**- Write down your thoughts for design, implementation, etc.**

**For Design**: creating three aspects:

1. EncryptionAspect.aj extends from OneWaySendAspect to handle the encryption process. This aspect uses the ConversationBegin pointcut.
2. DecryptionAspect.aj extends from OneWayReceiveAspect to handle the decryption process. This aspect uses the ConversationEnd pointcut
3. LoggingListenerTime extends from CompleteConnectionAspect to handle the connection. This aspect uses the ConversationBeginOnListener pointcut.

**For Implementation**: After understanding the code for (WeatherStationApplication) and according to my understanding to the application’s code in the previous activity and what I did, I think in making little changes to both the encryption and the decryption aspects that I created before by changing the applications name (the client and the receiver to be the Receiver and the Transmitter, respectively). Also, change the connection aspect (LoggingListenerTime.aj) to get the shared key from the key manager. One more thing is adding to the keyManager’s prcessList the user name and password for the server (i.e., the Transmitter).

These changed took just a couple of minutes after spending around two hours understanding the existing application.

**- Possible choices and your preferred choices**

I have also another choice to declare new aspects that are extending from:

1. ListenerJoinPointTracker and InitiatorJoinPointTracker to establish the connection between the KMClient and the KeyManager to get the shared key but I am not sure which one of the pointcuts I have to use.
2. MessageJoinPointTracker to do the encryption and decryption tasks but I am not sure which one of the pointcuts I have to use.

**- Possible problems you have faced**

The problem I faced is getting the application run because I had a problem with ParticipantRole.java class that is missing from the code. Also, most of the aspects that I wrote were inaccessible because of a little modification of the code Ali made in the Receiver class, after that everything work as expected.

**- Any suggestions**

There are no suggestions at all; it was a good employment of the coomj into these applications.

**- You comments, what was good thing/bad things about commj/example**

The good thing we have in Commj is the:

1. Structure of the connection aspects is very good
2. Structure of the communication aspects is very good
3. Structure of the messages aspects is very good
4. Structure of the aspects as all is very good,
5. The coverage of these aspects (cover most of the expected point cuts),
6. Weaving of these aspects is very easy,
7. The advices that are associated with these point cuts are very easy to understand.
8. It is very easy to use coomj in addition to basic understanding of the connection and communication concepts to develop a new applications or tasks.