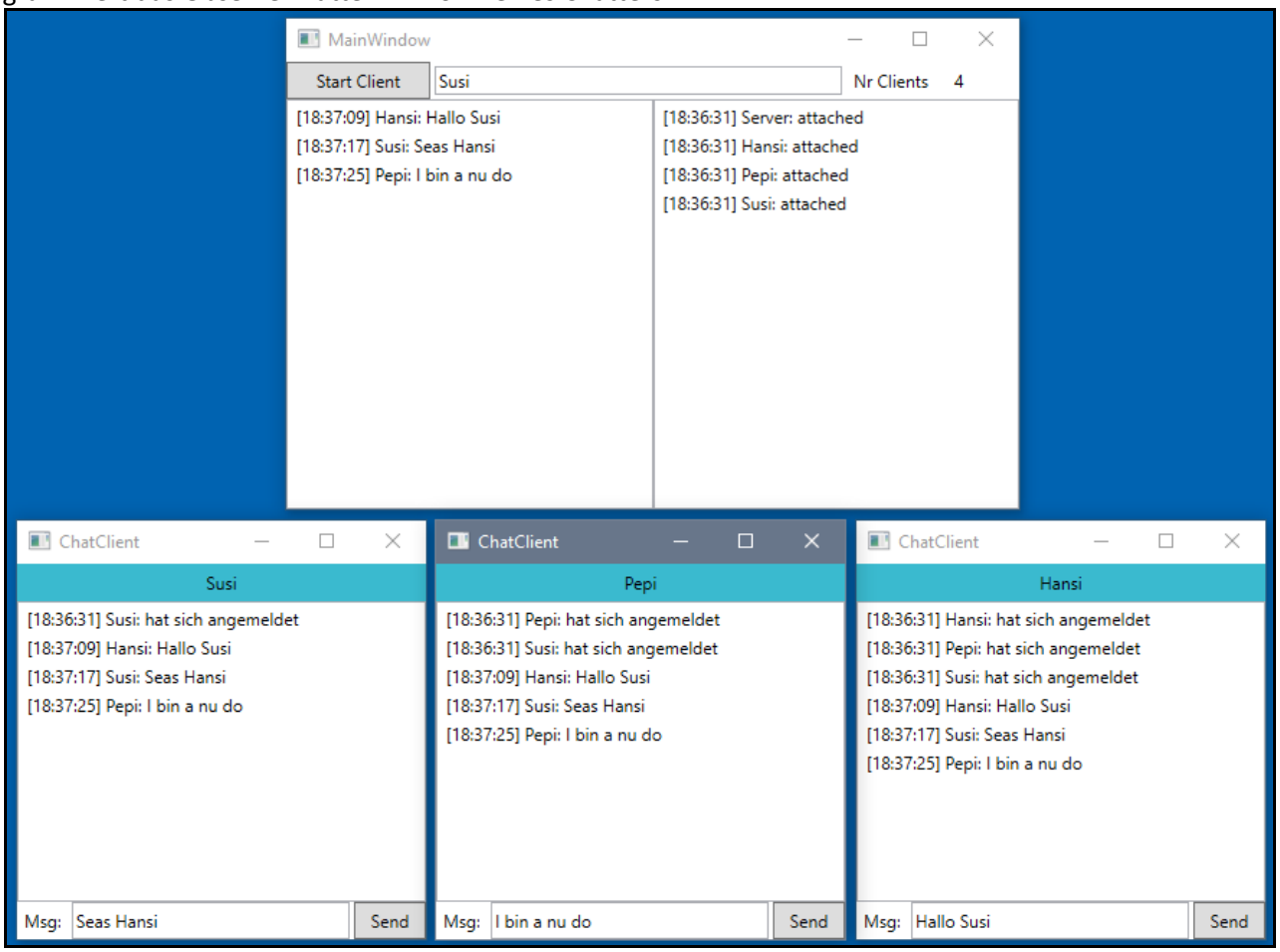
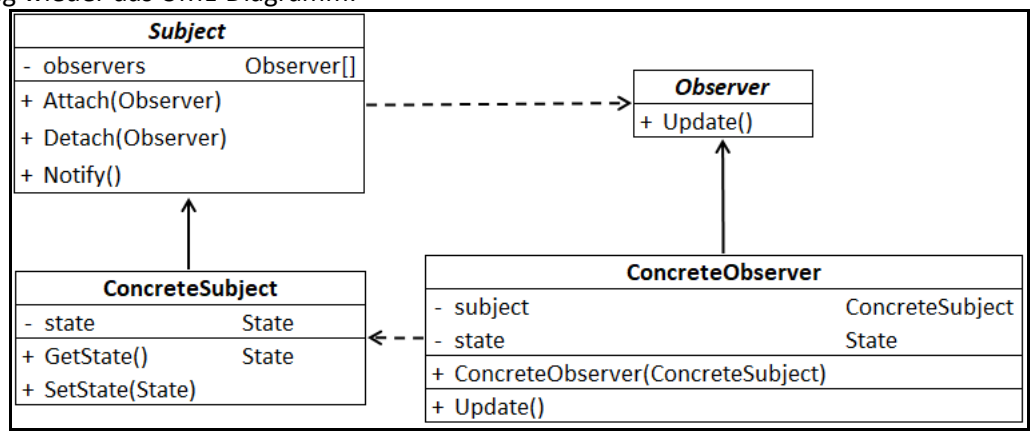
Übung Designpattern Observer
Chatter

Programmiert das Observer-Pattern in Form eines Chatters.



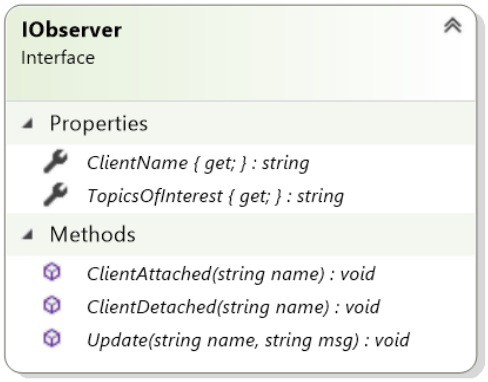
Es ist nicht sehr viel anders zu programmieren, wie die Observer-Übung mit der Uhr.

Zur Erinnerung wieder das UML-Diagramm:

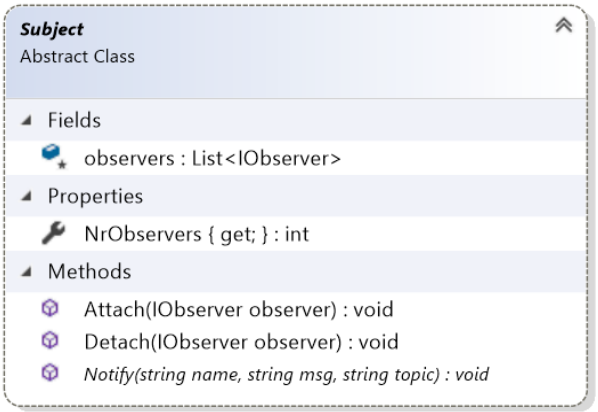


Folgendes ist zu beachten:

- die beiden Klassen/Interfaces **`Subject`** und **`IObserver`** sind praktisch unverändert zu programmieren.
- In IObserver habe ich noch Methoden für **`ClientAttached`** und **`ClientDetached`** hinzugefügt. Damit kann man sich benachrichtigen lassen, wenn sich ein Client an bzw. abmeldet.



- die Klasse **`Subject`** verwaltet die Liste der Observer. Sie ist relativ unabhängig von einer Aufgabenstellung zu implementieren. Von dieser Klasse darf es nur eine Instanz geben (so ähnlich wie das Model in MVC)



  topic entspricht dabei dem „Thema“ aus der Erweiterung unten.
- Die einzelnen „**Konkreten Beobachter**“ sind das MainWindow sowie die ChatWindows. Am besten wird das Subject im MainWindow-OnLoaded erzeugt. Es muss jedem ChatWindow übergeben werden.
- Die einzelnen Observer melden sich beim Subject an und implementieren das IObserver-Interface. In den Callbacks werden die Meldungen auf eine ListBox geschrieben.

Erweiterung

Fügt den Chat-Windows eine zusätzliche Textbox hinzu, in der man einen String für ein Thema eingeben kann, unter dem man seine Message verschicken möchte.
Das Subject ist dann so umzubauen, dass nur noch jene Clients benachrichtigt werden, die sich für das Thema angemeldet haben.
Es ist somit beim Attach auch zusätzlich ein String anzugeben, der diese Themen angibt (z.B. mit Beistrich getrennt). Eine andere Möglichkeit wäre, dass der Observer eine Property hat, die die Themen liefert – dann könnte man an nach erfolgter Anmeldung seine Themen ändern.