E131)

The main difference between the ObservableClient and ObservableServer and adapters is that the ObervableClient and server extend the observable class. So they are also handling the the calls necessary for the observable layer to function. Adapters usually only perform the translation necessary for classes to work together.

E132)

A)

Advantages)

-Nothing modification necessary to add new observers.

-You can add as many observers as you want

-It allows you to send data to multiple clients efficiently

Disadvantages)

-You are limiting direct communication between objects.

-It has to be subclassed.

-the order of which clients receives notification from the observable is not always the same.

B)

Advantage)

- Connections are synchronized on the ConnectionToClient object. This allows for messages from different clients to be processed concurrently. Server does processing for each connection in its own thread.

-Also the factory allows for different sub classes of ConnectionToClient with unique implementation of handleMessageFromClient.

Disadvantage)

-Cannot handle all incoming messages sequentially because they are not synchronized between ConnectionToClient subclasses. Also if the superclass (ConnectionToClient) gets changed the subclass will need updated.

E133)

Change Log:

**Client Side**

* ChatClient now extends ObservableClient
* ChatClient constructor argument changed from ChatIF to Observer
* In ChatClient all instances of “clientUI.display(“”) have been changed to use the observable layer. These instances now call the method *NotifyObservers*() that simply calls *setChanged*() and *NotifyObservers*(message).
* ClientConsole now extends Observer
* ClientConsole adds itself as an observer of its instance of ChatClient by calling client.*addObserver*(this) in the ClientConsole constructor
* Changed display() of ClientConsole to *update()* so that it overrides from parent class Observer

**Server Side**

* EchoServer now extends ObservableServer
* EchoServer constructor argument changed from ChatIF to Observer
* In EchoServer all instances of “serverUI.display(“”) have been changed to use the observable layer. These instances now call the method *NotifyObservers*() that simply calls *setChanged*() and *NotifyObservers*(message).
* ServerConsole now extends Observer
* ServerConsole adds itself as an observer of its instance of EchoServer by calling client.*addObserver*(this) in the ServerConsole constructor
* Changed display() of ServerConsole to *update()* so that it overrides from parent class Observer