GRASP overvejelser

Følgende er de reflektioner jeg har gjort mig i løbet af processen.

Information Ekspert

Som udgangspunkt finder jeg det ret naturligt at placere metoder der hvor den nødvendige viden er. I forbindelse med min AnimalCheck_SameLocation() ønskede jeg egentlig at placere metoden i Animal-klassen (af hensyn til cohesion), men eftersom metoden tjekker dyr i en List, var jeg nødsaget til at holde metoden i min SavannahGame-klasse.

Controller

Min Game_Controller- klasse er en direkte afledt klasse af dette princip. Ikke så mange overvejelser om dette, andet end det også sikrer nem udskiftelighed (af GUI) og lavere kobling.

Creator

Min AnimalFactory-klasse er også med tanke på Creator princippet og klassen står for at oprette de instanser jeg skal bruge i spillet. Igen fokus på lav kobling.

Coupling

På trods af at jeg har benyttet mig af flere patterns der gerne skulle sikre lavere kobling, så er jeg nok blevet fanget i at jeg ofte sender hele instanser af objekter mellem de forskellige klasser.

F.eks. i stedet for at bruge Animal som parameter i en motode, vil jeg fremadrettet bruge Animal.animalType som parameter, såfremt metoden ikke har brug for mere.

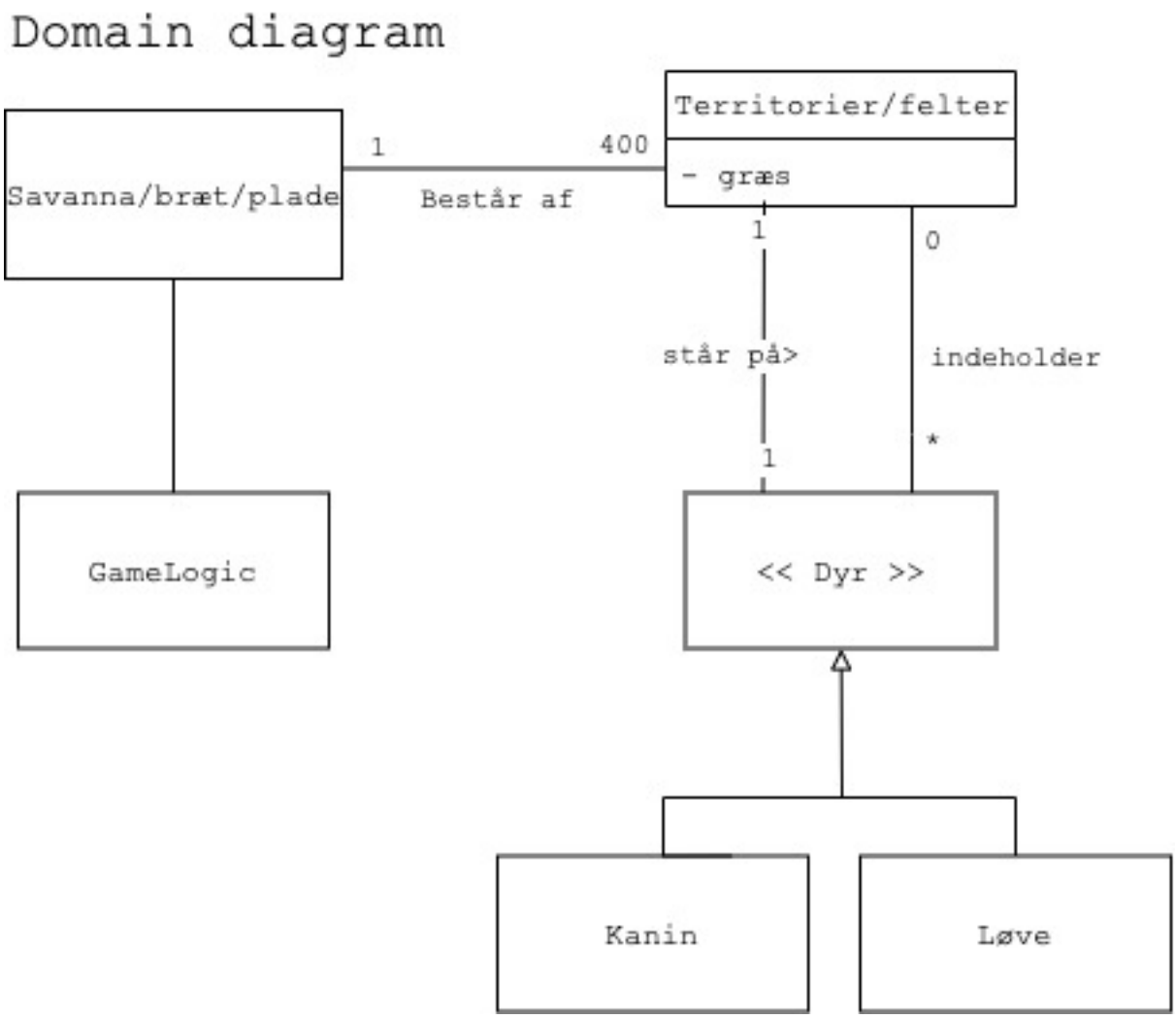
Derudover har jeg anvendt Proxy-pattern på mit persistente lag.

Cohesion

Ikke så mange overvejelser, andet end det nok er et princip der altid ligger i baghovedet.

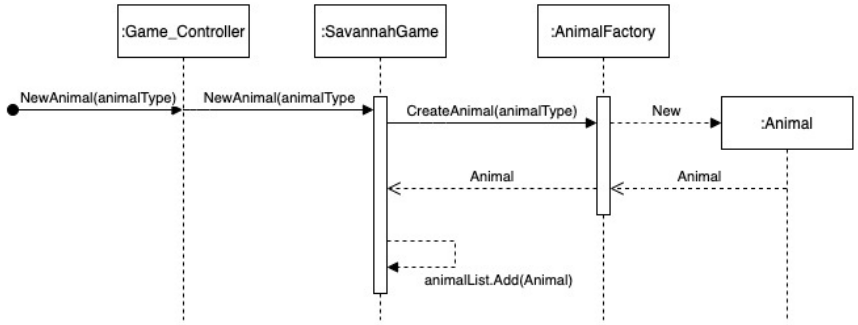
Generelt for ovenstående er også at der er anvendt interfaces og brugt polymorfi i den udstrækning det har været muligt.

Domænemodel

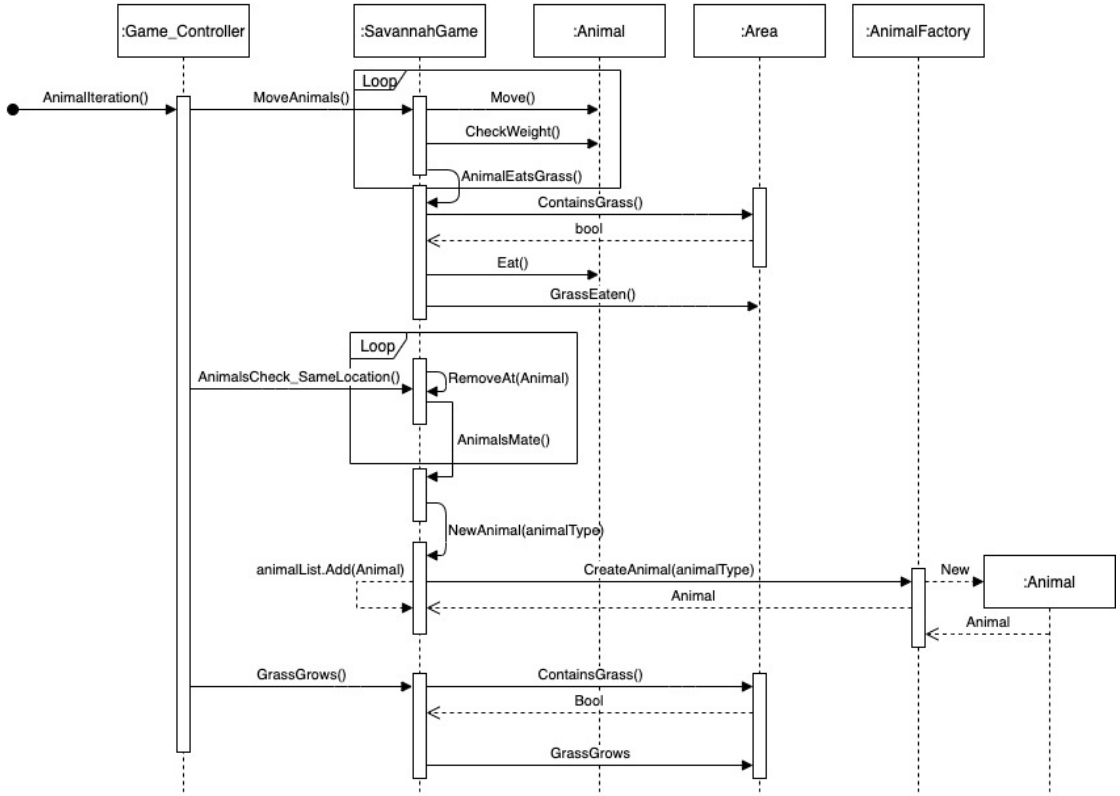


Sekvensdiagrammer

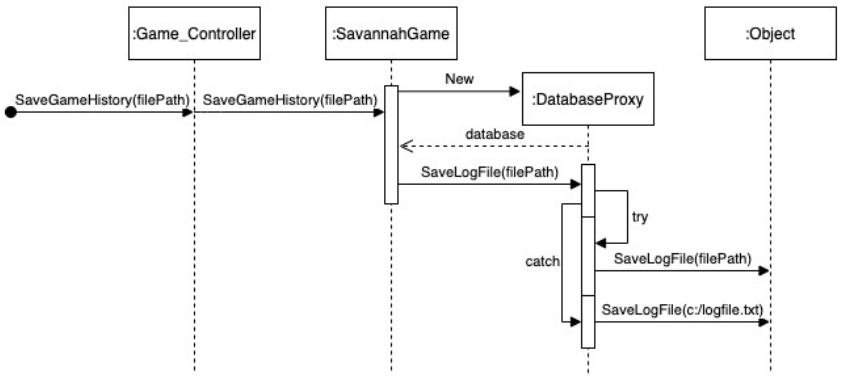
Sekvensdiagram: NewAnimal



Sekvensdiagram: AnimalIteration



Sekvensdiagram: SaveGameHistory



Design-klassediagram