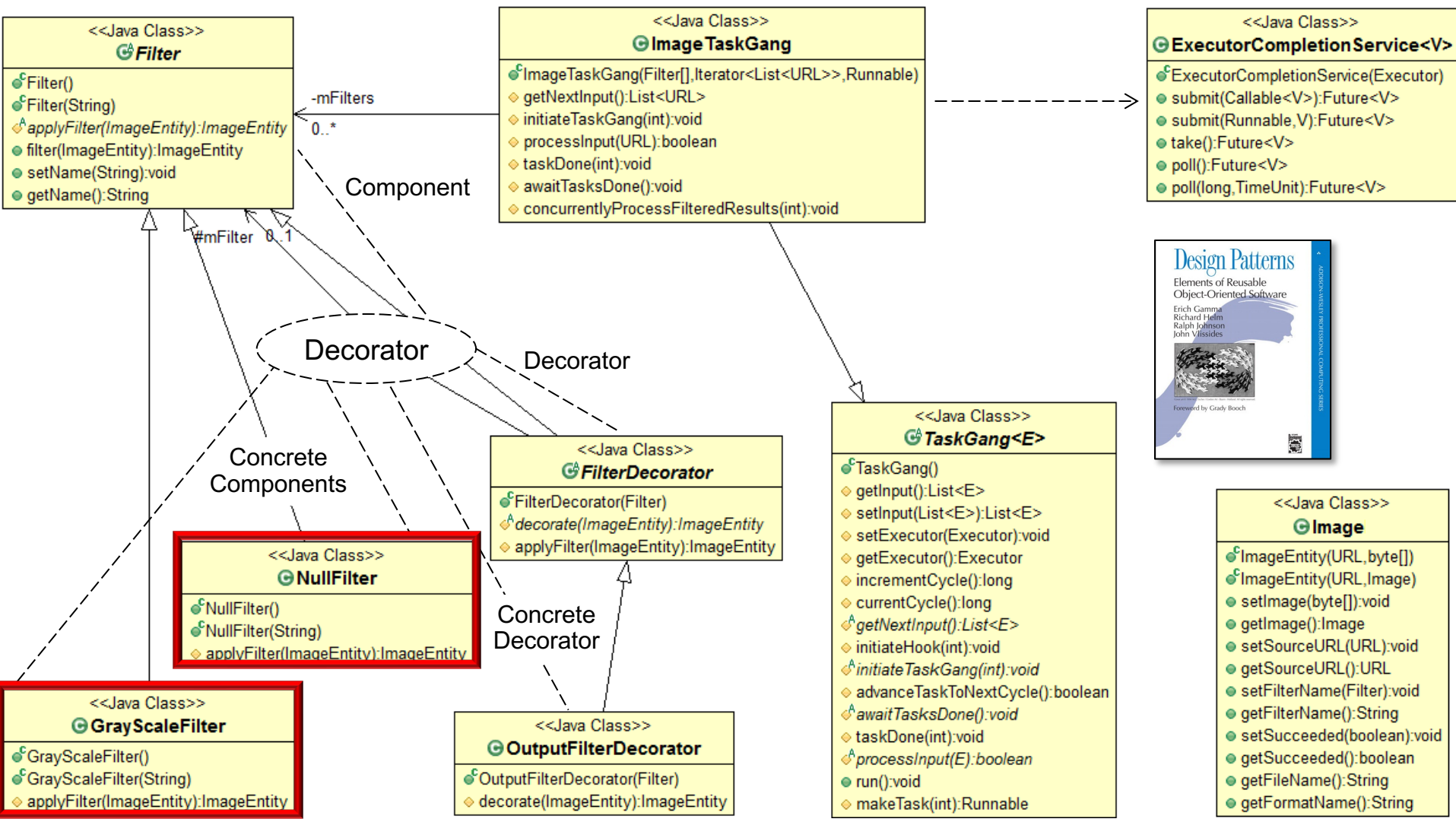
# Analysis of the Filter Class



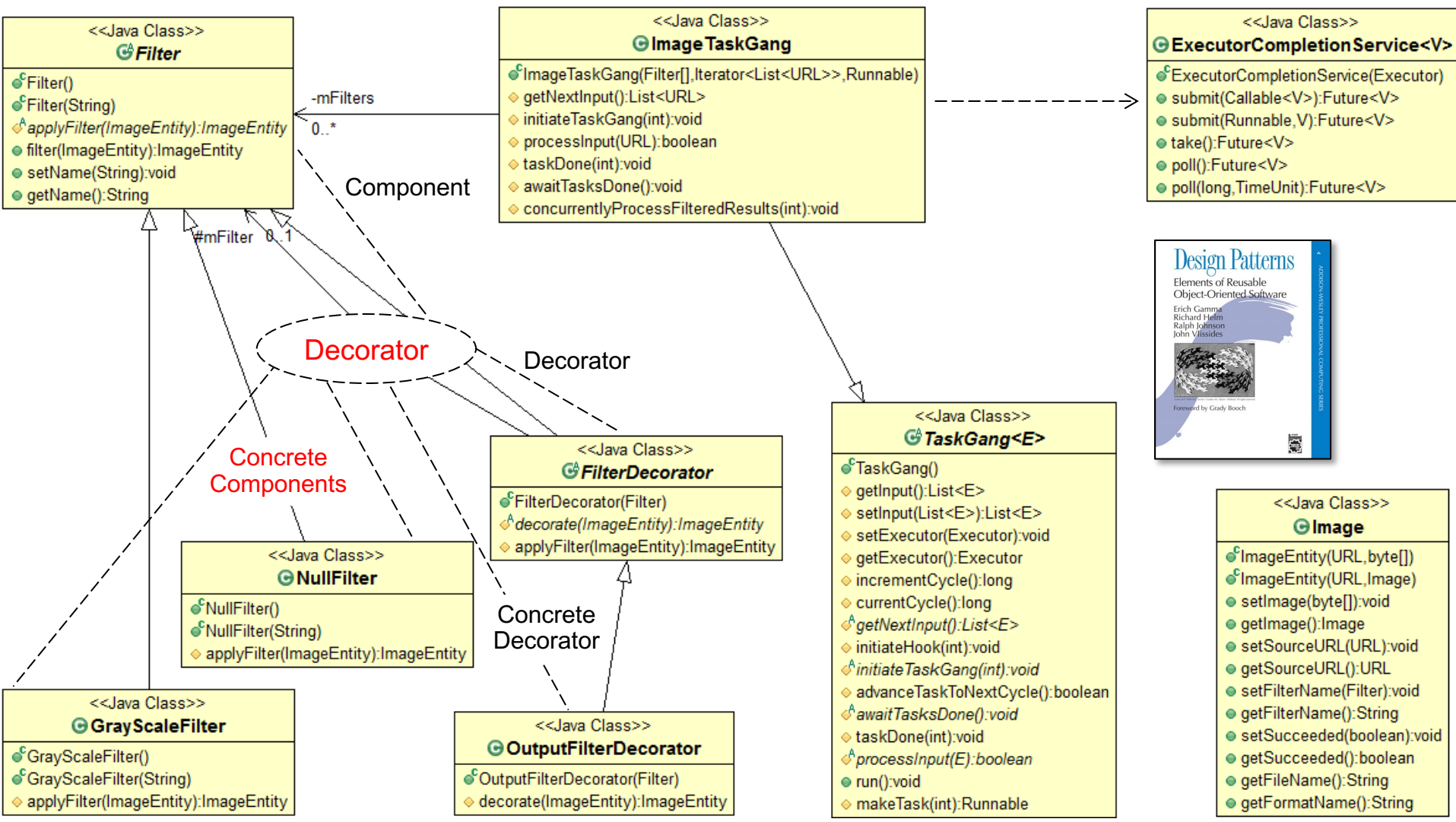
*Template Method* defines the skeleton of an algorithm in a method, deferring certain steps to subclass methods



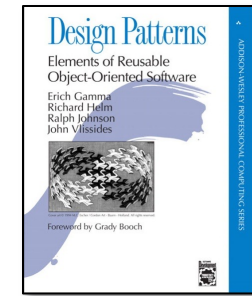
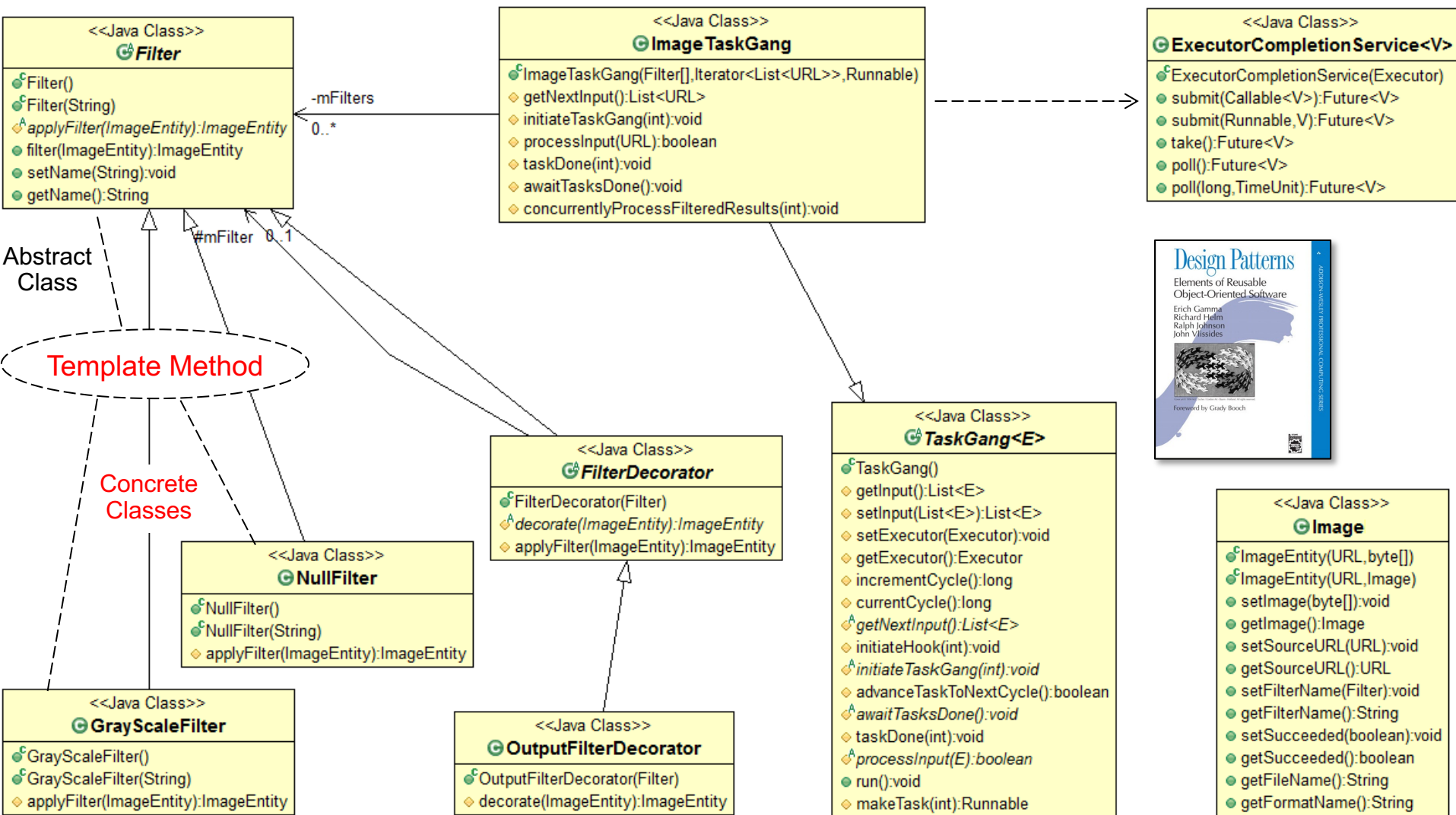
# Analysis of the GrayScaleFilter & NullFilter Classes



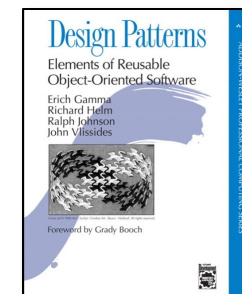
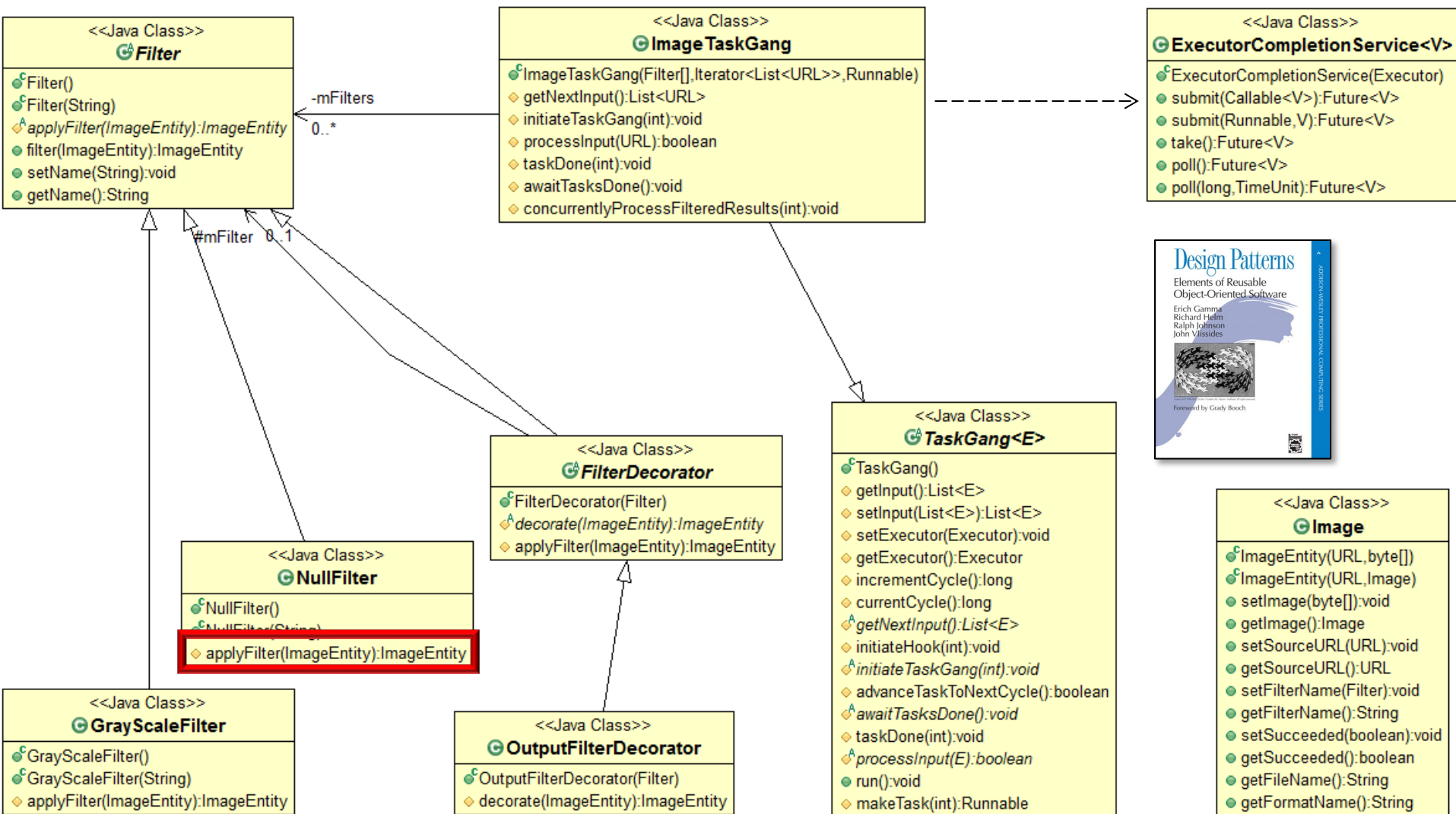
# Analysis of the GrayScaleFilter & NullFilter Classes



# Analysis of the GrayScaleFilter & NullFilter Classes

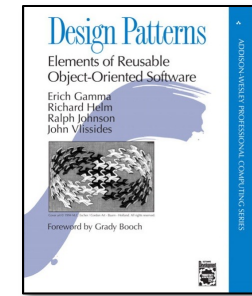
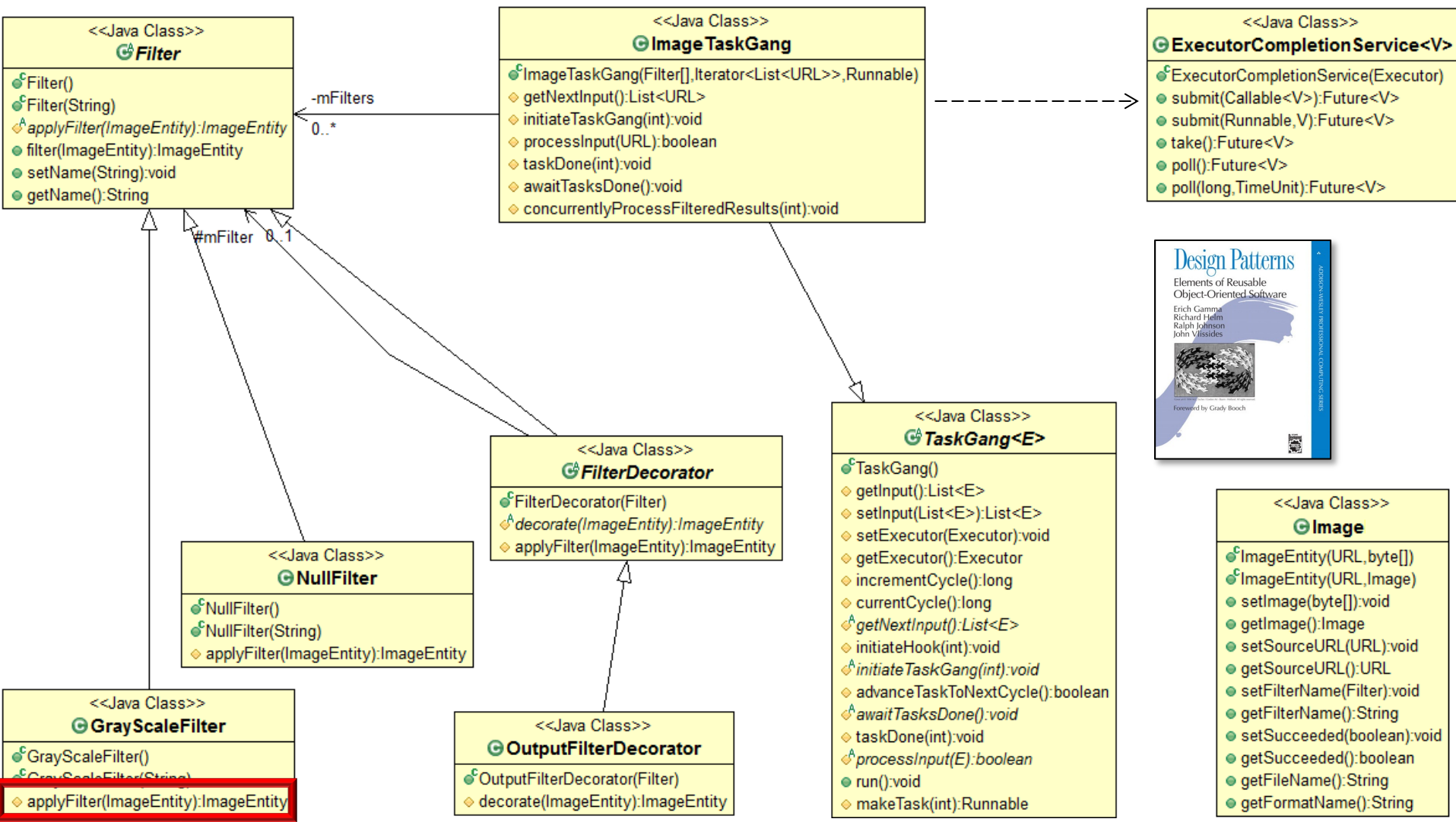


# Analysis of the GrayScaleFilter & NullFilter Classes



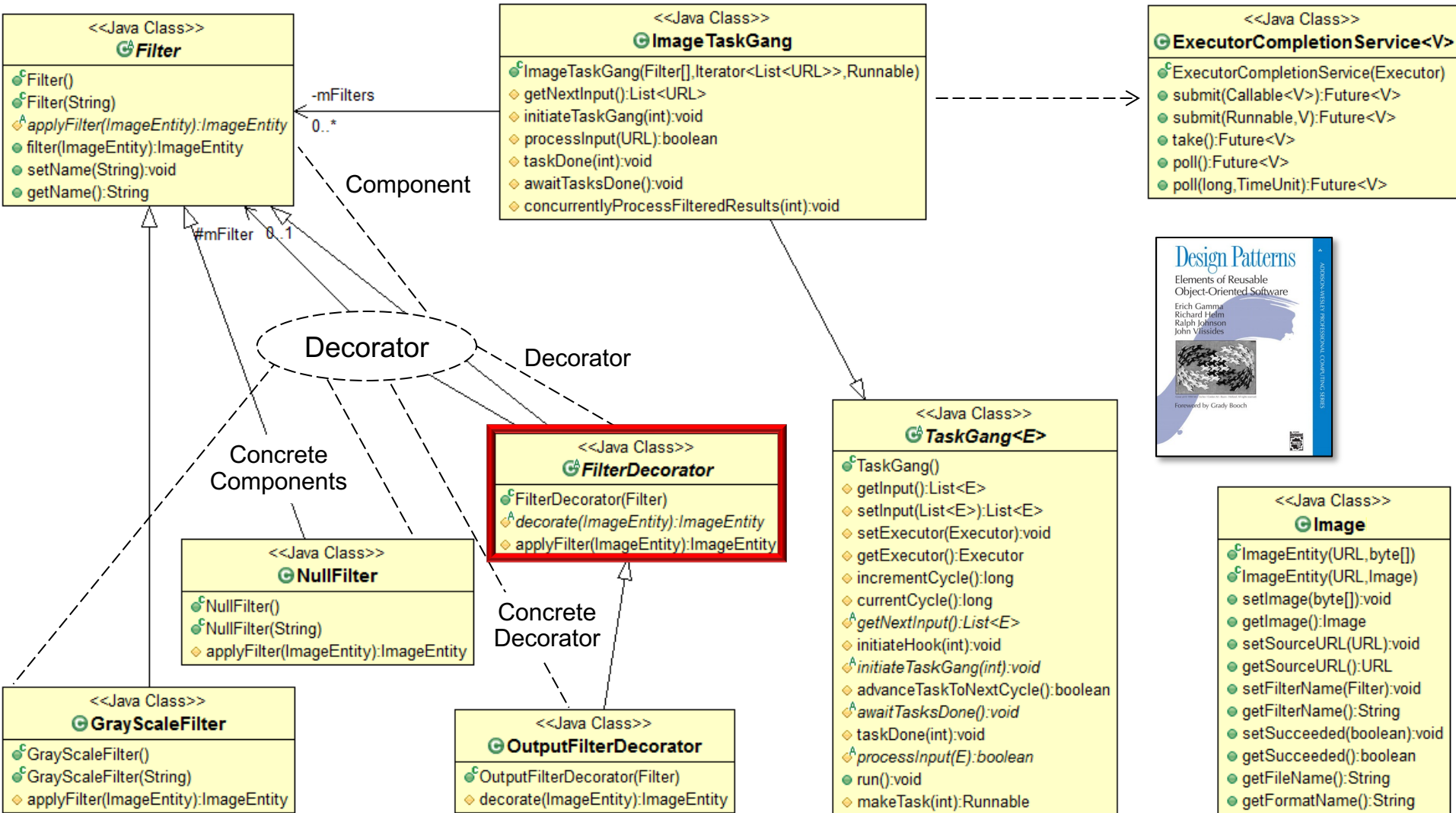
See [en.wikipedia.org/wiki/NOP\\_\(code\)](http://en.wikipedia.org/wiki/NOP_(code))

# Analysis of the GrayScaleFilter & NullFilter Classes

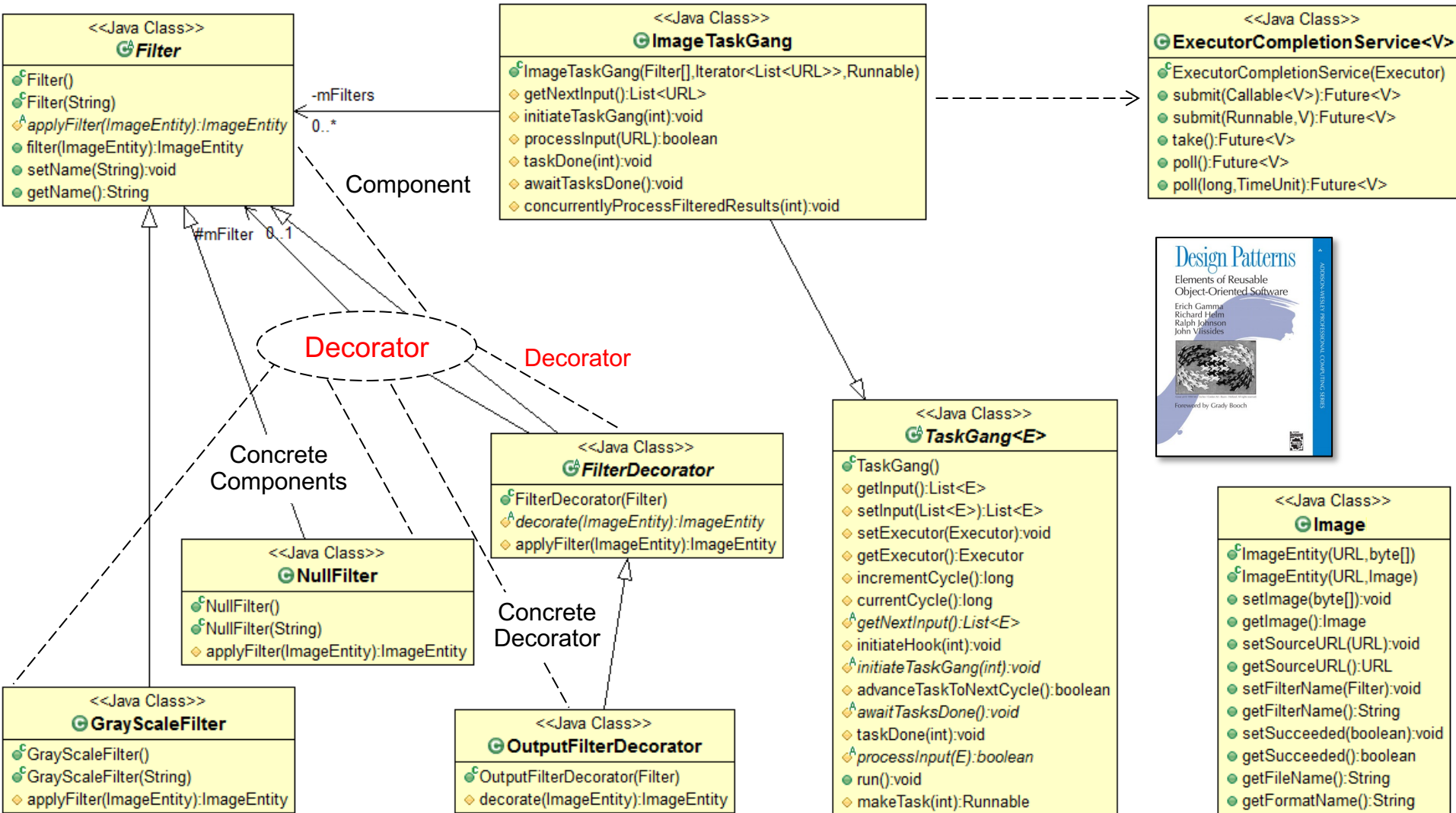


See [en.wikipedia.org/wiki/Grayscale](http://en.wikipedia.org/wiki/Grayscale)

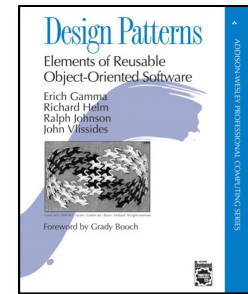
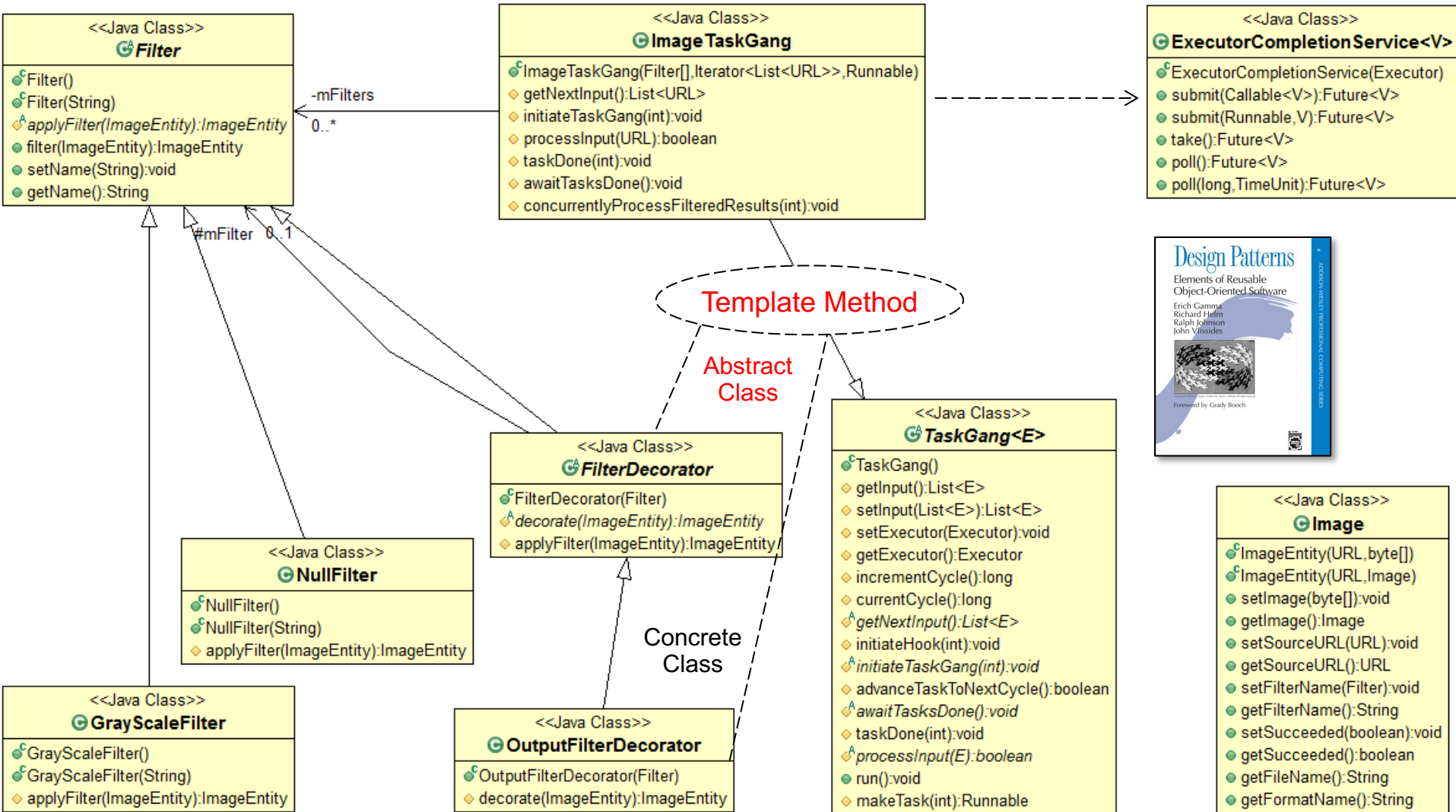
# Analysis of the FilterDecorator Class



# Analysis of the FilterDecorator Class

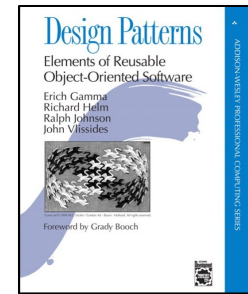
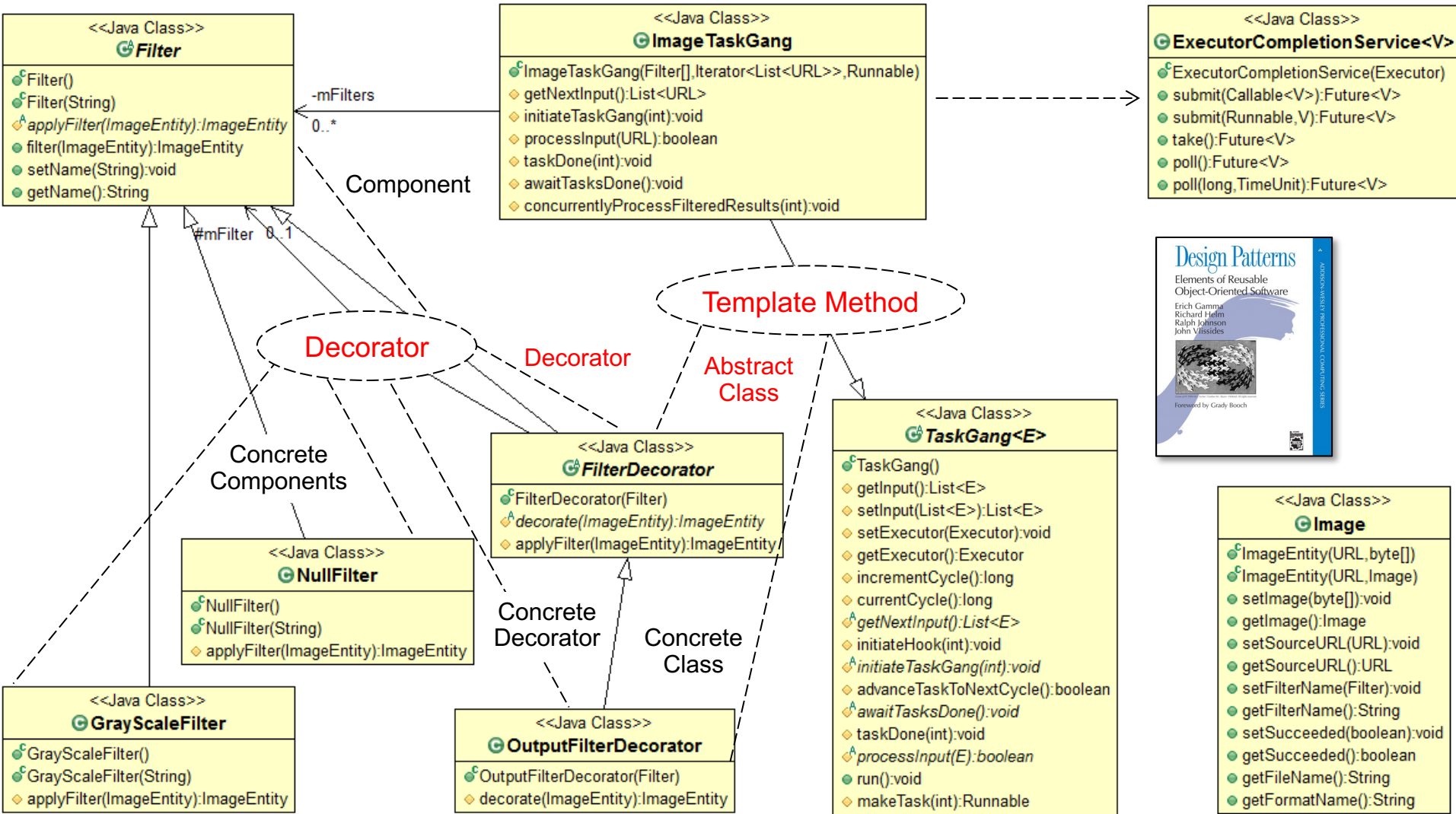


# Analysis of the FilterDecorator Class



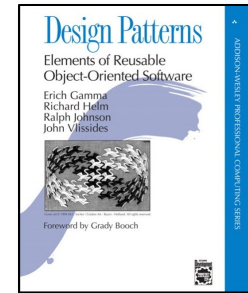
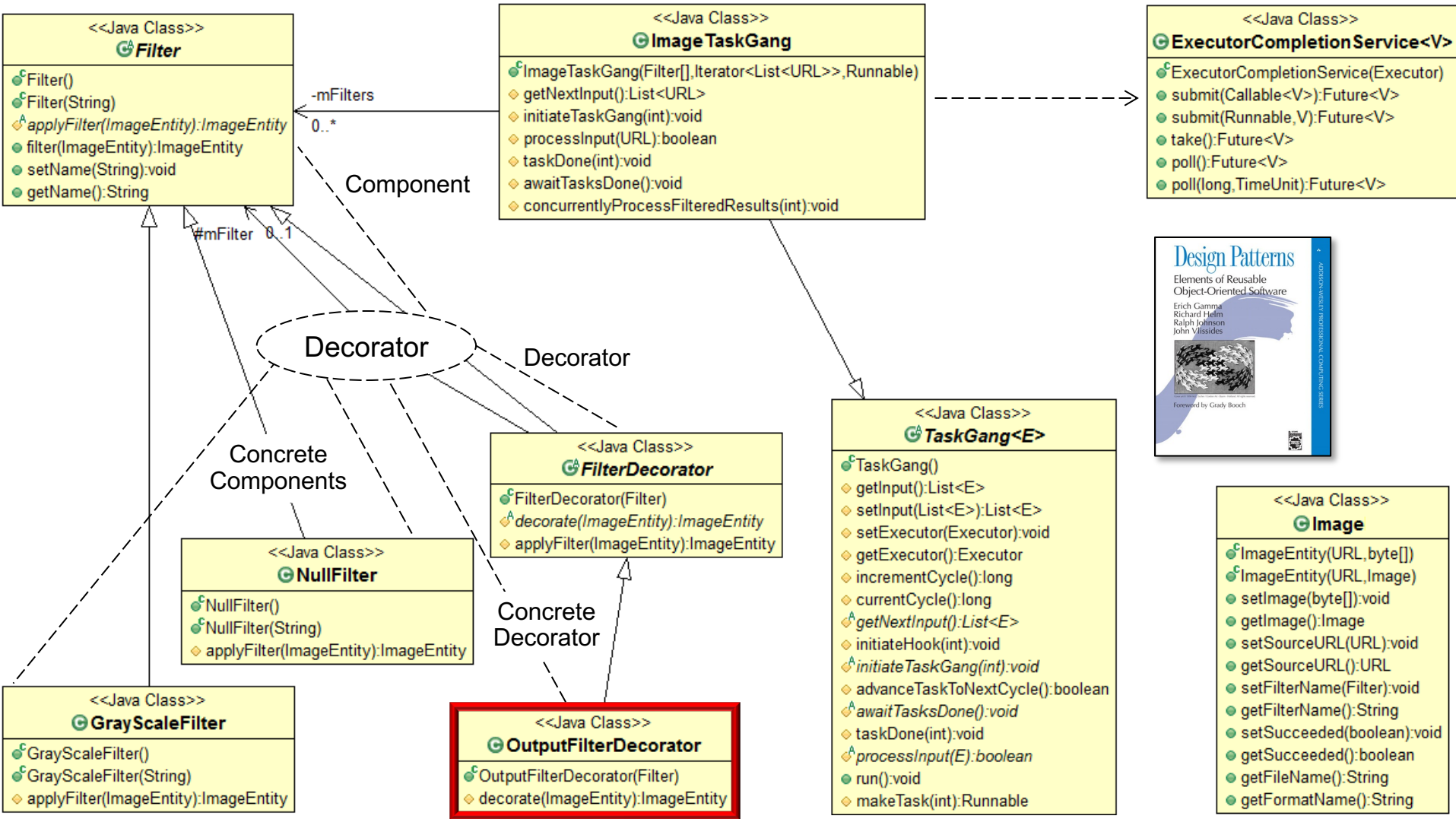


# Analysis of the FilterDecorator Class

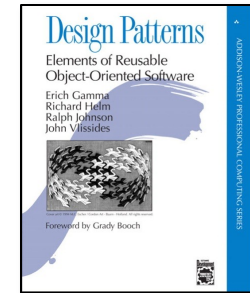
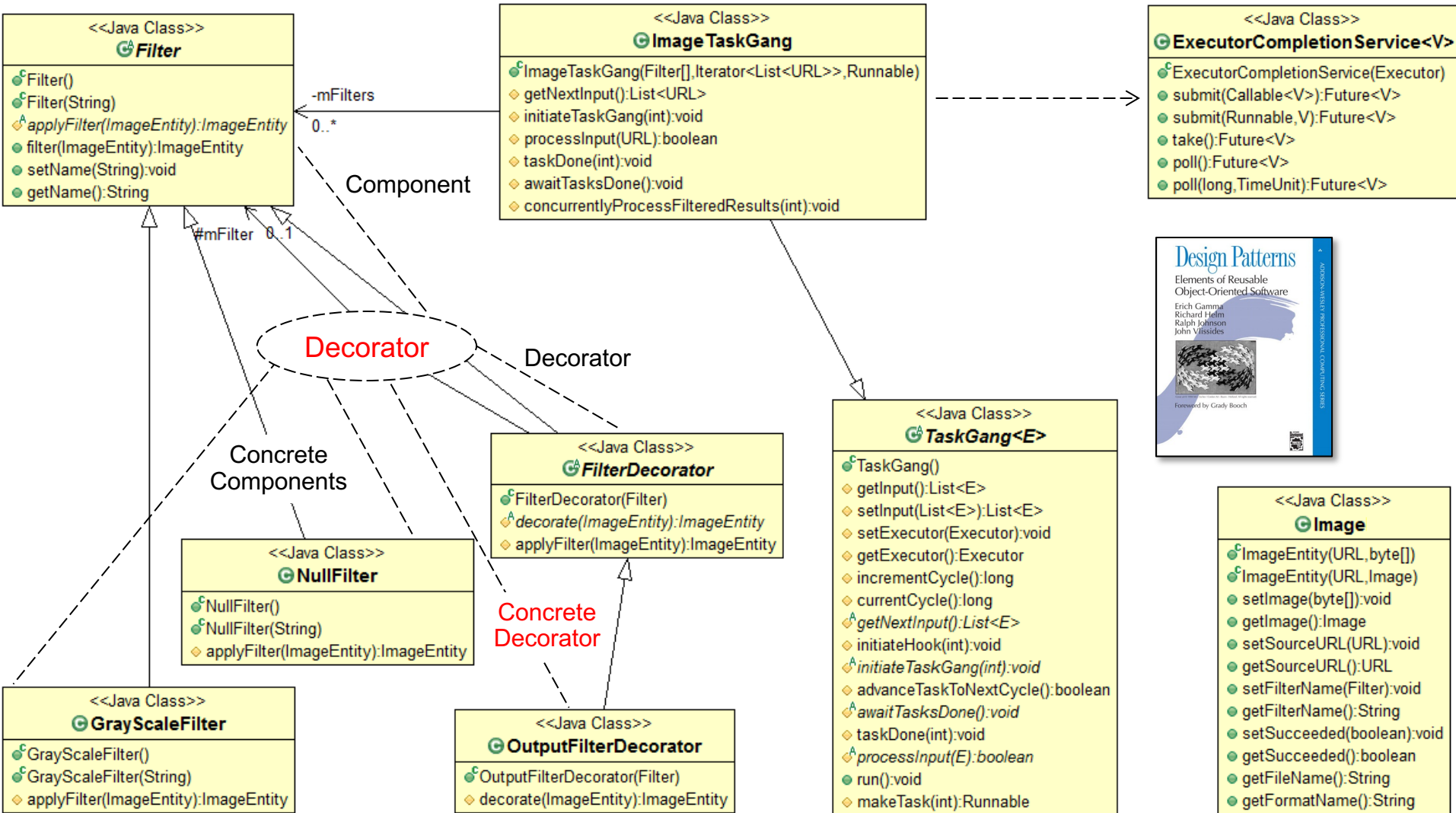


See [en.wikipedia.org/wiki/Decorator\\_pattern](http://en.wikipedia.org/wiki/Decorator_pattern) & [en.wikipedia.org/wiki/Template\\_method\\_pattern](http://en.wikipedia.org/wiki/Template_method_pattern)

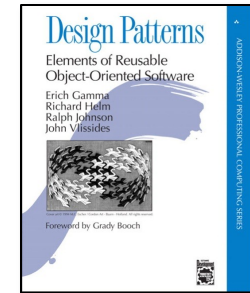
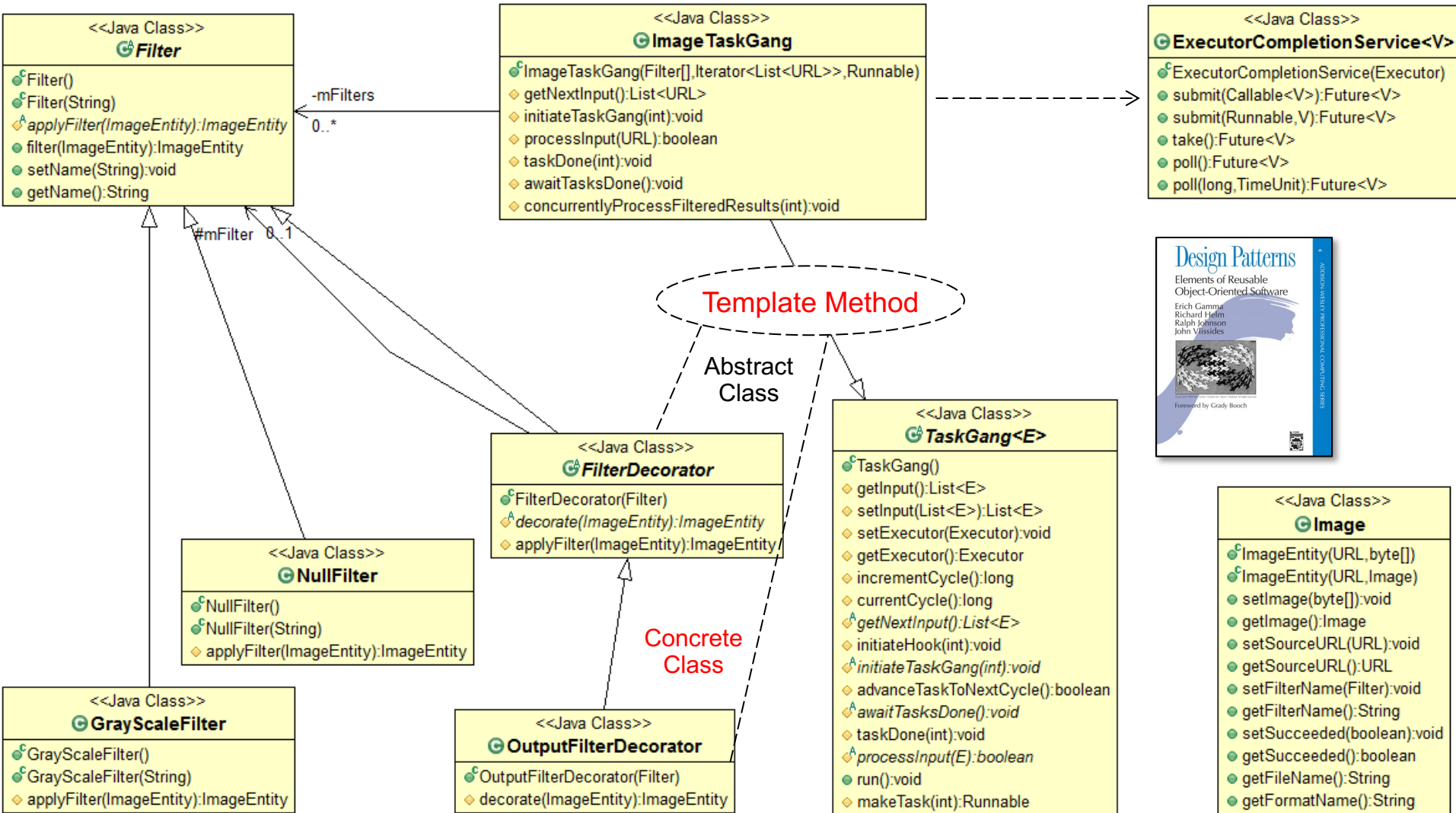
# Analysis of the OutputFilterDecorator Class



# Analysis of the OutputFilterDecorator Class



# Analysis of the OutputFilterDecorator Class



# Analysis of the Filter Hierarchy Classes

The screenshot displays an IDE window for the `ImageTaskGang` project. The left sidebar shows the project structure, with the `filters` package under `src/main/java/livelessons` selected. The main editor shows the `OutputFilterDecorator.java` file. The code includes a Javadoc comment and the class definition:

```
13  * A Decorator whose inherited applyFilter() template method calls the
14  * filter() method on the Filter object passed to its constructor and
15  * whose decorate() hook method then writes the results of the
16  * filtered image to an output file. Plays the role of the "Concrete
17  * Decorator" in the Decorator pattern and the role of the "Concrete
18  * Class" in the Template Method pattern.
19  */
20  public class OutputFilterDecorator
21      extends FilterDecorator {
22
23      /**
24       * Constructor passes the @a filter parameter up to the superclass
25       * constructor, which stores it in a data member for subsequent
26       * use in applyFilter(), which is both a hook method and a
27       * template method.
28       */
29      public OutputFilterDecorator(Filter filter) { super(filter); }
30
31
32      /**
33       * This hook method is called with the @a image parameter after it
34       * has been filtered with mFilter in the inherited applyFilter()
35       * method. decorate() stores the filtered Image in a file.
36       */
```

The IDE interface includes a top toolbar with icons for file operations and a bottom toolbar with tabs for Git, Run, CodeWhisperer Reference Log, Logcat, Profiler, Dependencies, TODO, Problems, Terminal, Services, and App Inspection. The status bar at the bottom shows "AWS: 2 Connections Expired", "CodeWhisperer", "20:14", "CRLF", "UTF-8", "4 spaces", and "master".

See [ImageTaskGangApplication/app/src/main/java/example/imagetaskgang/filters](https://github.com/leavesofcode/ImageTaskGangApplication/tree/master/app/src/main/java/example/imagetaskgang/filters)

---

# End of Analysis of the Filter Class Hierarchy