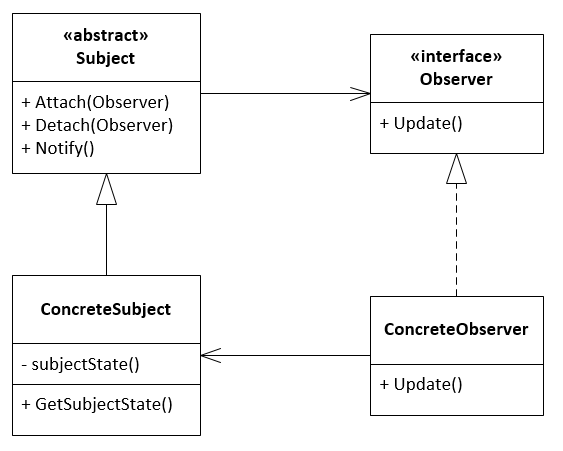
Patterns 2: GoF Observer

# Redegør for, hvad et software design pattern er.

Et software design pattern er en general genbrugelig løsning til problemer der tit opstår i en given kontekst i software design. Det er ikke et færdigt design der kan laves direkte til kilde kode. Det er en beskrivelse eller skabelon for hvordan et problem kan løses i mange forskellige situationer. Det er formaliserede bedste praksisser som en programmør kan bruge til at løse problemer med.

# Redegør for opbygningen af GoF Observer

Dette pattern er opbygget af subjects og observers, subjects er klasser som ændrer sig, og det ønsker vi at følge med i. Observers er dem der ønsker at blive opdateret når subjects ændrer sig.

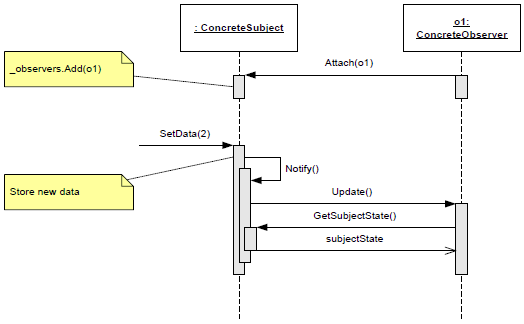


Her ses pull versionen push er næsten det samme her kender Observer bare ikke Subject.

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

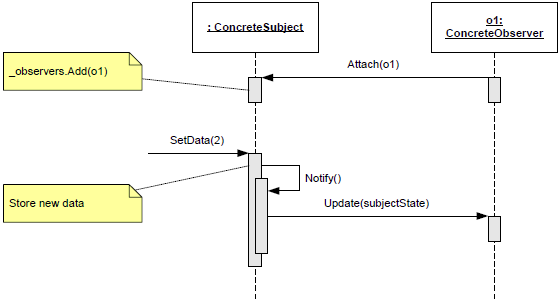
Der er to variationer push og pull.

Pull bruges når der er ønskværdigt at observeren bestemmer om den vil opdateres, det er en fordel at bruge i et simpelt system. Det er også nemt at implementere Subject og Observer som genbrugelige enheder.



Her ses sekvens diagrammet med et subject og en observer hvor pull er blevet brugt

Push er bedre at bruge hvis vi har meget data der kan ændre sig eller hvis alle observers skal have alle opdateringer. I tilfældet med meget data kan det være at kun lidt af dataen er blevet opdateret og det der denne vi ønsker at sende ud til observers.



Push varianten her ses det at observer ikke skal spørge om data men at det bliver skubbet ud til ham.

# Redegør for hvordan anvendelsen af GoF Observer fremmer godt software design

Fordelen ved at bruge observer pattern ses når vi har et system med meget kommunikation da det kan overskueligøre kommunikationen med sin genanvendelige struktur. Observer gør det også nemt at have lagdelt software da data som vi ønsker at udstille blot kan laves observable og derved ikke behøves at kende til laget ovenover.

# Redegør for fordele og ulemper ved anvendelsen af GoF Observer

Fordele:

* Opfylder OPC da der nemt kan tilføjes nye observere
* Lav kobling da subject ikke skal kende observer

Ulemper

* Kan ende i et meget komplekst system hvis pattern bruges for meget. Her er det vigtigt at overveje sine behov og bygge systemet efter disse.

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

* OPC er overholdt subjecter er lukkede for ændringer men kan udvides med observer.
* LSP overholdes også da concretesubject er en substitut for subject.
* DIP overholdes også da subject aldrig bør instantieres.