Danmarks Tekniske Universitet

02312, 62531, 62532
INDLEDENDE PROGRAMMERING, UDVIKLINGSMETODER TIL IT-SYSTEMER OG VERSIONSSTYRING OG TESTMETODER

CDIO del 3

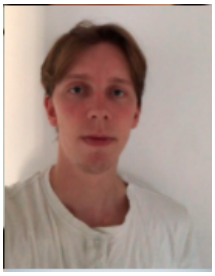
`https://github.com/ExoHumann/40_del3.git`

Authors


Krzysztof Kisiel s192884


Oliver Olsen s205443


Johannes Jensen s205413


Gustav Fauser s205446


Jonathan Hansen s205413


Alaa Hadood s205413

13. november 2020

1 Timeplan

Tid i timer Navn	Krav og analyse	Design	Implementering	Test	Programmering
Gustav Fauser		1			
Oliver Olsen		1			
Krzysztof Kisiel		1			
Johannes Jensen		1			
Jonathan Høj		1			
Alaa Hadood		1			

Tabel 1: Tabel over antal timer brugt på hver del af projektet

2 Summary

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Donec odio elit, dictum in, hendrerit sit amet, egestas sed, leo. Praesent feugiat sapien aliquet odio. Integer vitae justo. Aliquam vestibulum fringilla lorem. Sed neque lectus, consectetuer at, consectetuer sed, eleifend ac, lectus. Nulla facilisi. Pellentesque eget lectus. Proin eu metus. Sed porttitor. In hac habitasse platea dictumst. Suspendisse eu lectus. Ut mi mi, lacinia sit amet, placerat et, mollis vitae, dui. Sed ante tellus, tristique ut, iaculis eu, malesuada ac, dui. Mauris nibh leo, facilisis non, adipiscing quis, ultrices a, dui.

Indhold

3 Introduktion

IOOuterActive (os) har fået til at opgave at designe og implementere et Monopoly Junior spil. Der skal implementeres de væsentligste elementer fra brætspillet, for at spillet kan spilles. Spillet skal kunne spilles mellem 2-4 spillere.

4 Krav

For at spillet lever op til kundens forventninger opstilles der en række krav som spillet skal leve op til. Kravene er opstillet ud fra kundens vision(Nyborg og m.fl 2020)

1. Få et beløb når du passerer start feltet.
2. Tag chancekort læg det nederst i bunken.
3. Implementering af forskellige chancekort (beslutter vi sammen).
4. Fængsel betal for at komme ud eller brug "Get out of jail" kortet.
5. Parkering / Besøg i fængsel.
6. Der skal være mulighed for at købe et felt hvis man lander på det.
7. Hvis en spiller ejer et felt, skal andre spillere betale husleje, når det lander på dette.
8. Hvis spilleren ejer flere af samme farve felt koster det mere.
9. Man skal kunne definere sin brik(farve/type/pattern/fill).
10. Man skal kunne se hvem der har købt ethvert felt.
11. Når en spiller lander på et felt, skal de fortsætte fra det felt på næste tur.
12. Spillet skal spilles af 2 til 4 spillere.

5 Analyse

Der skal udarbejdes følgende artifacts (udover kravliste):

1. Use case diagram
2. Eksempler på use case beskrivelser - vælg mindst én, der beskrives fully dressed
3. Domænemodel
4. Et eksempel på systemsekvensdiagram
5. Et eksempel på sekvensdiagram
6. (Et) Designklassediagram

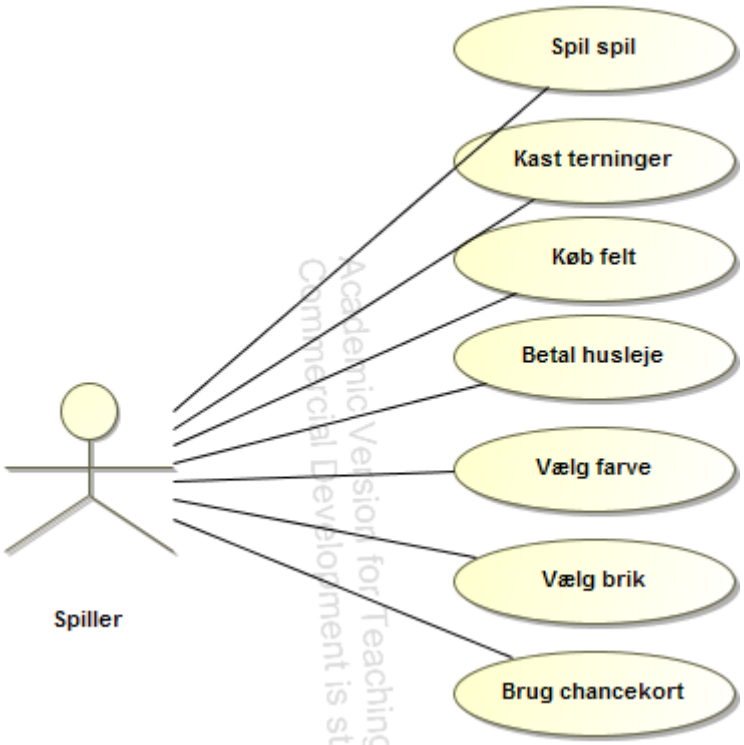
Der er vigtigt, at der er tydelig sammenhæng mellem beskrivelser og diagrammer. Det skal således være muligt at trace fra kravliste til analysedokumentation, designdokumentation og implementering. Beskriv ligeledes jeres brug af GRASP-mønstre.

5.1 Use cases

I opgaven er spillet, udarbejdet ud fra en række Use cases. Use cases beskriver det spillet skal bruges til at en bruger/spiller.

- Spil spil
- Kast terning
- Vælg farve/brik
- Køb felt
- Brug chancekort
- Betal husleje

De ovenstående use cases er illustreret i Figur 1



Figur 1: Use case diagram

Herunder i tabel 2 er use casen Spil spil blevet beskrevet fully dressed

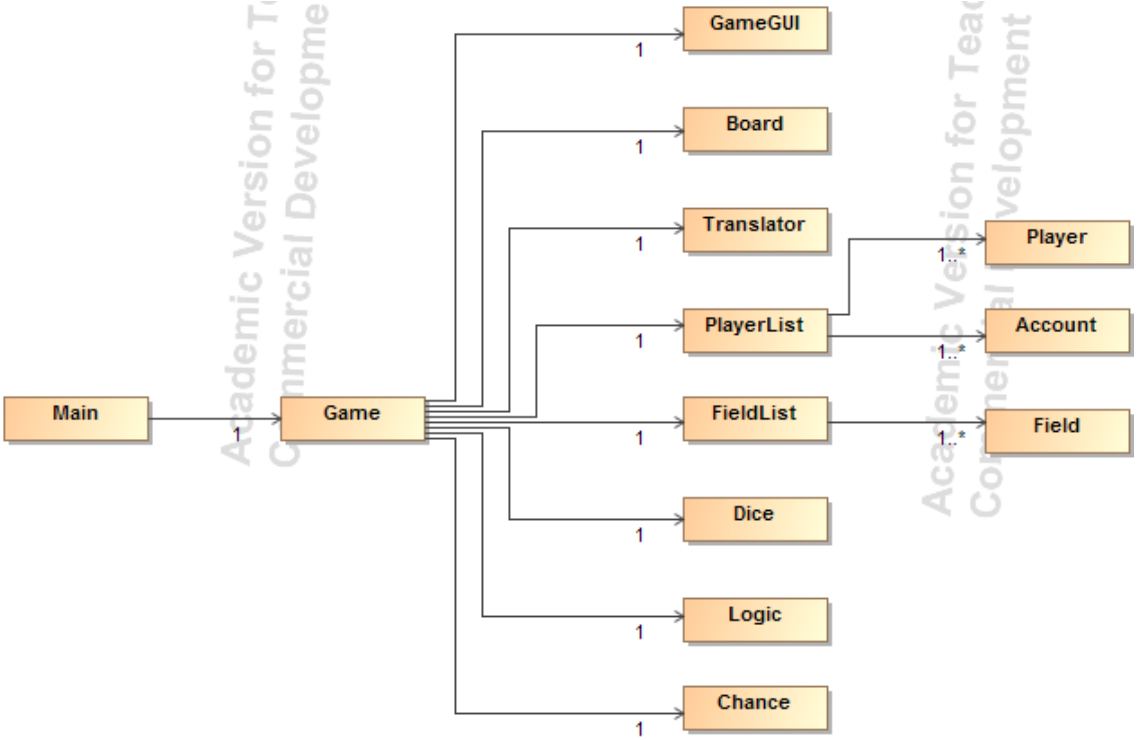
Use case	Spil spil
Omfang	UC1
Mål	Spilleren påbegynder spillet, for et antal spillere.
Primær aktør	Spilleren
Interessenter	De andre medspillere
Startbetingelser	Spilleren vil gerne spille spillet
Succes kriterie	Spillet starter med det valgte antal spillere
Hoved succes Scenarie	1. 2-4 personer vil gerne spille spillet. 2. Spillet startes op på computeren 3. Der vælges et antal spillere fra 2-4 4. Der vælges navn for spillerne • Spiller 2-4 vælger navn *gentag indtil alle spiller har valgt navn.* 5. Der vælges farve for spillerne • Spiller 2-4 vælger farve *gentag indtil alle spiller har valgt en farve.* 6. Der vælges en brik til alle spillerne • Spiller 2-4 vælger brik *gentag indtil alle spiller har valgt en brik.* 7. Spillet starter med de valgte navne, farver og brikker.

Udvidelser	• Der indtaster mindre eller flere spillere end 2-4 1. Spillet giver en fejlbesked 2. Spilleren ændrer antal spillere 3. forsætter fra 3. i hoveddelen • Der indtastet en forkert farve for en af spillerne 1. Spillet giver en fejlbesked 2. Spillet lukker ned • Der vælges en forkert brik, eller en brik der er optaget 1. Spillet giver en fejlbesked 2. Spilleren vælger en anden brik 3. fortsætter fra 7. i hoveddelen
Specielle krav	Ingen
Teknologi- og dataformats-variationer	Ingen
Hyppighed	Hver gang spillet starter fra ny
Diverse	Ingen

Tabel 2: Fully dressed beskrivelse af use case UC1

5.2 Domænemodel

Der er blevet udarbejdet en domænemodel (figur 2) over spillet, hvor sammenhæng mellem klasserne fremgår.



Figur 2: Domænemodel over Spil

5.3 Grasp-mønstrer

GRASP-mønstre er generelle design mønstre til at give ansvar til enkelte klasser i et program/spil.

Skaber Skaber mønsteret tildeler ansvaret for hvilken klasser der opretter nye objekter. I spillet har klassen "Game"ansvaret for at oprette nye objekter, som man kan se på figur 2

Lav kobling Lav kobling omhandler at klasser har lav afhængighed af andre klasser, samt ændringer i en klasse ikke påvirker andre klasser. Det betyder også at klasserne er lettere at genbruge i andre projekter. Lav kobling anvendes på alle klasserne i spillet.

Høj sammenhørighed Høj sammenhørighed omhandler at hver klasse har sit eget veldefineret og fokuserede ansvarsområde. Mønsteret er anvendt i mange af spillets klasser f.eks. i "Dice-klassen, som kun indeholder attributter og metoder der omhandler terninger.

5.4 Klasser

Udfra domænemodellen er der blevet defineret en række klasser, som spillet består af. Klasserne er blevet defineret ud fra MVC-princippet (Model view controller).

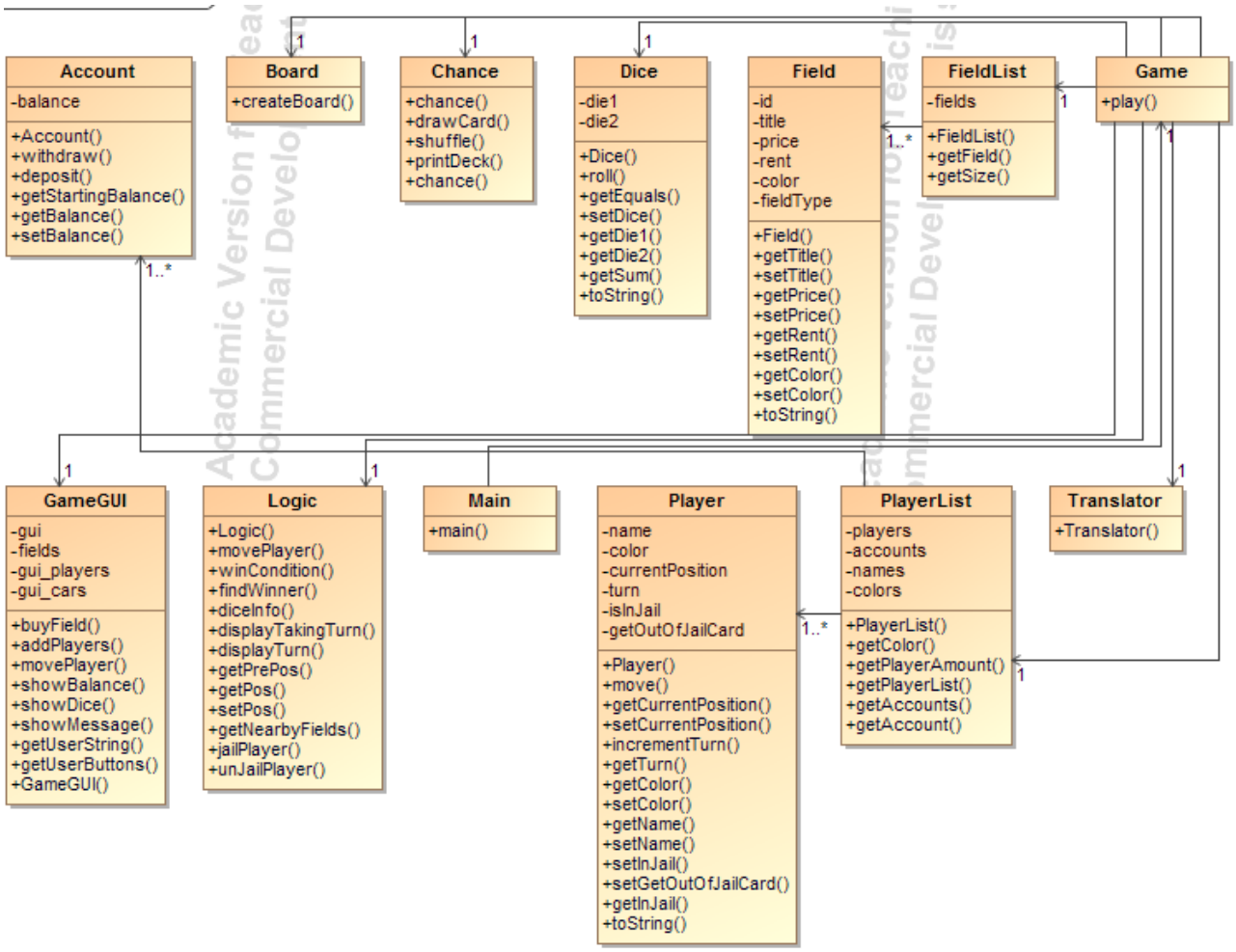
6 Design

6.1 Systemsekvensdiagram

6.2 Sekvensdiagram

6.3 Designklassediagram

Designklassediagrammet (figur 3) viser hvilke metoder de enkelte klasser skal indeholde.



Figur 3: Designklassediagram over spil

7 Implementering

8 Dokumentation

9 Konfiguration

10 Test

11 Projektplanlægning

12 Konklusion

Figurer

Tabeller

Litteratur

Nyborg, Mads og m.fl (2020). *CDIO del 3*. en. URL: https://docs.google.com/document/d/1UsCw9jeOvZlthzUrxIg0sF8ez1xwk-64oh6KivQvzHI/edit?usp=embed_facebook (bes. 09.11.2020).

13 Bilag