Nathan Cheung CSSE 374 12-16-14
CM 1586 Lab 2-2 Answers

1. Coupling

a) Content

In the DigitalBox constructor, getReceivers is called on the DataLine object before adding an IReceiver object to the list of IReceivers. The coupling could be looser if DataLine had an addReceiver method that took an IReceiver object. This way DigitalBox wouldn't be accessing the list of IReceivers in DataLine.

b) Common

This type of coupling exists between the DigitalBox, AviEncoder, and MpegEncoder classes. DigitalBox relies on the thresholds for AviEncoder and MpegEncoder to determine the value of its own threshold. The coupling could be looser if the DigitalBox just called a getThreshold method for each encoder instead of accessing the thresholds statically. This way, it wouldn't be able to set the thresholds.

c) External

There is external coupling in the received method of the AviEncoder class. The method uses the Util.transform method's protocol for handling data but doesn't call the Util class's method. Instead, it implements its own loops to achieve the same result. This coupling could be looser if the Util.transform method was called.

d) Control

There is control coupling between the DataLine, MpegEncoder, and AviEncoder classes. MpegEncoder and AviEncoder do rely on DataLine's TOTAL_BYTES field to determine whether a chunk of code gets executed but TOTAL_BYTES isn't being passed as a parameter. Therefore, the control coupling is very loose.

e) Stamp

There is stamp coupling in the received methods of the MpegEncoder and AviEncoder classes. Both methods take a Data object as a parameter because the encoders need the char array associated with that Data object. The coupling could be looser if the Data's char array was just passed as the parameter instead of the whole Data object.

f) Data

There is data coupling in the DataLine class's take method as well as the received method in the MpegEncoder and AviEncoder classes. The take method receives a buffer and does several things with its given parameter. The same could be said for the received method. A DataLine object is passed and it is manipulated. A Data object is passed and its char array is used to modify the encoder's buffer. The coupling for the encoder classes could be looser if the amount of unnecessary data was reduced by only passing the Data object's char array instead of the whole object.

g) Message

There is message coupling between the DigitalBox and the Util classes. If the BufferedReader throws an error, then the Util class is called to display the error. The coupling could be looser if there were no message being passed from the DigitalBox class to the Util class's method. Instead, the message would already be input in the println statement's parentheses.

2. Cohesion

a) Coincidental

The Util class has coincidental cohesion since it handles exceptions and also masks a buffer which are two very different things that probably shouldn't be grouped together.

b) Logical

There doesn't seem to be any logical cohesion.

c) Temporal

There is temporal cohesion between the Util and DigitalBox classes. The run method in the DigitalBox class processes input from lines 24-28 and also handles exceptions with the Util class on line 32. The Util class's processDataException method that is called is also supposed to log the error, notify the user, and close the file.

d) Procedural

There doesn't seem to be any procedural cohesion in the project since no modules appear to have sequential methods.

e) Communicational

There is communicational cohesion in the DataLine class since all of the methods operate on DataLine's own fields and data.

f) Sequential

There is sequential cohesion in the MpegEncoder and AviEncoder classes as their received methods alter the private StringBuffer field before passing themselves to other IReceiver objects. In this way, their output is the input to other received method calls.

g) Functional

There is functional cohesion in the Data class as each method does a single well-defined task.

3.

- DataLine’s take(char[] buffer) method does several actions and also returns an int.