Baselinedesign

1 Indholdsfortegnelse

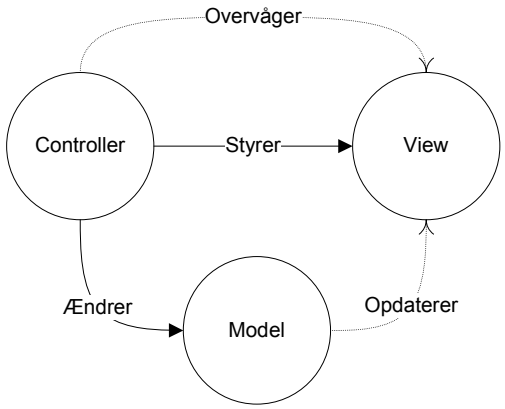
2 Figurfortegnelse

3 Formalia

Projekttitel: Sudoku til undervisningsbrug
Gruppenr.: 4
Deltagere: Studerende: Emil Erik Hansen, Julian Møller, Klaes Bo Rasmussen, Steen Nordsmark Pedersen.
Instruktor: Dennis Franck.
Lærer: Georg Strøm.
Baggrund: Der er blevet bedt om et Sudoku-program, der kan assistere i undervisningen i matematik i 1.-3. klasse.
Formål: At lave et Sudoku-program, der er let at bruge for børn samt kan assistere dem ved løsning af opgaverne.

4 Designparadigme

4.1 Model-View-Controller



Figur 1: Oversigt over Model-View-Controller-paradigmet

Vi har lavet vores design ud fra det klassiske model-view-controller (MVC) paradigme. Controlleren varetager styringen af selve programmet. Den overvåger brugergrænsefladen som ligger i View og sender instrukser til selve "sudokumaskinen", der befinder sig i Model-packagen.

"Figur 1: Oversigt over Model-View-Controller-paradigmet" viser de grundlæggende relationer mellem de tre komponenter. Styrken ved at bruge MVC ligger i de muligheder der er for at adskille programlogik, -kontrol og –grænseflade. Dette medfører både programmeringsmæssig frihed idet de tre komponenter kan interagere alene uden at have nogen idé om hvad der er af bagvedliggende kode i de andre. Dermed er der også mulighed for at skifte enkelte komponenter ud uafhængigt af de andre samtidig med at opdatering af programkode simplificeres – igen pga. seperationen.

4.1.1 Model

Model-komponenten indeholder det faktiske spil, derunder alle metoder til at ændre de underliggende data, herunder f.eks. vores implementation af genereringsalgoritmen. Derudover er der funktionalitet til at finde felter, der kan bruges hvis brugeren beder om at få hjælp til løsningen af Sudokuen. Model indeholder derudover også basale metoder til at udføre f.eks. matematiske operationer på Sudokuplader.

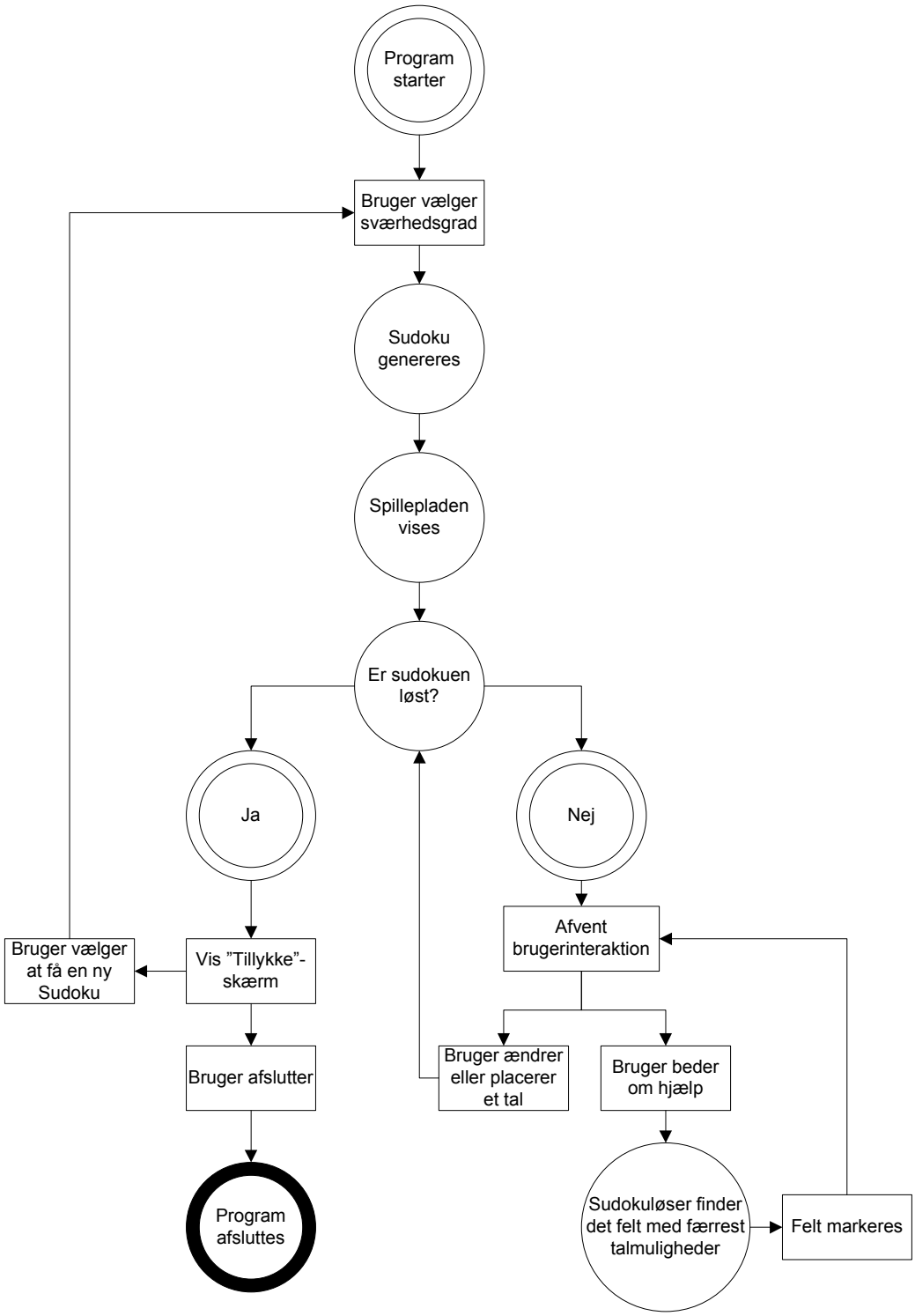
4.1.2 View

View står for at styre programmets visuelle fremtoning. Den indeholder de grafiske repræsentationer af spillepladen, menuer, knapper osv. Controlleren observerer disse elementer. Derudover modtager View-komponenten signaler fra Controlleren når der skal ændres i det grafiske layout. View modtager data fra Model om hvordan spillepladen ser ud og evt. hvilket felt der skal markeres ifm. hjælp.

4.1.3 Controller

Controlleren håndterer brugerens interaktion med grænsefladeelementerne i View-komponenten. Derved kan der udføres forskellige handlinger alt efter om brugeren f.eks. vil placere et tal, have hjælp eller noget helt trejde. Controlleren har også mulighed for at ændre de data der er at finde i Model-komponenten ifm. brugerens handlinger.

5 Programforløb



Figur 2: Oversigt over programmets forløb

På "Figur 2: Oversigt over programmets forløb" ses en oversigt over hvad der foregår når programmet startes. Som det første vises en skærm, hvor brugeren får mulighed for at vælge mellem de forskellige sværhedsgrader. Når brugeren har foretaget sit valg genereres en Sudoku med den valgte sværhedsgrad hvorefter spillepladen vises.

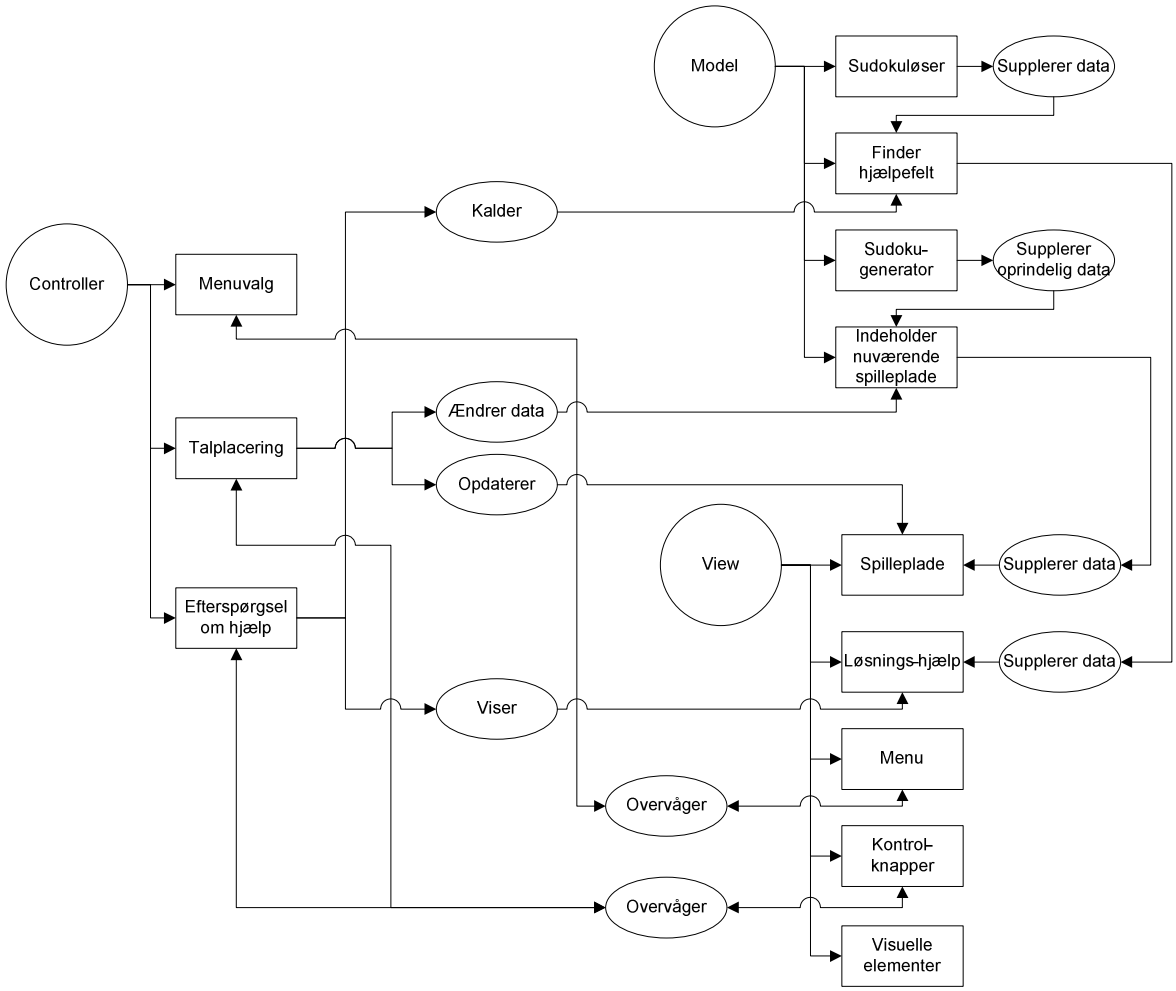
Herefter begynder den primære programløkke, der initialiseres ved at der kontrolleres hvorvidt Sudokuen er løst. Herefter deles programlogikken i to grene:

- Hvis Sudokuen ikke er løst afventes der brugerinteraktion. Brugeren har to muligheder: at bede om hjælp eller at placere/ændre et tal. Hvis der bedes om hjælp startes Sudokuløseren, der finder et løsbart felt hvorefter det markeres. Herefter afventes der igen brugerinteraktion hvor brugeren igen har samme tom muligheder. Vælges der muligheden for at ændre/placere et tal kontrolleres der igen om sudokuen er løst.
- Hvis Sudokuen er løst vises en "Tillykke"-skærm og brugeren har nu muligheden for at få genereret en ny Sudoku eller afslutte programmet. Hvis brugeren vil spille igen bliver vedkommende sendt tilbage til skærmbilledet med valget af sværhedsgrader.

Grunden til at tjekket om Sudokuen er løst ligger allerede dér, er for at simplificere programlogikken idét der dermed kun behøves én primær løkke, der kan vendes tilbage til hver gang der er blevet placeret/ændret et tal på spillepladen.

Under hele forløbet vil brugeren naturligvis også kunne vælge at få en ny Sudoku samt afslutte programmet vha. en menulinje.

6 Interaktion mellem moduler



Figur 3: Grafisk repræsentation af modulernes interne relationer

"Figur 3: Grafisk repræsentation af modulernes interne relationer" viser hvordan modulerne i programmet skal hænge sammen: Hvor komponenter får deres data fra samt hvilke komponenter der overvåger og/eller holder styr på andre komponenter.

6.1 Model

De primære elementer at lægge mærke til er at Sudokuløseren hænger sammen med funktionen til at supplere hjælp (beskrevet yderligere på side 7 i afsnittet "Sudokuløser").

6.2 View

Viewet indeholder alle de grafiske elementer i programmet. "Spilleplade" og "Løsningshjælp" får begge data fra Model-komponenten. Menuen og kontrolknapperne overvåges af Controller-komponenten (og beskrives i næste afsnit).

6.3 Controller

Controller-komponenten tager sig af al programlogikken. Når en bruger interagerer med elementerne i View sendes der signaler til Controller-komponenten der derefter udfører handlinger, baseret på hvilken kontrol der blev aktiveret i grænsefladen.

Det er Controller-komponenten der står for at opdatere data i Model samt at sørge for at de korrekte elementer vises af View-komponenten.

7 Algoritmer

I programmet gøres der brug af to algoritmer: En Sudokuløser samt en Sudokugenerator.

7.1 Sudokuløser

Løseren bruges når brugeren skal bruge vink undervejs. Ved at løse sudokuen et trin ad gangen og derved finde det næste, mest hensigtsmæssige tal at placere kan det fundne felt markeres og vises til brugeren. Vi implementerer vores egen algoritme da de sudokuer vi har med at gøre er så simple, at det ikke kræver en stor matematisk løsning så at løseren kun tjekker for niveau-1-løsninger[1] er tilstrækkeligt.

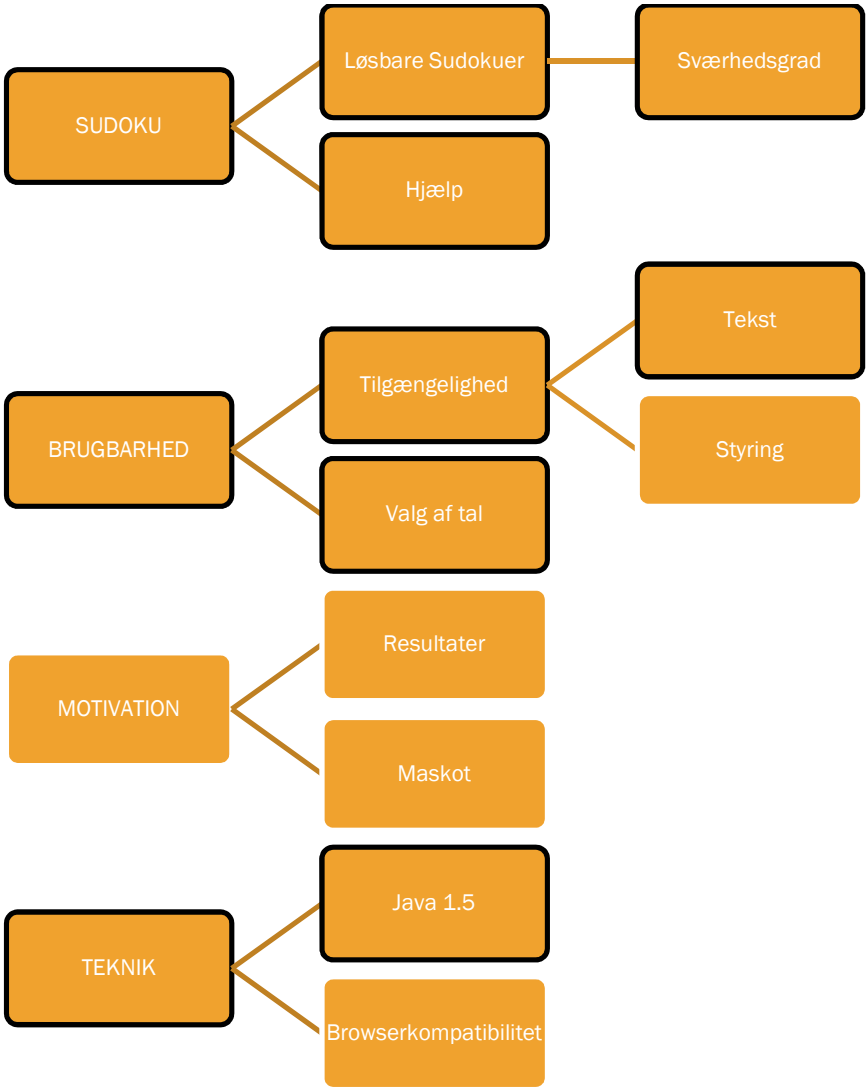
7.2 Sudokugenerator

Sudoku generatoren er i stand til at lave variable sværhedsgrader så børn med forskellige faglige niveauer alle kan blive udfordret. Vi har selv lavet algoritmen ud fra de regler og rammer der er indenfor Sudokuteori. Vi har lavet vores egen da vi ikke kunne finde en open source (og med passende licens) der var nem at omskrive til vores system. Sværhedsgraderne består af forskelligt antal forududfyldte felter. Sudokuernes lave sværhedsgrad (så de passer til brug i 1.-3. klasse) garanteres på følgende måde:

- Først genereres en gyldig Sudokuplade vha. permutation af en standardplade.
- Herefter fjernes tal fra pladen, hvor det dog sikres (vha. løseren) at det fjernede tal kan placeres af brugeren igen udelukkende vha. niveau-1-strategien.
- Kan tallet ikke placeres af brugeren vha. niveau-1-strategien forsøges der med et nyt tal.
- Dette fortsættes indtil det af sværhedsgraden definerede antal tal er fjernet fra pladen.

[1] En niveau-1-løsning angiver at man udelukkende vha. oplysninger om tallene i et felts række, kolonne og kvadrant kan finde feltets værdi.

8 Implementering af baselinekrav



Figur 4: Grafisk repræsentation af kravspecifikationen

Jvf. "Sudokugenerator"-afsnittet genereres der løsbare Sudokuer med valgbar sværhedsgrad.

Brugbarheden kan kun delvist opfyldes af baselinedesignet. Kravet om "Valg af tal" er opfyldt da det gøres muligt at placere nye tal samt erstatte allerede placerede tal. Kravene om "Tilgængelighed" og "Tekst" er ikke mulige at argumentere for i baseline designet da det først er muligt at dokumentere at disse krav er opfyldt når programmet er færdigudviklet. Ligeledes er kravet om "Teknik".

9 Videreudvikling

Idet vi gør brug af MVC-paradigmet er der som tidligere beskrevet nem mulighed for at udvide funktionaliteten i programmet med tiden. Fordelen er nemlig muligheden for at ændre hele komponenter uden at det vedrører de andre. Man kan f.eks. udskifte Model hvis man mener at en

anden måde at generere og løse sudokuer på der er bedre, uden at ændre noget i View eller Controller.

Desuden kan der f.eks. tilføjes en maskot til spillet ved blot at tilføje et element i View-komponenten og opdatere logikken i Controller-komponenten – dvs. uden at røre ved de bagvedliggende algoritmer og spilmekanikker.