Technical University of Denmark

32_CDIO2

Course:
02312, 02313, 02315

Authors:

Lucas Gramme, s164863
Alexander Nørgaard, s182934
Mads Jensen, s185124
Fredrik Mejlsbye, s185130
Maria Kajgaard, s185108

9. november 2018

Indhold

Resume

Vi har efter kundens ønske udviklet et brætspil for 2 personer, som vindes ved at man først når til scoren 3000. Specifikke krav til spillet er blevet givet af kunden, samt en liste over nice-to-do's af vores projektleder. Kravene er beskrevet detaljeret i vores "Krav" sektion.

Vision

Kunden vil have et spil mellem 2 personer. Spillet skal gå ud på at man slår med to terninger. Efter man har slået med terningen rykker man sin figur på spillebrættet antallet af felter, som man har slået. Værdien af feltet tillægges spillerens konto og viser den opdaterede score. Vinderen er den, der opnår 3000 point først.

Kravliste samt prioritering (MoSCoW)

Funktionelle krav vises med sort skrift
Ikke-funktionelle krav vises med blå skrift

Must have:

- Spillet skal være mellem 2 personer.
- Der slås med 2 terninger
- Terningværdien i et slag svarer til det tilsvarende felt på spilbrættet.
- Points som er tilskrevet det tilsvarende felt, bliver tilføjet til spillerens totalscore.
- Hver spiller starter med 1000 points.
- Vinderen er den der opnår 3000 points.
- Spilleren får et ekstraslag, hvis denne slår terningværdien 10.
- Ved hver tur starter spilleren fra felt nr.0
- Der skrives en tekst som kommentar, når en spiller lander på et felt. I denne står også den vundne/tabte pointværdi.
- Spillet skal kunne spilles på maskinerne i DTU's databarer uden bemærkelsesværdige forsinkelser.

Should have:

- Det skal være let at skifte til andre terninger.
- Spillet skal let kunne oversættes til andre sprog.

Could have:

- Spilleren samt pengebeholdningen skal kunne bruges i andre spil.

Implementation

Roll-knappen

Da vi skulle implementere vores GUI stødte vi på et besynderligt problem. Vi kunne ikke få lov til at lave en permanent knap. Vi kunne dog godt lave knapper der forsvandt når man trykkede på dem. Dette gjorde at vi var nødt til at blive kreative med hvordan vi så kunne implementere en knap der blev på skærmen. vores løsning blev følgende kode.

```
//Creates roll button
public void checkForRoll() {
    gui.getUserButtonPressed( msg: "",  ...buttons: "Roll");

}
```

Figur 1: Metoden vi skrev i GUI_Monopoly

```
//The game loop. is active as long as no player has won
private void gameLoop(){
    GUI_monopoly gui = new GUI_monopoly();

    do{
        gui.checkForRoll();
        gui.clearPlayerCars();

        players[currentPlayerIndex].main();

        gui.movePlayer(players[currentPlayerIndex].getPlayerRoll(),
                gui.getPlayer( playerNumber: currentPlayerIndex+1));

        gui.setPlayerBalance(gui.getPlayer( playerNumber: currentPlayerIndex+1),
                players[currentPlayerIndex].account.getScore() );


        if (players[currentPlayerIndex].CheckWin()) {
            System.out.println("Player " + (currentPlayerIndex + 1) + " won!");
        }

        nextPlayer();
    } while (!CheckForWin(players));
}
```

Figur 2: Koden vi skrev i vores Game klasse

På figur 1 ses en metode som vi lavede. denne meget simple metode gør ikke andet end at kalde en metode fra GUI klassen, som opretter en knap og venter til den bliver trykket på, for så at køre et stykke kode og fjerne knappen. Vi gav den ingen kode at køre, vi udnyttede blot at den venter til knappen trykkes på. På figur 2 kan man se hvordan vi

kalder metoden fra Game klassen. Vi kalder gui.checkForRoll() en gang for hvert loop. Dette gør at en ny knap bliver oprettet i starten af hvert loop. Dette gør i effekt at vi har en knap hver gang der er brug for den. fordi computere i dag er så hurtige og koden så let, opstår illusionen af at knappen ikke forsvinder. det går simpelthen for stærkt mellem at den forsvinder og en ny tager dens plads.

Use-Case beskrivelse; Spil spillet

Use-case Spil Spillet

Scope: Terningspil

Primary actor: Spillerne

Stakeholders and interests:
Spillere: Deres interesse er at vinde spillet samtidig med, at de har det sjovt med at spille spillet.

Main succes scenario:

- Spiller 1 slår terningerne
- bilen rykker sig hen på det rigtige felt
- feltets score lægges til eller trækkes fra spillerens score
- Den anden spiller får turen.

Disse trin gentages til en af spillerne får 3000 points eller vinder på anden vis.

Extensions (alternative flows)
2a hvis man slår 10

- 1. spilleren gives et ekstra kast

2b Når man skal vælge sprog

- 1. man får sit ønskede sprog.

Pre-condition

- Der skal være to spillere

Technology and data restrictions

- Kører på Windows-computere
- Ikke lavet til 4k skærme

GRASP

Klasse typer

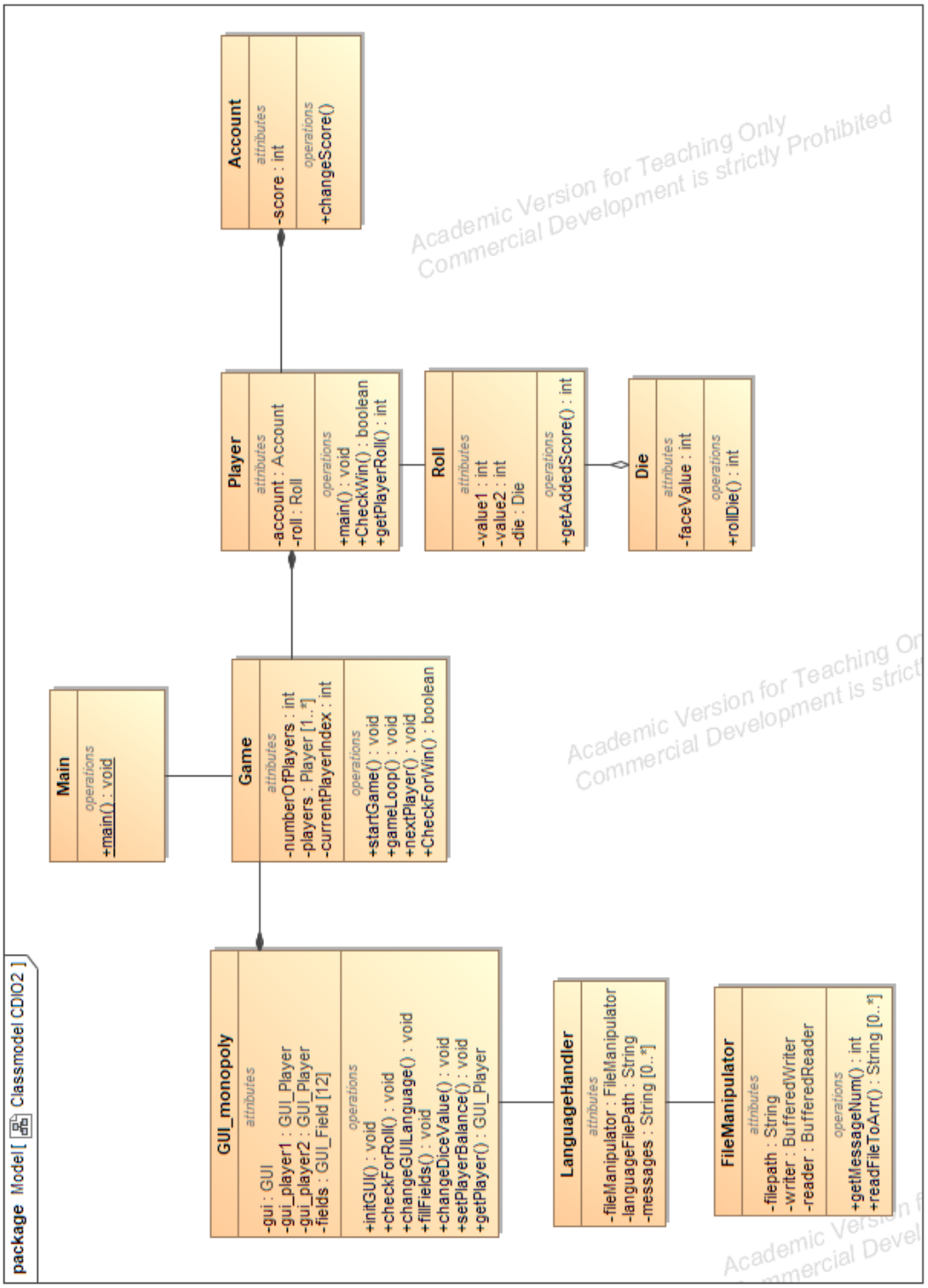
Vi har lært om tre typer af klasser i GRASP modellen. Klasserne

- Creator
- Controller
- Information expert

Vi har haft brugt dette i nogen grad i vores projekt. Vores Main klasse er en creatorklasse som laver Game og sætter spillet op for os. Game klassen er så en controller der binder Player og GUI sammen. En Information expert har vi i Roll klassen som samler information om terningerne der er i spil og samler dem så spilleren kan modtage det som den har brug for.

Lav kobling og høj binding

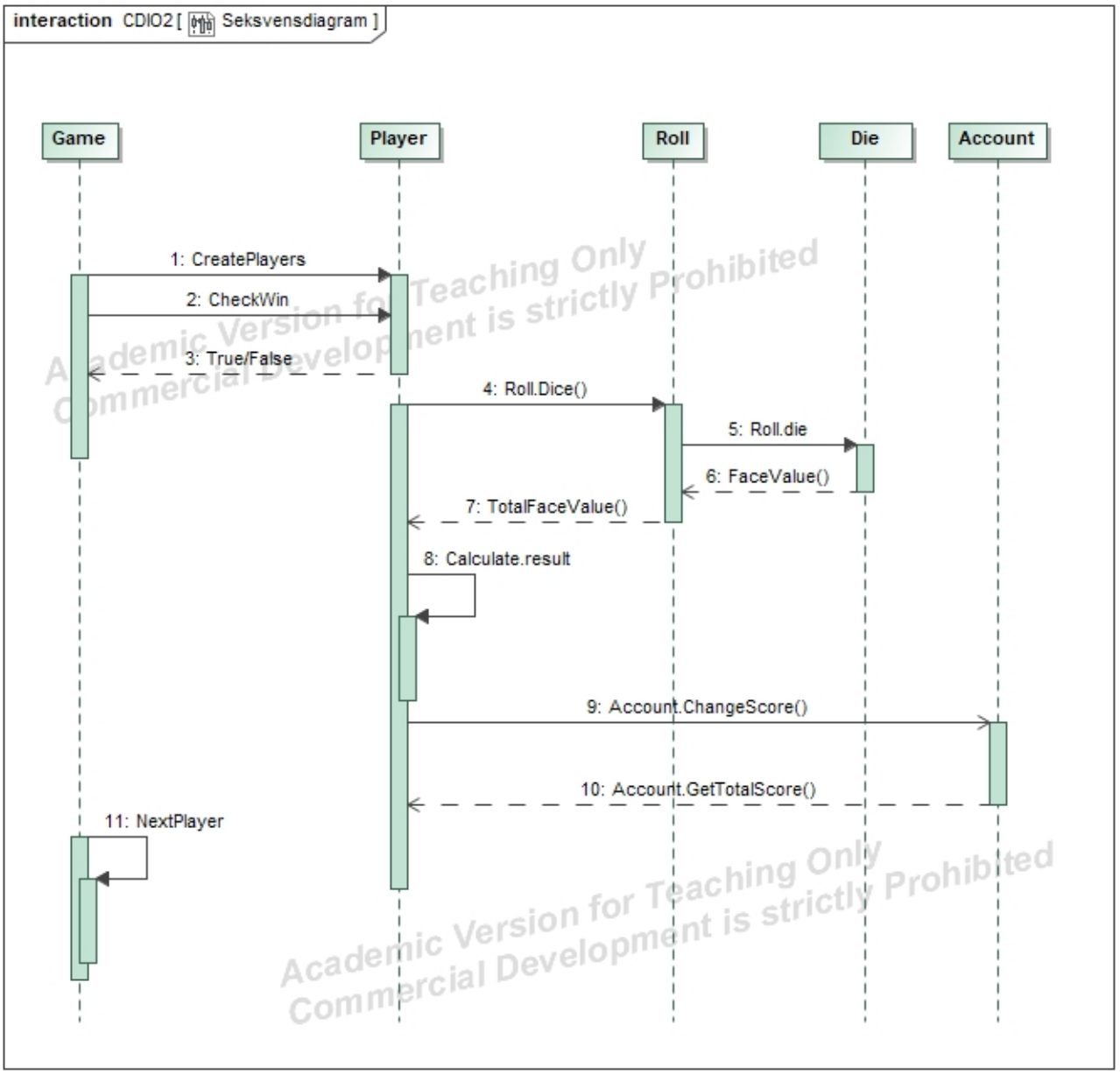
En anden GRASP metode vi har lært er Lav kobling høj binding. Vi har især haft fokus på den lave kobling. Kun to klasser kender mere end tre andre klasser, Game og Player. Dette gør at vi har relativt få koblinger (se eventuelt vores Design klasse diagram på figur 3) vores binding har vi tænkt på men det gik galt ved Player klassen som fik nogle ansvar der i bund og grund ikke hørte til der.



Figur 3: Design klassediagram over vores program

Sekvensdiagram

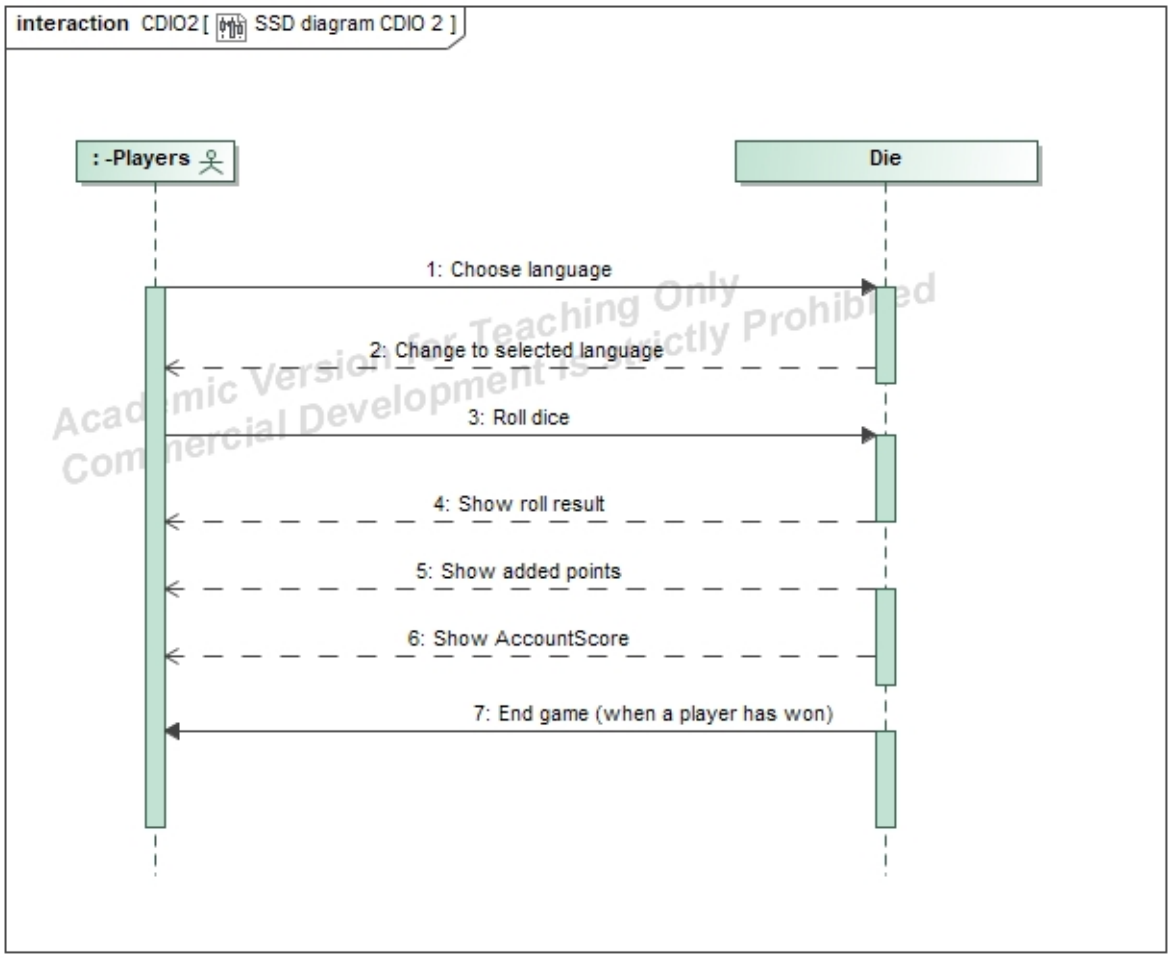
Neden for ses vores sekvensdiagram:



Figur 4: Sekvensdiagram

I dette diagram kan vi se den sekvensen hvorpå vores objekter interagerer hinanden. Dette startes ved at klassen Main oprette et game. Derfor har vi startet diagrammet fra objektet Game.

Systesekvensdiagram



Figur 5: Sekvensdiagram

I denne figur ser vi den rækkefølge en spiller interagerer med spillet på.

Konklusion

Vi har fået udviklet et spil som virker forholdsmæssigt efter hensigten. Vores kunde kan nu spille et "matador-style"spil hvor der slås med 2 terninger og landes et sted på brættet. Dog er det selvfølgelig ikke uden skønhedsfejl, og de følgende punkter er ting vi godt kunne tænke os at addressere en anden gang:

- Ved ekstratur printes dette kun i konsollen, ikke i GUI'en.
- Feltet i midten viser ikke den tekst som svarer til det felt figuren lander på. Lige nu skal man trykke på de individuelle felter for at få teksten frem.
- Vi ved at man ikke kan slå talværdien 1 da man slår to terninger. Men vi har lavet så man godt kan lande på felt 1. Derfor svarer alle værdier til talværdien n-1.
- Ting der skal adresseres: Hvorfor vi ikke kan have Dice og Roll i samme klasse, Hvorfor vi hele tiden kalder en ny "roll"-knap.

Figurer