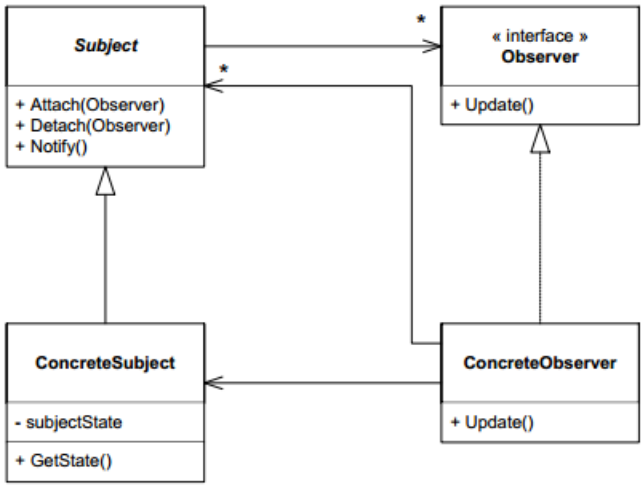
4: GoF Observer

1 Redegør for, hvad et software design pattern er

Et design pattern er kernen i en løsning til et problem, som opstår gentagende gange. Årsagen til, at det er et mønster, er, at problemet og løsningen af problemet kan anvendes igen og igen uden, at man gør det samme 2 gange.
Da design patterns altid skal tilpasses vores problem i vores kontekst. Et design pattern er derfor aldrig en færdig løsning på et problem, men en skabelon til løsning af et problem.

2 Redegør for opbygningen af GoF Observer.

Strukturen for Observer Pattern ser således ud:



2.1 Pull

Eksemplet nedenfor viser et eksempel på Pull varianten af observer pattern.

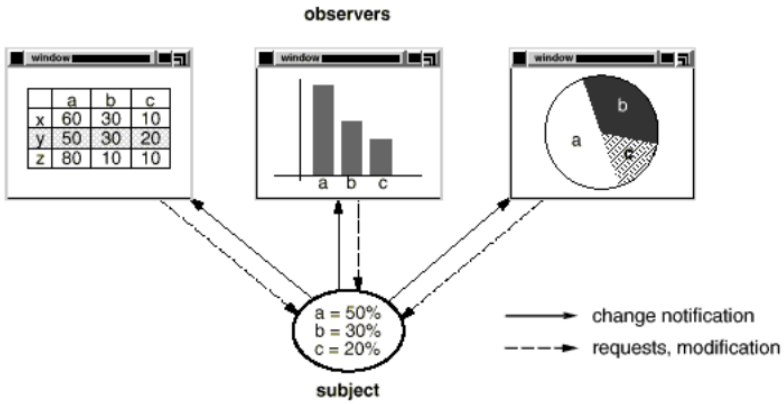


Figure 1: Pull variant

I pull varianten notify'er Subjectet(publisheren) om ændringer, hvorefter observer'erne requester staten på subjectet, hvis observeren har behov for dette.

2.2 Push

I Push varianten skubber subjektet informationen til observeren. Dette har den fordel at der kun kræves et kald hver gang der kommer en opdatering.

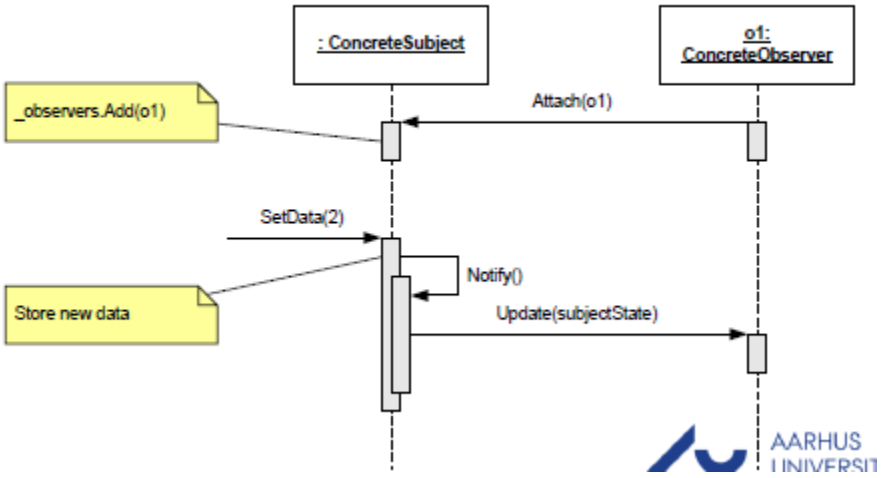


Figure 2: GoF Observer Push

3 Sammenlign de forskellige varianter af GoF Observer – hvilken vil du anvende hvornår?

De to typer GoF Observer: Pull og Push har forskellige egenskaber og jeg vil anvende dem som følgende:

Pull: Denne har den fordel at observeren kan beslutte om denne ønsker at få opdateringen. Det kan være tilfældet at subject, en alarmcentral, har to observers: Brand og redning samt politi. Subject udsender altså fx opdateringer om røveri, som barnd og redning ikke skal bruge og kan derfor ignorere beskeden. Jeg vil andså bruge pull når objekterne har forskellige præferencer.

Push Push har den fordel at den sender beskeden med det samme. observer skal altså ikke requeste denne. Denne metode vil jeg kun anvende hvis alle observere har samme præferencer og derfor altid skal have alle opdateringer. Her skal man dog overveje om der kan komme en udvidelse med en observer som har andre præferencer.

4 Redegør for, hvordan anvendelsen af GoF Observer fremmer godt software design

Fordelen ved Observer Pattern er at det i et system med meget kommunikation giver en overskuelig og genanvendelig struktur.
Anvendelsen af GoF Observer gør det også muligt at kommunikere mellem flere lag i fx MVP, da der ikke må være genskab til klasserne imellem begge veje, men kun en vej (fra øverste lag og nedad). I observer patternet er det kun 1 lag/klasse som har kendskab til den anden, derfor fremmer det software design i den context.

5 Redegør for fordele og ulemper ved anvendelsen af GoF Observer

Fordele:

- Løs kobling, da subjekt(publisher) ikke skal vide noget om observer(subscriberne)
- opfylder OPC, da man kan nemt kan tilføje flere typer af observers.

Ulemper:

- Kan tilføje unødig kompleksitet, hvis man ikke passer på. Igen tilpas løsningen på problemet til konteksten

6 Redegør for, hvilke(t) SOLID-princip(per) du mener anvendelsen af GoF Observer understøtter

- Der bliver overholdt OPC.
- LSP er også overholdt da underklasserne kan substituere superklasser/interfaces.
- DIP bliver ligeledes overholdt, da forbindelsen fra underklassen af observer til subjektet ikke bryder princippet, fordi SUBJECT klassen aldrig bør instantieres.