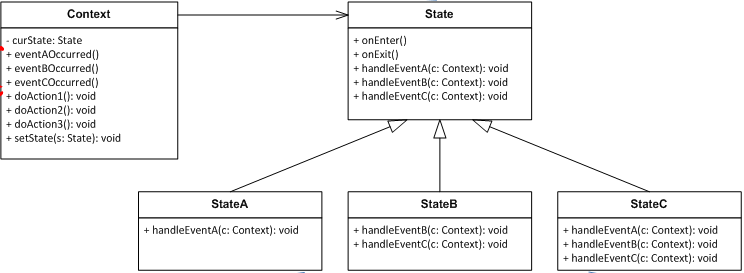
6. Patterns 4: State patterns

# Redegør for, hvad et software design pattern er

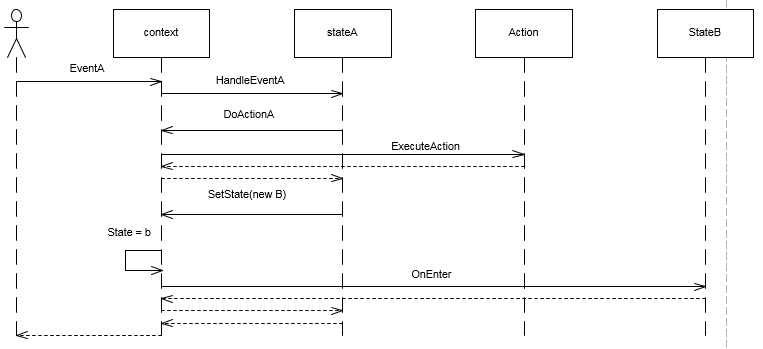
Et software design pattern er en general genbrugelig løsning til problemer der tit opstår i en given kontekst i software design. Det er ikke et færdigt design der kan laves direkte til kilde kode. Det er en beskrivelse eller skabelon for hvordan et problem kan løses i mange forskellige situationer. Det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

# Redegør for strukturen i GoF State Pattern

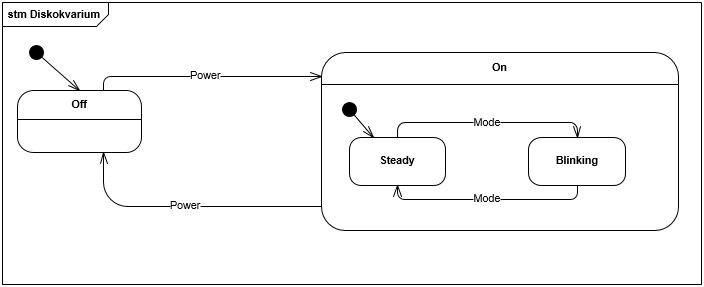
State pattern er opbygget via klasser hvor der er en klasse der agere kontekst det er denne klasse vi kalder når der sker events. Dertil er der en anden klasse som agere ”Superstate” det betyder at den kan blive kaldt med alle de mulige events der er i systemet. Fra denne SuperState nedarver de individuelle states og implementer de events de har behov for. Det er ikke konteksten der sætter state det gør de individuelle states.



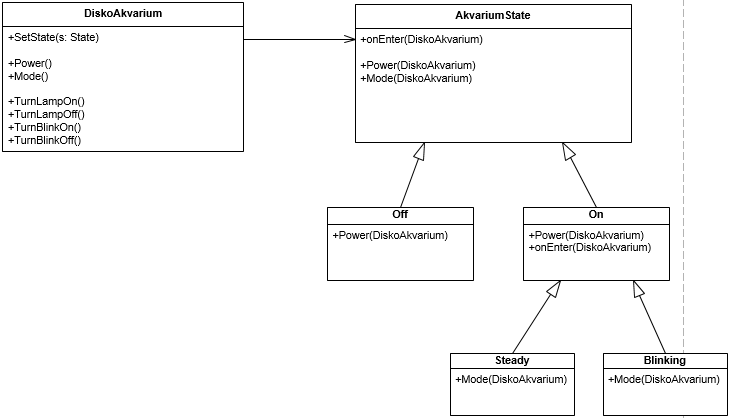
Her kunne det være sjovt at se et sekvensdiagram for hvordan kommunikationen foregår.



# Sammenlign switch/case-implementering med GoF State

Ved mindre state maskiner kan GoF state ikke svare sig her er det mere overskueligt med en switch case.

Men har vi for eksempel nestede states giver GoF nogle fordele.



# Redegør for fordele og ulemper ved anvendelsen af GoF State.

Fordele:

* Der opnås større overskuelighed ved store statemachines. Det er nemt at udvide
* Der er adskillelse mellem actions og state machine logikken.
* Det er fleksibelt og effektivt

Ulemper:

* Det kan være svært at denne sig et overblik over statemachinen ud fra kode.
* Der skal meget kode til!

# Redegør for, hvordan et UML (SysML) state machine diagram mapper til GoF State.

Der kan mappes fra et statediagram til GoF state så ledes:

* Alle states 🡪 Superstate
* Individuelle 🡪 SubState på superstate
* Starting point 🡪 Første state der bliver sat
* Events 🡪 De events der kan kaldes på konteksten
* Guards 🡪 Logik der skal implementeres i states
* Activity 🡪 Kald der laves tilbage på konteksten
* Tilstandskiftet 🡪 Hvilken State SetState skal kaldes med