Eksamen INF112 Våren 2023

Universitetet i Bergen, Institutt for informatikk

07.06.2023, 15:00–18:00

Generell info om eksamen

- *Oppgaver:* Oppgavesettet består av 4 oppgaver («Oppgave» 5 og 6 skal ikke besvares).
- *Besvare deloppgaver:* Hver oppgave har to eller tre deloppgaver + én bonusoppgave. Del opp svaret på oppgavene slik at det er lett å skille de ulike delene av besvarelsen fra hverandre (du står fritt til å bruke feks overskrifter og punktlister om du ønsker det).
- *Bonusoppgaver:* Hver oppgave har en ekstra deloppgave (merket z). Du kan gjøre disse hvis du har god tid, eller (i praksis) i stedet for andre deloppgaver.
- *Vekting:* Alle fire oppgavene teller 15% hver, og utgjør til sammen 60%, og prosjektet teller de siste 40% av karaktergrunnlaget. Bonusdeloppgavene gir inntil +3%, men du kan likevel ikke få mer enn totalt 60% på eksamen.
- *Hjelpemidler:* Alle skrevne og trykte hjelpemidler er tillatt.
- *Jukselapp:* Blant vedleggene nederst på skjermen finner du følgende fra pensum: pensumoversikten, referat fra gruppepresentasjonene, og Kanban/Scrum boken. Merk: At disse ressursene er vedlagt betyr ikke nødvendigvis at de vil være nyttige for å løse oppgavene. De følger med så du skal slippe å skrive de ut og ta dem med selv.
- **OBS!** Selv om du har tilgang på hjelpemidler/vedlegg, så betyr ikke det at du kan kopiere svarene derfra – vanlige regler for sitering, plagiat og fusk gjelder fortsatt, og du må skrive hele besvarelsen selv.
- *Tid:* Bruk gjerne tid i begynnelsen på å få oversikt over alle oppgavene. Selv om oppgavene er vektet likt, vil noen ta lengre tid enn andre. Pass på å ikke svare for mye på noen oppgaver slik at du slipper opp for tid før du er ferdig! Du har kun tre timer til rådighet.
- *Spørsmål:* Jeg (Anya) stikker innom etter 1–2 timer for å svare på spørsmål om noe er uklart. Les alle oppgavene på forhånd så du vet om du trenger å spørre om noe. Om det er krise kan også eksamensvaktene ta kontakt per telefon.
- *Uklarheter:* Oppfatter du oppgaven som uklar, oppgi hvilke antagelser du gjør i besvarelsen.
- *Programkode:* Hvis du føler behov for å legge ved kode, så finner du en Java-editor i «Oppgave» 6. Du kan klippe/lime derfra til den vanlige tekstboksen, eller evt. tydelig henvise til kodevedlegg i «Oppgave» 6.

Lykke til!

Anya Helene Bagge

1 – Metodikk (a/b – totalt 15% + bonus)

For semesterprosjektet kunne dere selv velge utviklingsmetodikk (Scrum, Kanban, Lean, XP, osv) og teknikker/praksis (f.eks. parprogrammering, continuous integration, TDD/BDD e.l.) dere ville bruke i prosjektarbeidet.

a) [5%]

Forklar kort hva gruppen din ble enige om å gjøre, og hva dere gjorde i praksis.

(F.eks. «Vi planla å bruke Kanban, men det var vanskelig å følge opp tavlen, så utover i semesteret endte vi opp med å gjøre . . . ». Ca 1 avsnitt.)

b) [10%]

Velg enten Scrum eller Kanban og sammenlikn metodikken med hva gruppen gjorde. **Hvilke av metodikkens retningslinjer fulgte dere / hva gjorde dere annerledes? Hva endte dere opp med å tilpasse eller evt. sløyfe?**

(Ca. 1–3 avsnitt.)

z) [bonus, +3%]

Du blir ansatt som INF112 gruppeleder våren 2024. Basert på erfaringene dine og det du har lært, hva slags råd vil du gi 2024-studentene om valg og tilpasning av metodikk? **Skriv 1–3 avsnitt der du forklarer og anbefaler en passende metodikk for en INF112-gruppe.**

Svaret skal være skrevet slik at det vil være forståelig for en fersk INF112-student.

2 – Testing (a/b – totalt 15% + bonus)

Figure 1: PeripheryPlanet – et kolonisimuleringsspill

Etter studiene har du fått jobb som spillutvikler i et lite selskap. Dere lager en kolonisimulator – *PeripheryPlanet* – der spilleren skal administrere en mengde (ganske hjelpeløse) kolonister som prøver å overleve på en avsidesliggende planet under primitive forhold. Spillet foregår på et relativt enkelt 2D-kart, og kolonistene må dyrke mat, bygge hus, og beskytte seg mot ville dyr og aggressive naboer. Du er ansvarlig for QA (Quality Assurance / testing).

Relevante biter av koden til spillet er lagt ved (`HumanColonist.java.pdf`, `PathFindingTest.java`), i tilfelle det hjelper deg å forstå ting bedre – men det er ikke nødvendig å lese koden for å løse oppgavene.

a) [5%]

Forklar kort hvilke former for testing (unit testing, etc) spillet bør gjennom før det er klart for salg, og om det er noe du mener er ekstra viktig eller mindre viktig.

b) [10%]

Spillet blir fort veldig populært og blir mye diskutert på sosiale medier. Selv om spillerne generelt er fornøyde, klager de på flere ting i spillet, blant annet at stifinningsalgoritmen er dårlig: kolonistene ser ut til å foretrekke å bevege seg på skrå, og benytter seg ikke av de fine brolagte veiene som spilleren bygger. *(Se vedlagt skjermbilde av Reddit-post (fig.1))*

Dere mangler dessverre gode tester for stifinning – forrige QA-ansvarlige hadde laget noen tester som satte en spillfigur på kartet, ba den gå til et gitt sted, og deretter sjekket ruten den fulgte, men testene feilet allerede i `@BeforeEach`-metoden hvor spillfiguren ble opprettet (`new HumanColonist(x,y)`):

```
java.lang.NullPointerException: Cannot invoke "com.badlogic.gdx.Files.internal(String)"
because "com.badlogic.gdx.Gdx.files" is null
    at com.badlogic.gdx.graphics.Texture.<init>(Texture.java:110)
    at inf112.peripheryplanet.pawns.HumanColonist.<init>(HumanColonist.java:22)
    at inf112.peripheryplanet.test.PathFindingTest.setupBeforeEach(PathFindingTest.java:27)
```

Du skjønner at konstruktøren prøver å laste inn et bilde (og feiler), så du prøver å gi den et tomt bilde i stedet, men da får du en annen (verre?) feilmelding:

```
java.lang.UnsatisfiedLinkError: 'java.nio.ByteBuffer com.badlogic.gdx.graphics
.g2d.Gdx2DPixmap.newPixmap(long[], int, int, int)'
    at com.badlogic.gdx.graphics.g2d.Gdx2DPixmap.newPixmap(Native Method)
    at com.badlogic.gdx.graphics.g2d.Gdx2DPixmap.<init>(Gdx2DPixmap.java:136)
    at com.badlogic.gdx.graphics.Pixmap.<init>(Pixmap.java:137)
    at com.badlogic.gdx.graphics.Texture.<init>(Texture.java:138)
    at inf112.peripheryplanet.pawns.HumanColonist.<init>(HumanColonist.java:21)
    at inf112.peripheryplanet.test.PathFindingTest.setupBeforeEach(PathFindingTest.java:27)
```

Siden du prøver å teste stifinning, og ikke grafikken, tenker du det er unødvendig å stresse med å sette opp grafikksystemet og laste inn bilder for å opprette testobjektene.

Hvilke løsninger kan du se for å teste stifinning og spillfigurenes oppførsel generelt, på en måte som er uavhengig av grafikksystemet? **Forklar, og nevn gjerne flere muligheter.**

OBS! Figuren er bare en illustrasjon, og du trenger ikke løse/finne ut av selve stifinningsproblemet

Figure 2: Stifinningen er rar og kolonistene går i sikksakk

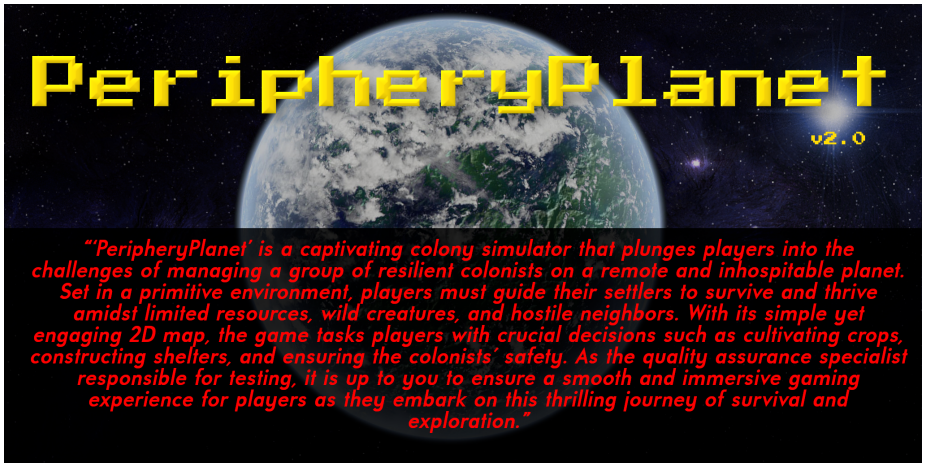
z) [bonus, +3%]

Brukerne fortsetter å klage på dårlig stifinning, og på Reddit blir det fremmet krav om at *PeripheryPlanet* bør ha valgfri stifinningsalgoritme: den vanlige/dårlige, som er laget for å kjøre raskt, og en som gir perfekt resultat, men gjør spillet tregere. Da blir alle fornøyde – de med eldre datamaskin kan fortsatt spille, mens de med ny, kraftig datamaskin kan nyte bedre stifinning. *(Se vedlagt skjermbilde av Reddit-kommentarer (fig.2))*

Sjefen din synes dette er en genial idé, og foreslår at spille utvides med et helt sett av forskjellige konfigurerbare valgmuligheter for spillfigurenes oppførsel. Du tenker at det kanskje vil gjøre jobben din vanskeligere. **Hvorfor kan dette være problematisk for kvalitetssikringen? Er det noe spesielt du bør passe på – eventuelt, noe som kan gjøre jobben lettere? Forklar.**

(Vedlegg: `HumanColonist.java.pdf`, `PathFindingTest.java`)

3 – Programmering og Design Patterns (a/b/c – totalt 15% + bonus)



La oss titte litt mer på implementasjonen av *PeripheryPlanet*.

a) [5%]

Se på grensesnittet `PathFinder.java` i vedlegget. Stifinning skjer ved å lage et `PathFinder`-objekt, kalle forskjellige metoder og så kalle `calculate()`. Hvorfor tror du det er gjort slik? Ville det ikke vært bedre å bare ha en `pathFind()` metode som returnerer stien direkte, i stedet for å gå via et eget `PathFinder`-objekt? **Forklar.**

Kjenner du igjen denne teknikken som en *design pattern* (mønster)? I såfall, **hva heter mønsteret?**

b) [5%]

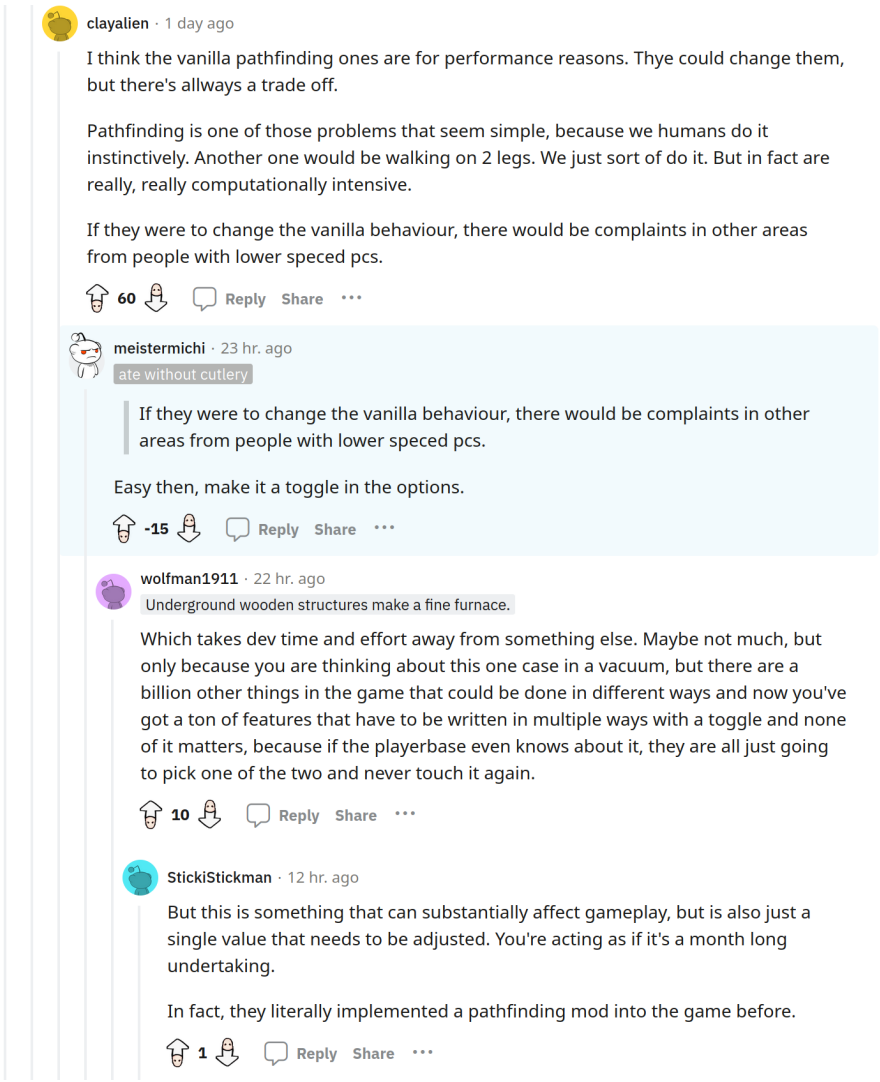


Figure 3: Brukere klager over stifinning, vil ha konfigurerbar algoritme

Design-teamet vil gjerne gjøre oppførselen til kolonistene og de andre spillfigurene (*«pawns»*) litt mer variert og interessant, og trenger en mer fleksibel utgave av stifinningen. De vil gjerne...

- at forskjellige typer pawns kan ha litt forskjellige regler for hvordan de beveger seg. F.eks., fugler kan fly og blir ikke hindret av sperringer; ender kan svømme og blir ikke hindret av vann; smådyr er sky og vil holde seg unna mennesker; barn er redde for mørke kroker osv.
- at ting som man har på seg også kan utgjøre en forskjell – f.eks., hvis man har rulleskøyter, så kan man bevege seg ekstra raskt, men bare på asfaltert vei

Dette burde kunne ordnes ved (blant annet) å justere kostnadene i stifinningsalgoritmen. F.eks., «rulleskøyter på vei» har 25% av kostnaden til «vasse i bekken», så da vil spillfigurene foretrekke veien med lavest kostnad. Design-teamet har foreslått en endring til `PathFinderImpl.java` (se vedlegg) som gjør dette, men koden ble ganske rotete (se forskjell på OLD og NEW `calculateCostForMapCell`).

Du titter på koden og ser raskt at dette bryter med alt du har lært i INF112, blant annet om SOLID-prinsippene. **Hvilke(t) SOLID-prinsipp(er) tenker du blir brutt i den nye `calculateCostForMapCell`? Forklar.**

c) [5%]

Hva ville vært en bedre løsning for å justere oppførselen til stifinneren avhengig av figuren som beveger seg? Finnes det design patterns som passer for dette, eller har du sett liknende problemer tidligere? **Skissér en løsning.** Du kan vise kode hvis det gjør det lettere å forkare, men det er ikke nødvendig.

z) [bonus, +5%]

Spillet ser ut til å lagre kartet som et vanlig Java hash map indeksert med 2D-vektorer – `Map<Vector2,MapElement>`.

- `Vector2`-klassen (fra `com.badlogic.gdx.math`) er *muterbar*, dvs. feltvariablene kan endres etter at objektet er opprettet. Kan det skape problemer slik som kartet er representert? **Forklar kort**
- Ville du valgt et annet API for kartet om du kunne velge? **Forklar kort**

(Vedlegg: `PathFinder.java`, `PathFinderImpl.java`, `MapCell.java`)

4 – Helseproblemer (a/b – totalt 15% + bonus)

Helse Midt-Norge har siden 2015 jobbet med å få på plass et nytt, felles pasientjournalsystem, pasientadministrasjon og helseportal for innbyggerne i Midt-Norge (ca. 700000 innbyggere). Arbeidet ledes av *Helseplattformen AS*, et offentlig eiet selskap med ca. 320 ansatte. I 2019 signerte de kontrakt med det amerikanske selskapet *Epic Systems* som utvilker journalsystemet som er basis for Helseplattformen. Epics system dekker 78% av amerikanske pasienter – systemene deres er også innført i andre land, bla. Danmark og Finland, med

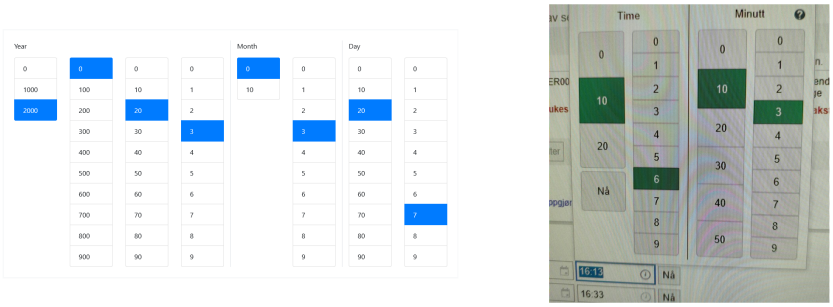


Figure 4: Absurd dato/klokkeslett-velger fra Helseplattformen (Skjermbilde, privat)

varierende hell. Selv om Epic har laget den underliggende teknologien, har Helseplattformen selv gjort betydelig utviklingsarbeide for å tilpasse til norske forhold og brukere: *«På Epics «foundation system», som er plattformen med alt det grunnleggende innholdet i journalløsningen, er det bygd et stort antall applikasjoner og integrasjoner etter spesifikasjoner fra helsetjenesten i regionen.» (Fra helseplattformen.no)*

Helseplattformen har skapt en god del avisoverskrifter i løpet av våren (se diverse avisklipp under). Opprinnelig budsjett for innføringen var på ca. 3,3 mrd NOK (hvorav 1,2 mrd til Epic) – men hittil er det brukt over 4 mrd kroner, uten at innføringen er ferdig. St. Olavs hospital i Trondheim har ekstrakostnader på 25 millioner i uken og må kutte i psykisk helse.

Helsepersonell klager på at systemet er tungrodd og vanskelig å bruke:

- Viktige henvisninger, blant annet for kreftpasienter, har ikke blitt sendt fordi legene ikke har skjønt meldingssystemet. Helseplattformens nå avgåtte direktør skyldte på «brukerfeil».
- Pasienter har fått feile medisiner eller feil dose
- En pasient er død av slag etter feil i journalføring
- Helsepersonell blir slitne, stresset og har lettere for å gjøre feil

(Systemet har også sine styrker – bla. med analyse av helsedata, og nyttige funksjoner som å advare om et nytt legemiddel til en pasient er i konflikt eksisterende medisiner eller sykehistorie. Dette er noe legene har måttet gjøre manuelt tidligere.)

a) [7%]

Bildet øverst viser brukergrensesnittet for å velge dato og klokkeslett (dato: 2000+0+20+3, 0+3, 20+7 → 2023-03-27; tid: 10+6, 10+3 → 16:13).

Hvis du tenker tilbake på hva du har lært om å utarbeide krav, spesifikasjon, brukerhistorier, akseptansekriterier, arbeidsoppgaver osv. – **WTF??** Hva kan ha gått galt i utviklingsprosessen til dato/tid-velgeren? **Forklar kort.**

(For akkurat denne oppgaven kan du dette ikke er den foretrukne måten å håndtere dato/tid på for helsepersonell i Midt-Norge.)

b) [8%]

Helseplattformen bygger på et anerkjent, ferdig utviklet system, støttet av tusenvis av utviklere (Epic har 10000 ansatte). Helseplattformen AS har ca. 320 ansatte som har jobbet med prosjektet gjennom flere år, i samarbeid med hundrevis av fageksperter (både medisinsk, administrativt og IT-faglig) fra helseforetakene og kommunene (se sitat under – *Slik taes beslutninger i felleskap*). De har laget kravspesifikasjon, hatt ukentlige møter med faglig ledergruppe, og et budsjett på flere milliarder kroner. Likevel har de støtt på store problemer når systemet skulle taes i bruk.

Som programvareutvikler, hva slags råd ville du gitt til helseforetak som skal utvikle/anskaffe liknende systemer? Hva tenker du er viktigst å legge vekt på i utviklingsprosessen? **Forklar kort.**

z) [bonus, +3%]

Hva tenker du selv om gjennomføring av store IT-prosjekter i samfunnet? Er det umulig å gjøre en god jobb, eller kommer problemene av dårlige avgjørelser? Kunne INF112-kullet våren 2023 gjort en bedre jobb? **Forklar kort hva du selv mener.**

Vedlegg

Vedleggene er bare til illustrasjon, det er ikke nødvendig å studere dem nøye.

Sitat fra Helseplattformen.no → Om Oss → Prosjektet:

> Slik taes beslutninger i fellesskap
>
> Det er helsepersonell som tar beslutninger om utviklingen av journalløsningen. En faglig beslutningsstruktur er etablert for å ta felles beslutninger om hvordan løsningen skal settes opp og brukes. Her har over 400 fageksperter deltatt fra alle fagområder i helsetjenesten. Over 220 fageksperter har vært involvert fra mer enn 20 kommuner, de fleste fra Trondheim kommune. Fra helseforetakene er det enda flere, med overvekt fra St.Olavs hospital.
>
> De fleste fagekspertene er helsepersonell, men det er også egne team for data og IKT, administrative oppgaver og så videre. Den faglige beslutningsstrukturen fatter faglige beslutninger som berører helse, data og teknlogiske problemstillinger i løsningen.
>
> Det samme prinsippet ble brukt i anskaffelsesprosjektet, der helsepersonell fra hele regionen deltok i arbeidet med kravspesifikasjon og evaluering av tilbud i konkurransen om å levere journalløsningen.

Det fagekspertene ikke har blitt enige om, det som ligger utenfor mandatet og beslutninger som vil utløse store endringer eller kostnader, løftes videre til Felles beslutningsgruppe. Dette er en faglig ledergruppe med ukentlige møter, der alle organisasjonene som skal ta i bruk løsningen er representert. Denne strukturen skal bestå også etter innføring slik at videreutvikling av løsningen bestemmes i fellesskapet.

I den faglige beslutningsstrukturen inngår også flere råd som er oppnevnt av Felles beslutningsgruppe.

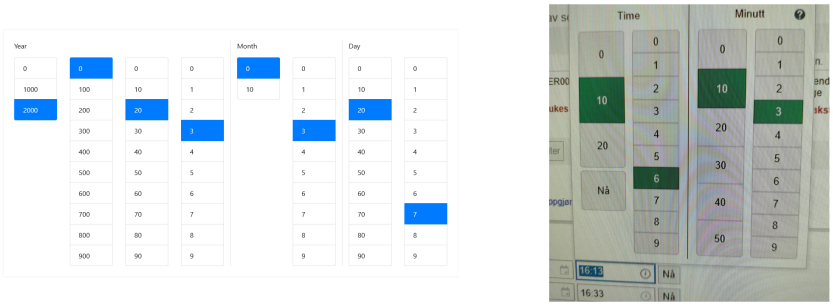


Figure 5: Slik velger man (angivelig) dato/klokkeslett i brukergrensesnittet til Helseplattformen (Skjermbilde, privat/Nils Ivar Leraand)

LEDER

Helseplattformen – en IT-skandale i Midt-Norge

Monica Engstrøm *Om forfattaren*

Når er en sak så alvorlig at leger sender e-post til statsråder og stortingsrepresentanter? Jo, når pasientsikkerheten i en hel helseregion er truet, helsetjenestens kapasitet nedskaleres med viten og vilje og penger og prestisje veier tyngst.

Den 12. november 2022 ble journalsystemet Helseplattformen, levert av amerikanske Epic Systems, innført ved St. Olavs hospital. Vi har nå fått et system som er usedvanlig lite brukervennlig. Dette viktigste verktøyet for drift og dokumentasjon som et sykehus har, bidrar derimot til truet pasientsikkerhet, redusert effektivitet og enorme kostnader. Hvordan kunne dette skje?

Helseplattformen var et pilotprosjekt i Midt-Norge før et eventuelt landsdekkende felles journalsystem. I tillegg var målene med Helseplattformen lettere informasjonsflyt gjennom hele helsetjenesten samt bedring i effektivitet, ressursbruk, kvalitet og pasientsikkerhet (1). For flere år siden ble det klart at resten av landet ikke skal bruke Helseplattformen. Fastleger i Midt-Norge vil ikke ha systemet fordi det gir ineffektiv drift og stort inntektstap. Målene ble altså borte, men systemet ble innført likevel. Og det virkelig alvorlige er at tapet av effektivitet truer pasientsikkerheten. Da Epic ble valgt som leverandør, forelå det god dokumentasjon på problemer i andre land som Danmark, Finland og Nederland. Vi har nå fått et dyrt IT-system fra 1990-tallet tilpasset et amerikansk samfunn og helsevesen.

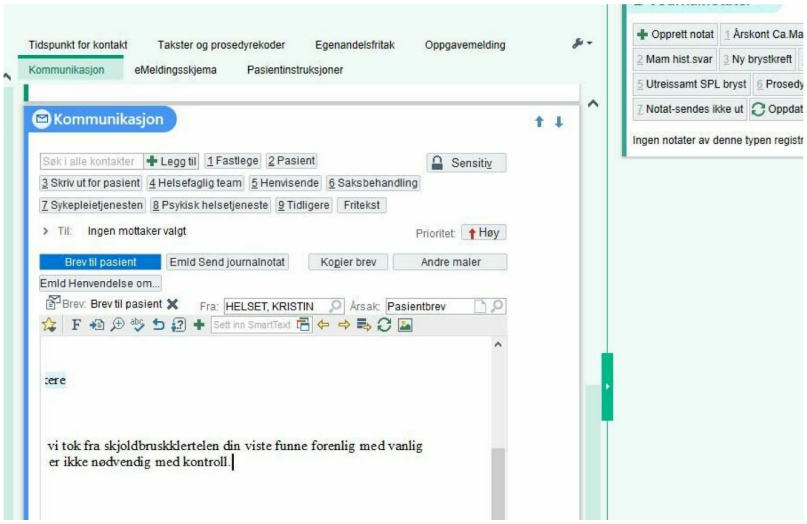
«Vi var alvorlig bekymret på forhånd, og det ble verre kaos enn fryktet»

Skylder på brukerne for at over 16.000 brev ikke kom frem

Direktøren for Helseplattformen sier det er brukerfeil og ikke systemsvikt som er årsaken til at over 16.000 elektroniske brev ikke kom fram dit de skulle. Legeforeningen mener tilliten til styret er tynnslitt.

Slik skriver du brev i Helseplattformen: – Jeg vet fortsatt ikke om jeg gjør det riktig

St. Olav-overlege Kristin Helset forteller at hun ikke får tilbakemelding på om meldinger har gått til rette mottakere: – Det er en diger skjerm med mye skrift, hvor det meste ikke brukes, sier hun. Løsningen får slakt av designeksperter.



Helseplattformen

– Vi har opplevd at pasienter har fått medisiner det ikke er ment at de skal ha

Det nye IT-systemet som er innført i Helse Midt-Norge, Helseplattformen, skaper store problemer ved St. Olavs Hospital i Trondheim.

Norge | Nytt journalsystem i Midt-Norge

Omstridt datasystem tapper landets fjerde største sykehus for penger – rammer satsing på psykisk helse

Problemene med det omstridte journalsystemet Helseplattformen gir St. Olavs hospital i Trondheim ekstrakostnader på 15–25 millioner kroner i måneden. Nå må sykehuset utsette planer å bygge et senter for psykisk helse.

Pasient døde av slag: Fylkeslege kobler dødsfallet til Helseplattformen

Mangelfull journal kan ha ført til at en pasient døde ved St. Olavs hospital. Sykehusdirektøren tar saken svært alvorlig, og har varslet Helsetilsynet.

DØDSFALL: En pasient døde av slag på St. Olav i desember. Det er lite dokumentasjon i journalen og dermed vanskelig å vite hva som skjedde, sier fylkeslegen.

FOTO: BENT LINDSETMO / NRK

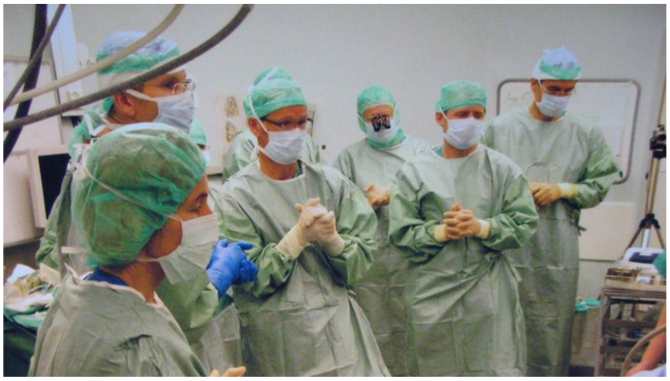
Marthe Svendsen
Journalist

Grete Thobroe
Journalist

Publisert 13. jan. kl. 17:02
Oppdatert 13. jan. kl. 19:33

Datasystem som skal innføres i Norge er utskjelt i Danmark: – Det er helt gak-gak

Helse Midt-Norge kaller et nytt elektronisk journalsystem som skal innføres «et stort teknologisk løft». I Danmark har det gitt pasienter feil dose medisin og beskrives som vanskelig å bruke.



Leger ved St. Olavs hospital i Trondheim blir de første som tar i bruk den nye elektroniske pasientjournalen som blir kalt Helseplattformen. Danske kolleger advarer sterkt mot datasystemet som skal brukes.

FOTO: ST.OLAVS HOSPITAL

Kjartan Rørslett
Journalist

Publisert 22. feb. 2019 kl. 10:46
Oppdatert 22. feb. 2019 kl. 13:03

Helvetesplattformen

En oversikt over skriverier om Helseplattformen.

Send inn tips!

Alle nyheter

16. mai 2023 — Aftenposten

Skulle koste 3,7 milliarder. Regningen bikker 4 milliarder før systemet er 30 prosent innført.

25. april 2023 — NRK

Riksrevisjonen starter undersøkelse av Helseplattformen

22. april 2023 — NRK

Nær 500 legar har signert bekymringsbrev om Helseplattformen

17. april 2023 — Statens helsetilsyn

Rapport fra tilsyn ved St. Olavs hospital etter innføring av Helseplattformen

05. april 2023 — Dagens Medisin

Brukervennlighet anbefales som eget fokusområde i revisjon av Helseplattformen

31. mars 2023 — Dagens Medisin

Helseplattformen forvirret lege - ble usikker på om medisin var seponert

28. mars 2023 — Dagens Medisin

En bærekraftig plattform?

28. mars 2023 — Dagens Medisin

Hvor mange interaksjonsdesignere trengs for å sette en medisinsk diagnose?

27. mars 2023 — NRK

Kritisk løsning manglet i journalsystem – Rystet på vegne av pasientene

27. mars 2023 — Digi

24. mars 2023 — Digi

23. mars 2023 — Aftenposten

HumanColonist.java

```java
package inf112.peripheryplanet.pawns;

import java.util.List;
import java.util.Map;

import com.badlogic.gdx.Gdx;
import com.badlogic.gdx.graphics.Pixmap;
import com.badlogic.gdx.graphics.Pixmap.Format;
import com.badlogic.gdx.graphics.Texture;
import com.badlogic.gdx.graphics.g2d.SpriteBatch;
import com.badlogic.gdx.math.Vector2;

import inf112.peripheryplanet.map.MapCell;
import inf112.peripheryplanet.map.PathFinder;

public class HumanColonist implements Pawn {
    private Texture image;
    private Vector2 position;
    private Vector2 target;

    public HumanColonist(int x, int y) {
        this.position = new Vector2(x, y);
        //this.image = new Texture("colonist.png");
        this.image = new Texture(100, 100, Format.RGB888);
    }

    public void setTarget(int destX, int destY) {
        target = new Vector2(destX, destY);
    }

    public void step(Map<Vector2,MapCell> world) { // do single timestep
        if (target != null) {
            if (position.equals(target)) { // are we there yet?
                target = null; // yes!
            } else {
                List<Vector2> path = PathFinder.finder(world).from(position).to(target).calculate();
                if (path != null) {
                    // take one step & update map
                    world.get(position).pawn(null);
                    position = path.remove(0);
                    world.get(position).pawn(this);
                } else {
                    // ??? target is unreachable
                }
            }
        }
        // ... or do something else
    }

    public void draw(SpriteBatch batch) {
        batch.draw(image, position.x, position.y);
    }

    public Vector2 position() {
        return position;
    }
}
```

PathFindingTest.java

```java
package inf112.peripheryplanet.test;

import static org.junit.jupiter.api.Assertions.assertEquals;

import java.util.HashMap;
import java.util.List;
import java.util.Map;

import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

import com.badlogic.gdx.math.Vector2;

import inf112.peripheryplanet.pawns.HumanColonist;
import inf112.peripheryplanet.pawns.Pawn;


public class PathFindingTest {

    private HumanColonist player;
    private List<Vector2> correctPath = List.of(new Vector2(2, 2),
            new Vector2(2, 3), new Vector2(2, 4), new Vector2(3, 5));
    private Map<Vector2,Pawn> world;

    @BeforeEach
    void setupBeforeEach() {
        player = new HumanColonist(2, 2);
        world = new HashMap<>();
        world.put(player.position(), player);
    }

    @Test
    void testPathFinding() {
        player.setTarget(3, 5);
        for(Vector2 step : correctPath) {
            assertEquals(step, player.position());
            player.step(world);
        }
    }
}
```

PathFinder.java

```java
package inf112.peripheryplanet.map;

import java.util.List;
import java.util.Map;

import com.badlogic.gdx.math.Polygon;
import com.badlogic.gdx.math.Vector2;

public interface PathFinder {

    /** create a new path finder */
    static PathFinder finder(Map<Vector2, MapCell> map) {
        return new PathFinderBaseImpl(map);
    }

    /** where to start from */
    PathFinder from(Vector2 pos);

    /** where we want to go to */
    PathFinder to(Vector2 pos);

    /**
     * a waypoint we want to include in the path (can be given multiple times)
     */
    PathFinder via(Vector2 pos);

    /**
     * a (dangerous?) area we should avoid (can be given multiple times)
     */
    PathFinder avoid(Polygon area);

    /** prefer the shortest path */
    PathFinder shortest();

    /** prefer the safest path */
    PathFinder safest();

    /** prefer the fastest path */
    PathFinder fastest();

    /**
     * calculate and return the path (null if destination is unreachable)
     */
    List<Vector2> calculate();
}
```

PathFinderImpl.java

```java
package inf112.peripheryplanet.map;

import java.util.ArrayList;
import java.util.List;
import java.util.Map;

import com.badlogic.gdx.math.Vector2;

import inf112.peripheryplanet.pawns.Bird;
import inf112.peripheryplanet.pawns.Cat;
import inf112.peripheryplanet.pawns.Dodo;
import inf112.peripheryplanet.pawns.Duck;
import inf112.peripheryplanet.pawns.Pawn;

// pathfinder algorithm implemented here
public class PathFinderImpl extends PathFinderBaseImpl implements PathFinder {
    public PathFinderImpl(Map<Vector2, MapCell> map) {
        super(map);
    }

    // OLD: pathfinder will prefer cells with *low* cost
    protected double calculateCostForMapCell(MapCell mapCell) {
        if(mapCell == null)
            return 1; // default cost
        else if(mapCell.pawn() != null)
            return Double.POSITIVE_INFINITY; // already occupied
        else
            return mapCell.terrain().movementCost();
    }

    // NEW: pathfinder will prefer cells with *low* cost
    protected double calculateCostForMapCell(MapCell mapCell, Pawn pawn) {
        if(mapCell == null)
            return 1; // default cost
        else if(pawn instanceof Bird && !(pawn instanceof Dodo))
            return 1; // birds (except dodos) can fly, ignore terrain
        else if(mapCell.pawn() != null && !(pawn instanceof Cat)) // cats can sneak past other pawns
            return Double.POSITIVE_INFINITY; // already occupied
        else if(mapCell.terrain() instanceof Water && pawn instanceof Duck) // can swim
            return 1;
        else if(pawn.footwear() instanceof RollerSkates && mapCell.terrain() instanceof Paved)
            // roller skates are extra fast on paved terrain
            return mapCell.terrain().movementCost() / 2;
        else
            return mapCell.terrain().movementCost();
    }

    protected List<Vector2> calculatePath() {
        List<Vector2> result = new ArrayList<>();
        // ...
        return result;
    }

}
```

MapCell.java

```java
package inf112.peripheryplanet.map;

import java.util.List;

import inf112.peripheryplanet.pawns.Pawn;
import inf112.peripheryplanet.terrain.Terrain;

public interface MapCell {
    /** terrain at this location (determines movement cost) */
    Terrain terrain();

    /**
     * the pawn (colonist, animal, etc) currently at this location (only one
     * allowed!)
     */
    Pawn pawn();

    /**
     * replace the pawn (colonist, animal, etc) currently at this location (only one
     * allowed!)
     */
    Pawn pawn(Pawn newPawn);

    /** other items at this location (latest added first) */
    List<MapElement> items();

}
```