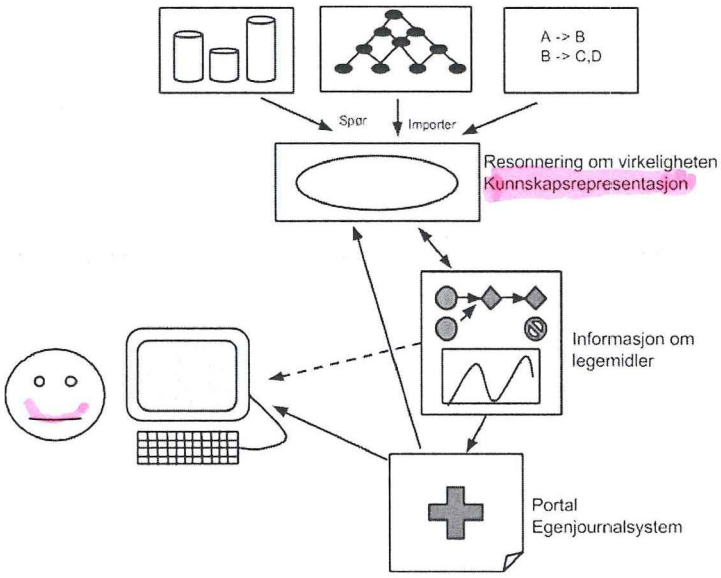
Kapittel 6

Teknologi

For å kunne presentere nyttig informasjon om legemidler for pasienter må det finnes informasjon å vise frem, og en måte å vise den frem på. Dette krever teknologi. Figur 6.1 viser en modell for et system for å presentere personlig informasjon om legemidler, samt eksterne moduler modellen baserer seg på og integreres med. Det illustreres hvordan en kunnskapsrepresentasjon kan skape det nødvendige datagrunnlaget for å presentere personlig informasjon om legemidler til pasienter. En kunnskapsrepresentasjon kan bygges opp av flere eksisterende datakilder med informasjon om pasienter og legemidler. Figuren viser også hvordan presentasjon av informasjon om legemidler kan integreres med eksisterende informasjonssystemer rettet mot pasienter.



Figur 6.1: Oversiktsbilde av en modell for et system for å vise legemiddelinformasjon til pasienter

Det er nødvendig med en kunnskapsrepresentasjon for legemidler for å kunne vise personlig informasjon om legemidler. I dette kapittelet blir det sett nærmere på Semantisk Web som teknologi for å strukturere data, og muliggjøre resonnering.

Oppbygging av en kunnskapsrepresentasjon om legemidler krever data. For å kunne vise personlig informasjon om legemidler må kunnskapsrepresentasjonen inneholde data om den enkelte pasient, samt om legemidler generelt. En viktig kilde til informasjon om den enkelte pasient er elektroniske journalsytemer. Informasjon om hvilke legemidler som markedsføres i Norge finnes, men informasjonen er dårlig strukturert. Internasjonalt er det gjort flere forsøk på å bygge ontologier om legemidler og virkestoffer.

Det finnes mange elektroniske journalsystemer. Noen av disse er egenjournalsystemer som gjør at pasienten får tilgang til informasjon fra journalsystemet. I noen tilfeller kan pasientene også endre eller legge til informasjon i egenjournalsystemer. Kanskje kan journalsystemer integreres slik at brukerne bare trenger å forholde seg til ett system. Ved å integrere informasjon om legemidler i eksisterende egenjournalsystemer spares arbeid med å bygge opp en egen portal med eget påloggingssystem.

6.1 Semantic Web

Semantisk Web er mye brukt i hverdagen. Eksempler på bruk av Semantisk Web finnes i Facebook sin "Like"-knapp, Google sine "snipplets" og Apple sin taleapplikasjon Siri [200].

Begrepet Semantisk Web kan brukes om tre ulike deler: visjon, prosjekter og teknologi.

Visjon

Bakgrunnen for Semantisk Web er basert Berners-Lee, Hendler, og Lassila visjon[201] om at:

> "The Semantic Web will bring structure to the meaningful content of Web pages, creating an environment where software agents roaming from page to page can readily carry out sophisticated tasks for users."

Målet med Semantisk Web er å gjøre data på Internett maskinlesbar. Ved hjelp av Semantisk Web skal maskiner klare å tolke og hente nyttig informasjon som er lenket sammen. Semantisk Web er en utvidelse av Internett [202].

Projekter

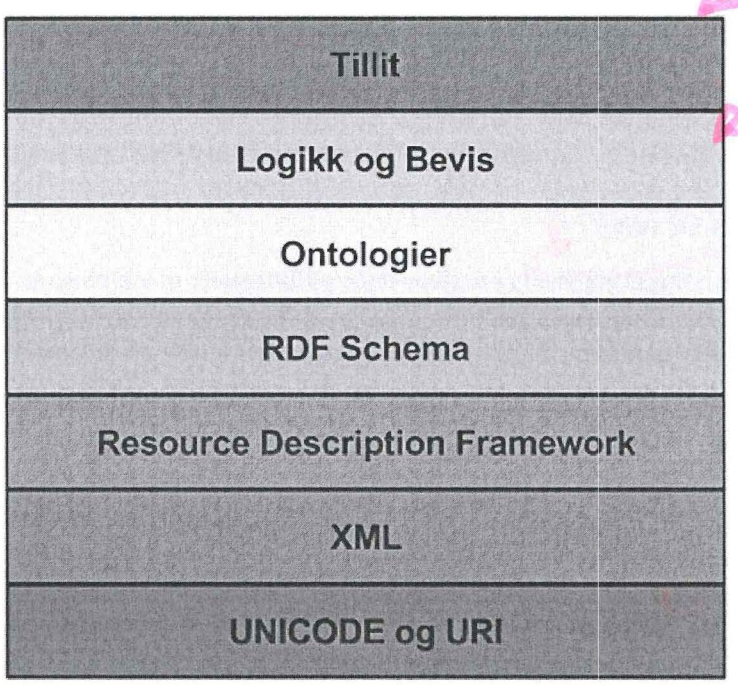
Semantisk Web (the Semantic Web) er en samarbeidsbevegelse ledet av det internasjonale standardiseringsorganet World Wide Web Consortium (W3C). Prosjektet om å utvide Internett, slik at meningen til informasjonen blir maskinlesbar, kalles the Semantic Web Activity. Semantic Web Activity ble opprettet i 1998, og har i dag navnet Data Activity. Deres visjon[203] er:

> "(The overall vision of the Data Activity is)...that people and organizations should be able to share data as far as possible using their existing tools and working practices but in a way that enables others to derive and add value, and to utilize it in ways that suit them. Achieving that requires a focus not just on the interoperability of data but of communities."

Arbeid utført av Data Activity inkluderer utviklingen og vedlikehold av Resource Description Framework Model, Resource Description Framework Schema, Web Ontology Language og SPARQL Protocol and RDF Query Language [204].

Teknologi

Begrepet Semantisk Web brukes om teknologier, verktøy og standarder brukt som byggeblokker til et system som skal støtte visjonen om et Internett med mening. Semantisk Web er bygget opp av en lagarkitektur med ulike teknologier eller komponenter. Figur 6.2 gir et overblikk av de ulike komponentene i Semantisk Web.



Figur 6.2: Semantisk Web lagarktekturen over Semantisk Web teknologier

UNICODE, standard for tegnrepresentasjon, og URIer, standard for identifisering av ressurser, benyttes som grunnlag for representering av tegn og identifisering av ressurser i de fleste dataspråk og i Semantisk Web.

Extensible Markup Language(XML) er et markeringsspråk som definerer et sett av regler for koding av dokumenter til et format som både kan leses av mennesker og maskiner [205]. XML og relaterte standarder danner en felles måte å strukturere data på Internett.

Resource Description Framework(RDF) er en metadata-representasjonmodell som bruker URIer til å identifisere Web-baserte ressurser og en graf-modell for å beskrive relasjoner mellom ressurser.

Resource Description Framework Schema(RDFS) er et modelleringsspråk som beskriver klasser av ressurser, og ressursenes tilhørende egenskaper i en

RDF-modell.

Ontologier er en måte å representere kunnskap på. En ontologi er en oversikt over begreper innen et område, hvilke egenskaper de har og hvordan de forholder seg til hverandre. Må sjekke om denne brukes andre steder:[206]. Ontologier kan bli modellert som et tre med noder eller et kart som beskriver relasjoner.

Logikk- og beviskomponenten er et (automatisk) resonneringssystem som benyttes ved bruk av ontologier. Logikken brukes til å utvide ontologiske språk for økt funksjonalitet og til å lage applikasjonsspesifikk deklarativ kunnskap. Ved bruk at dette systemet kan programvareagenter finne ut om ressurser tilfredsstiller ulike krav eller ikke. Systemet kan utføre bevis validering.

Tillit-komponenten skal gi en garanti for kvaliteten på informasjonen på nett. I dag fungerer ikke komponenten optimalt siden den ikke kan garantere påliteligheten av all informasjon på nettet.

Ved hjelp av Semantisk Web-teknologier er det mulig å integrere og kombinere data fra ulike kilder, og for maskiner å tolke dataene. Semantisk Web-teknologier gjør det mulig å forstå hvordan data er relatert til objekter i den virkelige verden [207].

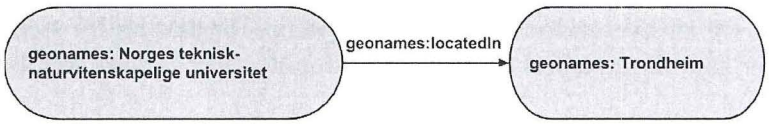
under skal vi se nærmere på følgende 6 komp

6.1.1 Resource Description Framework

RDF er en standardmodell for utveksling av data på nett. RDF definerer en enkel måte å uttrykke forholdet mellom ulike dataelementer som kan være på forskjellige formater[208].

RDF brukes i mange forskjellige sammenhenger for å beskrive informasjon på Internett. RDF utvider Internetts linkstruktur ved å bruke Uniform Resource Identifier(URI) eller navn for å navngi relasjoner mellom objekter. Disse objektene, navnene og forholdene blir omtalt som tripler i RDF.

En trippel er en setning med strukturen: <subjekt> <predikat> <object>. Triplene danner en direkte, rettet graf der kantene representerer den navngitte forbindelsen mellom to ressurser. Ressursene er representert ved noder, se Figur 6.3. Kantene i grafen representerer forholdet to ressurser har til hverandre. Tripler kan inneholde blanke noder, der en blank node representerer objekter uten egne navn eller URIer. Disse nodene kan benyttes for å få at noden eksisterer, men uten å si noe om hva noden representerer. Ved bruk av denne modellen gir RDF muligheten for at strukturert og semistrukturert data kan deles på tvers av ulike applikasjoner [209].



Figur 6.3: Et eksempel på en RDF graf

En ressurs er unikt identifisert ved hjelp av en URI eller en Uniform Resource Locator (URL). En URI er en kompakt streng av tegn og er generaliserte versjoner av URLer som brukes til å finne websider i nettleseren. Med andre ord, alle URLer er URIer, men ikke motsatt. URL er altså en subtype av URI hvor man identifiserer og navngir en ressurs ved hjelp av lokaliseringsinformasjon eller ressursens adresse.

6.1.2 Resource Description Framework Schema

Det finnes flere språk for å definere vokabular for RDF-modeller. Et av disse er RDFS. RDFS gir muligheten til å definere klasser som brukes til å skive ressurser, og til å definere egenskaper ressursene skal ha [210].

I RDFS er klasser og egenskaper ganske likt som i objekt-orienterte programmeringsspråk som Java og C#. En forskjell er at i stedet for å definere en klasse ut i fra egenskaper dens instans har, definerer RDFS egenskaper ut i fra klasser av ressurser de kan falle innenfor. Ved hjelp av denne måten å definere egenskaper på er det lettere å utvide beskrivelsen av en eksisterende ressurs [211].

RDFS gir standarder for å beskrive klassehierarkier, egenskapshierarkier, sette restriksjoner på hvilke ressurser som kan ha en type egenskap og på hva verdien av denne egenskapen kan være, definere ressurser som datatyper, literaler og ressurs og berikelser til definering av lister [212].

6.1.3 Web Ontology Language

Web Ontology Language (OWL) er et kunnskapsrepresentasjonsspråk på lik linje med RDFS [210]. Språket brukes til publisering og deling av data ved hjelp av ontologier. I OWL er det mulig definere terminologi som kan brukes i RDF-dokumenter. Det vil si å kunne beskrive klasser, egenskaper og instanser brukt i web-dokumenter. Både semantikken, datatypene og dataverdiene i OWL er basert på "description logic".

Klasser og roller i OWL skiller ressurser som representerer klasser, eller grupper med fellestrekk, fra ressurser som representerer roller, eller egenskaper.

OWL bruker i tillegg til klasse og rolle også betegnelsen individ. Individer representerer instanser av klasser.

Det finnes to typer egenskaper i OWL: objektegenskaper og datatypeegenskaper. Objektegenskaper spesifiserer parvis forhold mellom ressursene. Datatypeegenskaper spesifiserer forholdet mellom en ressurs og en datatypeverdi.

I tillegg til klasser og egenskaper har OWL støtte for å relatere instanser. Man kan si at to instanser er lik hverandre ved hjelp av egenskapen owl:sameAs. Dette kan være nytttig da entiteter kan bruke forskjellige identifikatorer selv om de refererer til samme ting. For et eksempel på OWL syntaks se Figur 6.4.

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:dc="http://purl.org/dc/elements/1.1/">

    <!-- OWL Header Example -->
    <owl:Ontology rdf:about="http://www.linkeddatatools.com/plants">
        <dc:title>The LinkedDataTools.com Example Plant Ontology</dc:title>
        <dc:description>An example ontology written for the LinkedDataTools.com RDFS &
        OWL introduction tutorial</dc:description>
    </owl:Ontology>

    <!-- OWL Class Definition Example -->
    <owl:Class rdf:about="http://www.linkeddatatools.com/plants#planttype">
        <rdfs:label>The plant type</rdfs:label>
        <rdfs:comment>The class of plant types.</rdfs:comment>
    </owl:Class>

</rdf:RDF>
```

Figur 6.4: Et eksempel på OWL syntax[213].

OWL gjør det mulig å gjenbruke ontologier ved hjelp av owl:import. Når en ontologi importerer en annen er semantikken i den importerte ontologien sann i ontologien som importerer.

6.1.4 SPARQL Protocol and RDF Query Language

SPARQL Protocol and RDF Query Language (SPARQL) er et RDF-spørrespråk for uthenting og endring av data. Det finnes også andre RDF-spørrespråk, blant annet SeRQL [214] og Versa [215]. Spørringene i SPARQL kan inneholde tripler, konjunksjoner, disjunksjoner eller andre mønstre. De fleste spørringene er på trippelformat [216].

Resultatene av en spørring i SPARQL formuleres som en usortert liste av løsninger, som gjenspeiler hvordan spørregrafen matcher RDF-dataene. Det kan være null, en eller flere løsninger til en spørring. Spørregrafen er en spørring på trippelformat, hvor nodene i trippelen danner en graf. Ved en spørring i SPARQL vil spørregrafen matches med RDF-dataene det spørres mot. I en slik spørring må alle de frie variablene brukt i spørringen bli bundet til resultatet. En fri variabel vil si en variabel som ikke er blitt definert i

spørringer. I SPARQL blir slike variabler definert eller bundet ved hjelp av substitusjon. Spørreresultatene kan inneholde blanke noder. Se Figur 6.5 for et eksempel på data, en spørring og et resultat av en SPARQL-query.

Data:

```
@prefix foaf:  <http://xmlns.com/foaf/0.1/> .

_:a  foaf:name   "Johnny Lee Outlaw" .
_:a  foaf:mbox   <mailto:jlow@example.com> .
_:b  foaf:name   "Peter Goodguy" .
_:b  foaf:mbox   <mailto:peter@example.org> .
_:c  foaf:mbox   <mailto:carol@example.org> .
```

Query:

```
PREFIX foaf:   <http://xmlns.com/foaf/0.1/>
SELECT ?name ?mbox
WHERE
  { ?x foaf:name ?name .
    ?x foaf:mbox ?mbox }
```

Query Result:

name	mbox
"Johnny Lee Outlaw"	<mailto:jlow@example.com>
"Peter Goodguy"	<mailto:peter@example.org>

Figur 6.5: Et eksempel på SPARQL-syntax[216].

Resultat av spørringer gjort i SPARQL kan representeres på forskjellige maskinlesbare formater, blant annet XML, JavaScript Object Notation (JSON), Comma Separated Values (CSV) og Tab Separated Values (TSV) [217].

6.1.5 Description logic

Description logic er en familie med logikkbaserte språk for kunnskapsrepresentasjon som kan brukes til å representere terminologi for et domene på en strukturert måte [218]. Begrepene brukes til å spesifisere egenskaper ved objekter og individer som hører til et domenet.

Begreper innen description logic beskrives med logiske operatører: Konjunksjoner, disjunksjoner, unioner av sett, negasjon, samt eksistenskvantorer($\exists$) og allkvantorer($\forall$). Begrepene kan brukes til å lage regler og sannheter i en description logic kunnskapsbase. Begreper kan deles inn i to deler: En terminologidel og en påstandsdel.

Terminologidelen består av beskrivelser av begreper i domenet. Beskrivelsene er begreper, roller og relasjoner mellom disse i form av regler. Påstandsdelen består av beskrivelser av konkrete situasjoner. Situasjoner beskrives ved å gi individer egenskaper i form av sannheter.

Description logic gjør det mulig å resonnere. Det vil si trekke ut implisitt kunnskap fra den eksplisitte representerte kunnskapen. En typisk resonneringsoppgave kan være å finne ut om en beskrivelse ikke er motstridende, eller om en beskrivelse er mer generell enn en annen.

Prolog

Prolog er et deklarativt, logisk programmeringsspråk. I et deklarativt programmeringsspråk er alle programsetningene definisjoner, representert i Prolog som sannheter og regler. Prolog kalles et logisk programmeringspråk fordi det er satt sammen av logiske slutninger. De logiske slutningene er uttrykt i form av relasjoner. Ved kjøring av Prolog-kode utføres det en spørring på relasjonene og deres verdier [219].

En sannhet er en Prolog-setning som inneholder en identifikator etterfulgt av en n-tuppel av konstanter. Identifikatoren blir tolket som navnet på en matematisk relasjon, mens sannheten spesifiserer at n-tuppelen er i en relasjon med identifikatoren. Et eksempel på en sannhet kan være: edge(a,b). Et annet navn på en relasjonsidentifikator som ofte er brukt i sammenheng med Prolog er predikat. I eksempelet over er prediktatet edge. Når en tuppel av verdier inngår i en relasjon sier vi at tuppelen tilfredsstiller predikatet.

En regel er en setning som gir vilkår for hva som skal til for at en tuppel tilfredstiller et predikat. Et eksempel på en regel finnes i Figur 6.6 som også inneholder en forklaring på hvordan regelen skal tolkes.

```
tedge(Node1,Node2) :-
      edge(Node1,SomeNode),
      edge(SomeNode,Node2).
```

The pair `(Node1,Node2)` satisfies the predicate `tedge` if there is a node `SomeNode` such that the pairs `(Node1,SomeNode)` and `(SomeNode,Node2)` both satisfy the predicate `edge`.

Figur 6.6: Et eksempel på en regel i Prolog med forklaring[219].

6.1.6 ClioPatria

ClioPatria er en SWI-Prolog-applikasjon som gir en "ready-to-run" web-server [220]. ClioPatria gir mulighet til å lese og skrive RDF. Lesing og skriving gjøres ved hjelp av spørringer og lagring av sannheter på formen rdf(Subject, Predicate, Object) i Prolog [221].

ClioPatria gir tilgang til en SPARQL-server som prosesserer SPARQL-spørringer over HTTP. ClioPatra støtter i tillegg SeRQL og Sesame HTTP-protokoll. Resonnementsbibliotek, også kalt entailment modules, er del av

tilleggsfunksjonaliteten til ClioPatra. Denne gjør at ClioPatria-systemer kan forholde seg til såkalte "entailment reasoners". Dette vil si at man kan legge til støtte for logikk for entailment, som er relasjonen mellom to utsagn der hvis utsagn A er sann så må også utsagn B være sann.

ClioPatria gir mulighet til å lage brukere og benytte OpenID-tjenester for å benytte ekstern autorisering for økt sikkerhet. I tillegg tilbyr ClioPatria et web-basert grensesnitt som gir utviklere muligheten til å browse RDF, laste og losse grafer, teste spørringer interaktivt, tilgang til dokumentasjon om HTTP tjenester og tilgang til kildekoden til ClioPatria.

Fordelene med å bruke ClioPatria er at man kan bruke enkle rdf(Subject, Predicate, Object)-spørringer i Prolog for å realisere alle de grunnleggende graf-mønstergjenkjenningene som finnes i dataene. En kan i tillegg gi et navn til en spørring og bygge opp komplekse spørringer ut i fra mindre komplekse spørringer. Dette forenkler vedlikehold av komplekse spørringer[222].

Istedet for å bruke forhåndsdefinerte SPARQL og SeRQL-funksjoner og betingelser kan Prolog-predikater brukes om tilstander eller funksjoner hvor som helst i et søk. Andre relasjoner i predikatene enn det som er mulig med SPARQL eller SeRQL kan innføres. Siden ClioPatria og RDF-tjenesten er tett koblet til Prolog gir dette mulighet for resonnering og utforskning av RDF-grafen.

6.2 Kunnskaps- og informasjonskilder

intro

6.2.1 The Drug Ontology

The Drug Ontology(DrOn) er en ontologi for legemidler tilgjengelig i USA. DrOn ble laget fordi det var behov for korrekt og konsistent legemiddelinformasjon på et format som kan benyttes i applikasjoner med Semantisk Web, slik at det er mulig å gjøre spørringer på National Drug Codes(NDC)[1]. Ved bruk av DrOn kan det gjøres spørringer på virkestoff, effekt av legemiddel og den terapeutiske klassen legemiddelet tilhører. Ontologien inneholder historiske data NDCer.

DrOn er laget ved hjelp av gjenbruk av RxNorm, Universal Resource Identifiers fra Chemical Entities of Biological Interest (ChEBI) og Protein Ontology (PRO). Prosessen bak DrOn er delt inn i 3 deler. Trekke ut relevant data fra RxNorm[2], transformering av data og lagring av data i et normalisert

[1] National Drug Codes er unike produkt identifikasjoner bestående av 10 siffer fordelt på 3 segmenter som brukes i USA for medesiner brukt av pasienter [198].

[2] RxNorm er et normalisert navnesystem for generiske og merkede medikamenter. Det er et verktøy for å støtte semantisk interoperabilitet mellom medisinske terminologier og

Relational Database Management System, og omgjøring av data til OWL ved hjelp av Universal Resource Identifiers fra ChEBI. Historiske data av RxNorm er brukt til å koble sammen entiteter til klasser i DrOn [199].

I 2013 inneholdt DrOn over 500 000 klasser. Blant disse finner vi klasser fra Minimum information to reference an external ontology term (MIREOT), OWL, ChEBI, PRO og RxNorm. DrOn er i dag på listen til The Open Biological and Biomedical Ontologies (OBO) over godkjente ontologier. Listen er basert på prinsipper fra OBO for ontologiutvikling med mål om å skape referanseontologier i biomedisinske domene.

For å teste DrOn er det laget en applikasjon hvor man kan søke etter medikamentingredienser og få ut en liste over alle madikamenter som inneholder det angitte stoffet. Ved bruk av denne applikasjonen kan man finne over 20 000 legemidler. Se http://ingarden.uams.edu/ingredients.

6.2.2 FEST

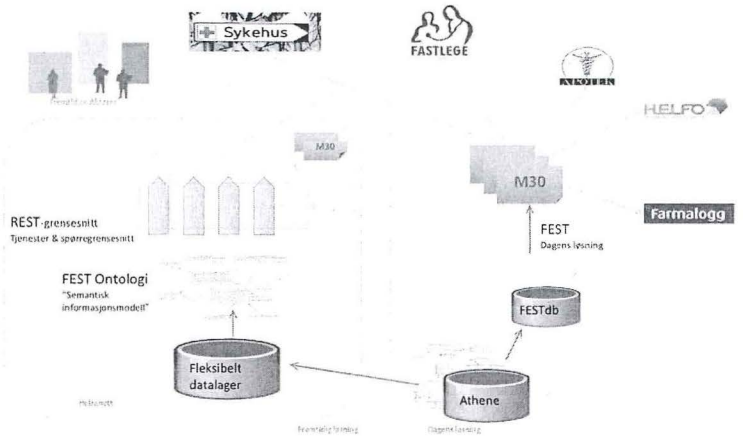
FEST er en database som inneholder legemiddelinformasjon fra flere kilder. Det er hovedsaklig Legemiddelverket som behandler og produserer innholdet, men FEST har andre samarbeidspartnere som tilbyr informasjon om legemidler [113].

Grunndataene i FEST er nedlastbare og kan benyttes i ulike systemer. Slik FEST fungerer i dag trenger systemer som ønsker å bruke FEST bare å tilpasses slik at de kan motta data på riktig format. FEST består av ulike kataloger med data. Dette gjør det mulig å laste ned kun de delene som er relevant. Data blir strukturert i FEST-databasen før den sendes ut i form av meldinger med helseinformasjon.

Dagens løsningen støtter i dag ikke Tim Berners-Lee sin 5-stjerners skala [208] for publisering av data, noe som er et av målene FEST har for fremtiden. Det er blitt foreslått en restrukturering av FEST. I den nye løsningen er det foreslått å ta i bruk ontologier. I tillegg er det blitt foreslått et tjeneste- og spørre-grensesnitt som skal erstatte dagens meldingsformat for deling av data. Grensesnittet skal fungere som et mellomlag mellom FEST-ontologien og leverandørsystemer som ønsker å bruke FEST-data. Planen er å benytte REST-grensesnitt. Et oversiktsbilde av dagens løsning og et forslag til en fremtids løsning (*****-FEST) vises i Figur 6.7.

---

farmakologisk kunnskap [223].



*Figur 6.7: Høyre del av figuren viser en oversikt over dagens løsning av FEST. Venstre del av figuren viser en oversikt over fremtidens løsning av *****-Fest*

6.2.3 SAFEST

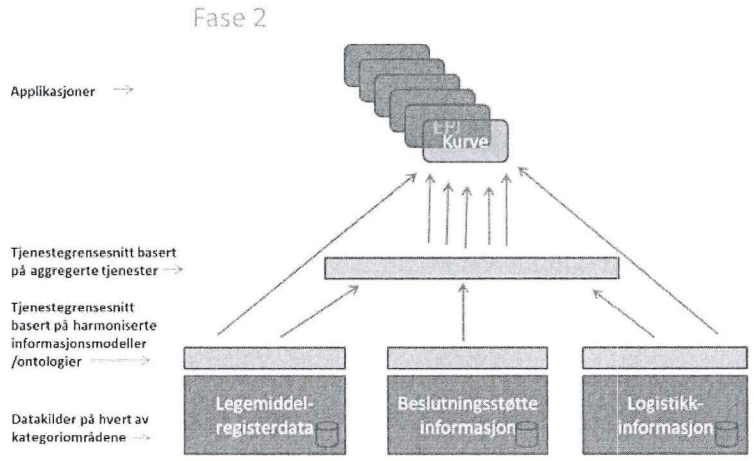
I SAFEST-prosjektet skal det skal vurderes om FEST kan forbedres, og tilpasses spesialisthelsetjenestens behov innenfor legemiddelforsyning og legemiddelhåndtering. Målet er at spesialisthelsetjenesten skal bruke nasjonale legemiddelregister for å bidra til økt pasientsikkerhet [118].

Tjenestegrensesnittet i SAFEST skal samle og presentere informasjon om legemiddelhåndtering på en fleksibel og kontrollert måte. Tjenesten er ikke implementert enda. Det som presenteres her er et målbilde for tekniske løsninger tjenesten kan komme til å ha. Detaljer rundt informasjonskilder, informasjonsmodeller og innholdet i tjenesten er fortsatt under arbeid.

Det er ønskelig å samle data på et sted, med god kvalitet og på en måte som gjør at de alltid er tilgjengelig for brukeren. SAFEST har derfor bestemt at det bør etableres et felles tjenestegrensesnitt som støtter oppkobling mot tre informasjonskategorier. Disse tre informasjonskategoriene består av legemiddelregisterdata, beslutningsstøtte og logistikkinformasjon.

For at en slik løsning skal fungere optimalt kreves det også at det etableres informasjonsmodeller innen hver kategori. Informasjonsmodellene bør samordnes på tvers av kategoriene slik at modellene baseres på samsvarende ontologi, og tilrettelegger for konsistent kobling til informasjonen tilgjengelig.

Det er ulike planer for hvordan data i informasjonskategoriene skal importeres og skapes. Legemiddelregisterdata skal importeres fra Legemiddelverkets system, beslutningsstøtteinformasjon skal skapes og tilgjengeliggjøres av fagmiljøene og logistikkinformasjon skal vise til oppslag fra sykehusapotekenes systemer. For en oversikt over systemet se Figur 6.8.



Figur 6.8: Målbilde for STAFEST løsningsarkitektur[118].

6.2.4 Medication & Pharmacological Resource Portal 0.1 (Pille)

Medication & Pharmacological Resource Portal, også kalt Pille, er en ressurstjeneste laget av NTNU. Pille er et samlested hvor store datakilder om medikamenter og farmakologi lagres. Det er et system hvor utviklere kan ta i bruk dataene uten å bry seg om datauthenting og datamanipulering. Sentrale teknologier brukt i Pille er ClioPatria, Prolog og SPARQL [224].

Pille inneholder data fra blant annet FEST, DrOn, Legemiddelhåndboka, WHO og ATC-registeret. Det ligger også støtte i applikasjonen for at flere kilder kan integreres i systemet uten mye tilpasning. Innholdet i Pille er organisert ved hjelp Semantisk Web. Dette gjør det lettere å bruke og å gjøre spørringer mot informasjonen [225].

6.3 Egenjournalsystemer

6.3.1 Indivo

Indivo er en helsejournalplattform laget av Children's Hospital Informatics Program(CHIP). Den fokuserer på å gi brukere kontroll over sin egen helsejournal. Det er en plattform som muliggjør utvikling av personlige helseapplikasjoner hvor pasienter kan se og legge inn merknader til informasjonen [226].

Prosjektet startet i 1998 og har etter dette blitt brukt i Microsoft sitt nært beslektede prosjekt HealthVault. Det har også vært med som inspirasjon til

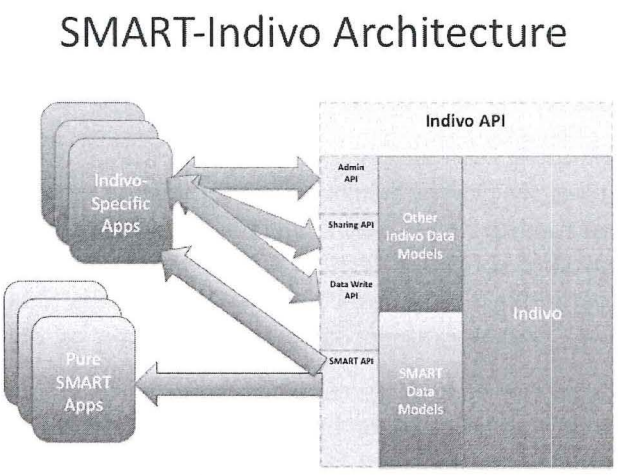
Google Health sin arkitektur, samt at det er brukt i Dossias helsejournal [227].

Indivo er en open-source plattform, bygget slik at brukere lett kan utvide og tilpasse tredjepartsapplikasjoner for å forbedre styring og analyse av sine helseopplysninger. For å øke utviklingen av disse tredjepartsapplikasjonene tilbyr Indivo en rekke funksjonaliteter. Disse er sikker lagring, kategorisering, aggregering av helsedata, single sign-on og standrardbasert dataaksessdelegasjon, et web-basert API som er enkelt og åpent, og samlet brukervasling [228].

Indivo X, som per dags dato er den nyeste utgaven av Indivo, benytter SMART-plattformen som skal gjøre det lettere å lage helseapplikasjoner som kan kobles til helsejournalen [229]. SMART-platformen finnes i to versjoner: SMART Classic og SMART on FHIR. SMART Classic bruker et REST API basert på RDF, mens SMART on FHIR benytter den nye helsestandarden kalt Fast Healthcare Interoperability Resource(FHIR) [230].

SMART-plattformen består av tre deler: applikasjoner, beholdere og brukergrensesnitt. Applikasjonene viser i denne sammenhengen til de forskjellige systemene som brukerere av SMART har laget, disse systemene kan være i form av helseapper. Beholderene peker til dataleveradørene i SMART-universet. Disse kan være alt fra personlig styrte helsejournalsystemer til helsedatautveksling. Beholderene skal koble SMART API'et til apper og gi en kontekst til miljøet appen kjøres i. SMART-grensesnittet gjør det mulig for applikasjoner og beholdere å snakke med hverandre ved hjelp av SMART API'et. API'et holder applikasjonene og beholderene adskilt, slik at brukerne kan kombinere applikasjoner og beholdere etter eget ønske.

Indivo ønsker å ta bruk av fordelene ved å lett kunne knytte andre applikasjoner til sitt system ved SMART arkitekturen, og samtidig kunne implementere ny funksjonalitet som er nødvendig for Indivo systemet. Resultatet blir da en blandingsarkitektur mellom SMART og Indivo kalt SMART-Indivo. Figur 6.9 gir et oversiktsbilde over arkitekturen.



Figur 6.9: Et oversiktsbilde av SMART-Indivo arkitekturen [229]

I tillegg til SMART-plattformen benytter Indivo seg av eksisterende teknologier som OAuth, OpenID og REST. OAuth brukes i Indivo til autentisering og autorisasjon av brukere og datamaskiner [231]. Mange av funksjonalitetene i OAuth er modifisert til å passe inn i Indivo arkitekturen. REST brukes når helseapplikasjoner skal sende og opprette HTTP-samtaler [232].

6.3.2 Dossia

Dossia er en personlig helsejournaltjeneste. Tjenesten gjør det mulig for brukere å samle kopier av sine egne helsedata fra ulike steder, slik at brukeren selv kan lage og nyttiggjøre sin egen personlige og bærbare elektroniske helsejournal [233].

I 2006 ble Dossia grunnlagt som et samarbeidsprosjekt mellom flere arbeidsgivere. Målet var å gi sine ansatte en helsejournal. I 2008 ble den første piloten av Dossia gjennomført [234].

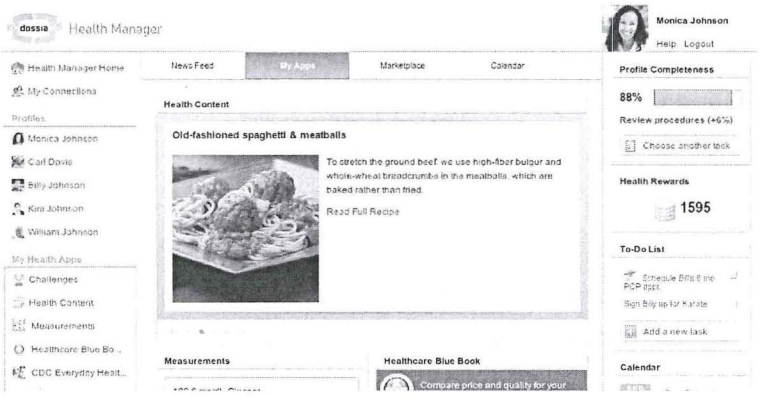
Dossia er basert på open source-plattformen Indivo. I likhet med Indivo bruker Dossia OAuth for autentisering og autorisering, OpenID for deling av identiteter mellom Doissa partnersystem og REST for HTTP-kommunikasjon [235].

Dossia har et egetdesignet API som gjør sikker lesing og oppdatering av helsedata mulig. Dette innebærer at brukeren kan legge inn informasjon om legemidler, allergier og vaksinasjoner, samt lese inn XML-dokumenter og laste

opp bilder. Dossia API'et er et åpent grensesnitt som gjør det mulig for andre å bygge helseapplikasjoner som kan kobles opp til Dossias helsejournal [236].

Dossia benytter seg av Dossia Schema. Dossia Schema er en sammenslåing av ulike skjemastandarder til et sett omfattende og fleksible XML-skjemaer. Disse skjemaene skal sørge for lett og sikker overføring av data mellom helsejournalen og andre leverandører av helsedata. Dossia Schema vil endre seg etterhvert som Dossia-systemet får nye leverandører og datakilder [237].

Dossia User Interface er et brukergrensesnitt som Dossia bruker og lar andre leverandører av helseappliksajoner bruke. Slik kan brukere lett kjenne seg igjen i helseapplikasjoner som kan kobles til deres helsejournal hos Dossia. Hoveddelen til brukergrensesnittet er Dossia Health Manager, se Figur 6.10. Denne inneholder en samling av programmer som kan hjelpe brukeren å oppdatere helsejournalen sin, samt informasjon som skal hjelpe brukeren å bli mer involvert i sin egen helse [238].



Figur 6.10: Brukergrensesnittet Dossia Health Manager [239]

6.3.3 Microsoft HealthVault

Microsoft HealthVault er en skybasert plattform som skal lagre og vedlikeholde helse- og fitnessinformasjon [240]. Systemet fungerer som en egenopprettet journal hvor brukerne styrer innholdet og hvem som har tilgang til det. Det er et sted der personer kan samle, lagre, bruke og dele helseopplysninger på nett [241].

Opplysningene kan legges inn manuelt av brukerne. Dette kan være opplysninger om legemidler, vaksinasjoner, vekt osv. Systemet kan også importere opplysninger fra regneark, digitale journaler og scannede dokumenter. Det er lagt inn støtte for at en rekke apper og utstyr skal kunne dele informasjon med HealthVault-journalen [242].

Microsoft HealthVault legger vekt på at legen skal kunne hjelpe pasienten med å forbedre sin journal. Dette kan gjøres på ulike måter. Hvis legen bruker utstyr som kan kobles opp mot HealthVault, kan utstyret kobles mot pasientens journal og informasjonen lagres der. Pasienten kan be legen om tilgang til helseopplysningene sine i et format som gjør det enkelt for pasienten å overføre opplysningene til journalen.

REST er en arkitekturstil som brukes i HealthVault. REST bruker HTTP forespørsler til å håndtere dataoperasjoner over nett. I tillegg benytter HealthVault seg av SOA, som gjør at systemet er tilgjengelig gjennom webtjenester [243]. SOA-tjenesten har bare tilgang til én journal av gangen, noe som er med på å øke sikkerheten til HealthVault-plattformen [244].

Webtjenestene som kan kobles opp mot systemet kan implementeres ved hjelp av Simple Object Access Protocol(SOAP). De er innkapslet i software development kits som tilbyr en enkel måte å integrere webtjenesten med helsejournalsystemet. Dette gjøres ved å bruke utviklermiljøet Microsoft.NET [245].

Microsoft har i forbindelse med dette laget et HealthVault.NET Software Development Kit (SDK) som de har planer om å gjøre tilgjengelig for andre utviklere. Dette er for at andre utviklere lettere skal kunne lage applikasjoner som kan koble seg til HealthVault, samt å gi utviklere muligheten til å ta i bruk HealthVault.NET SDK funksjonaliteter [246].

Ved hjelp av HealthVault sitt XML-interface får Microsoft HealtVault en generisk datamodell[3] for helserelaterte data. Datamodellen inneholder også metoder for å aksessere data. All helsedata i HealthVault er beskrevet i XML[248].

HealthVault benytter seg av Drop-Off and Pick-Up (DOPU) og Patient Connect for å sende helsedata til HealthVault uten å ha direkte kontakt med HealthVault-journalen den sender til. Teknologiene fungerer bra for applikasjoner som ikke har et brukergrensesnitt. DOPU brukes når systemet ønsker en en-gang overføring av data til brukere [249], mens Patient Connect benyttes hvis en videre utveksling av data er ønskelig [250].

I HealthVault brukes en tilkoblingsmåte for å gi datatilgang kalt Direct. Direct er utviklet av The Direct Project og er en tilkobling som gir en sikker måte å sende meldinger på. The Direct Project spesifierer en standardisert måte for systemer å sende autentisert og kryptert helseinformasjon til troverdige mottakere over internett. Direct brukes kun når kontakten mellom kildene er godkjent av brukeren [251].

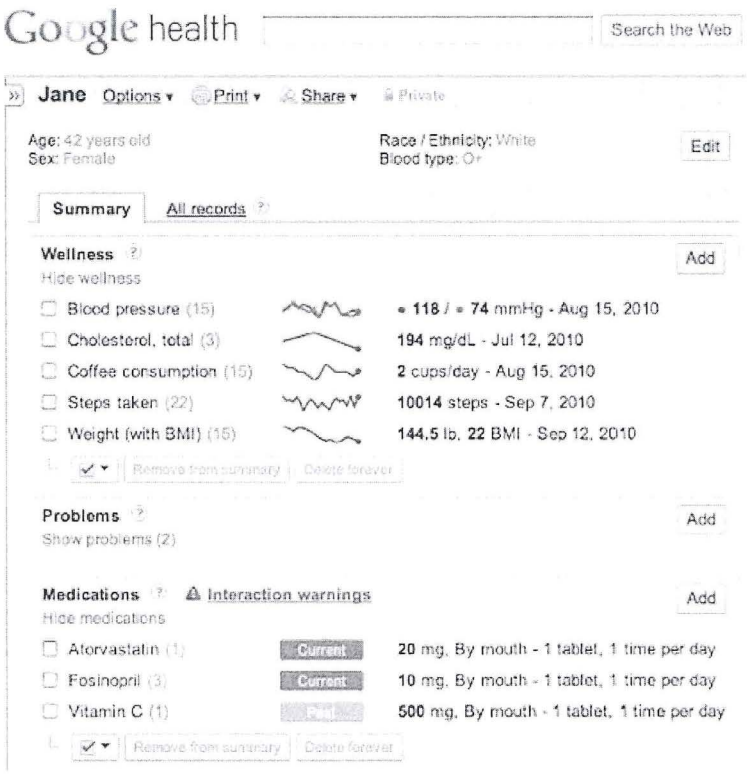
[3]En generisk data model er en model som definerer generelle standariserte relasjonstyper. Det vil si at den antar så lite som mulig om dataene og lar heller dataene bli beskrevet gjennom bruk av såkalte "referanse data". Domene kunnskap er fanget i standard instanser i stedet for i entitet typene [247].

HealthVault bygget på Microsoft Windows Portable Device arkitektur for å gi støtte for tilkobling av apper og helseutstyr. Dette er for å gi apper og utstyr mulighet til å dele data med HeathVault [252]. Microsoft Windows Portable Device gir HealthVault infrastrukturen til å standardisere dataoverføringer mellom journalen og en applikasjon. På toppen av Windows Portable Device teknologien finner vi HealthVault Connection Center som er verktøyet som gjør det mulig å legge inn data i HealthVault [253, 254]. For en oversikt av HealthVault og tilhørende tjenester se Figur 6.11.

Figur 6.11: En oversikt av HealthVault og tilhørendetjenester [255]

6.3.4 Google Health

Google Health var en personlig helsejournal laget av Google. Systemet fungerte ved at brukere la inn helseinformasjon i systemet manuelt eller ved dataoverføring fra samarbeidshelsetjenester. Når informasjonen var lagt inn i systemet kunne Google Health gi brukeren en sammenslått journal av all informasjonen, vise informasjon om helsen til brukeren og vise mulige interaksjoner mellom legemidler og allergier [256]. Figur 6.12 viser hvordan Google Health så ut.



Figur 6.12: Google Health slik det så ut før det ble lagt ned [257]

I 2011 annonserte Google at Google Health skulle legges ned. Google mente at årsaken til at Google Health ble lagt ned var fordi systemet ikke fikk den virkningen Google hadde håpet på [258]. Det er derimot flere kilder som nevner andre grunner til at Google Health ble lagt ned. En av årsakene som nevnes er at Google var for tidlig ute med helsejournalen, noe som førte til at flere eiere av datakilder ikke ønsket å dele informasjon med dem. Dette til tross for at Microsoft HealthVault, Dossia og Indivo utviklet sine helsejournaler tidligere eller rundt samme tid. I tillegg spekuleres det rundt dårlig sikkerhet, liten involvering av leger og dårlig finansiering av projektet [259–261].

Google Health brukte SOA, som tilbyr funksjonalitet som gjør det lettere for utstyr og applikasjoner å dele informasjon med Google Health. Google Health var implementert ved hjelp av Atom Publishing Protocol, REST, HTTP, XML og RSS [243, 245]. De fleste av disse standardene brukes til kommunikasjon innad i systemet. RSS og Atom brukes til fordeling og publisering av data [262].

Really Simple Syndication (RSS) er en standard som gjør det mulig for en nettleser å hente materiale fra internett fortløpende og automatisk. RSS

bruker en type XML basert på RDF for å videreformidle deler av innhold fra nettsider som oppdateres jevnlig. Uten RSS er brukere nødt til å sjekke nettsider daglig for nye oppdateringer [263]. Atom Publishing Protocol er en enkel HTTP-basert protokoll for publisering og redigering av ressurser ved bruk av XML. Atom er utviklet som et alternativ til RSS, men Google Health støttet og kunne benytte seg av begge standardene [264, 265].

Google Health benyttet et egetlaget API. API'et har ikke vært tilgjengelig siden Google Heallth ble nedlagt, og det eksisterer veldig lite informasjon om det.

Google Health benytter også Google Data API og Accounts API. Google Data API tilbyr systemet en enkel standardprotokoll for lesing og skriving av data på nett. Account API tilbyr to modeller for autentisering og autorisering for web-baserte applikasjoner, og autentisering for installerte klientapplikasjoner. Account API benytter OAuth.

6.3.5 Min Helse

Min helse er Norges helsejournal. Min Helse gir en rekke selvbetjeningstjenester: Mine egenandeler, Mine resepter, Mine vaksiner, og kjernejournal. For å få tilgang til Min helse sine tjenester må brukere logge inn med BankID. I tillegg til de større tjenestene nevnt over kan pasienten ved hjelp av Min helse bytte fastlege selv, melde bivirkninger til legemiddelverket og bestille Europeisk helsetrygdkort. Innholdet i Min helse leveres av ulike organisasjoner i helsesektoren. Helsedirektoratet er ansvarlig redaktør.

Tjenesten "Mine egenandeler" gir pasienten en oversikt over egenandelene sine og utbetalte refusjonsbeløp. De egenandelene Min Helse gir oversikt over gjelder kun for frikort for helsetjenester egenandelstak 1. Dette gjelder egenandeler hos lege, psykolog, røntgeninstitutt, i poliklinikk, røntgeninstitutt, pasientreiser og for medisiner på blå resept [266].

"Mine Resepter" gir pasienten oversikt over alle e-resepter rekvirert av lege. Her kan pasienten se om resepten er klargjort på apoteket, få en oversikt over gyldige e-respter og gjenstående utleveringer på resepter. Pasienten kan i tillegg se en oversikt over barnas resepter og oversikt over andres resepter som pasienten har fullmakt til å se. Oversikten over resepter kan også gis på papir [267].

Ved bruk av tjenesten "Mine vaksiner" kan pasienten skrive ut vaksinasjonskort og sjekke vaksinasjonsstatus. "Mine vaksiner" gir pasienten en oversikt over egne vaksiner og andres vaksiner som pasienten har fullmakt til å se. For at pasienten skal kunne se sine vaksinasjoner kreves det at helsestasjon, lege eller vaksinasjonssenteret har registrert vaksinasjonene i det nasjonale

vaksinasjonsregisteret SYSVAK [268].

Kjernekjournalen på Min Helse inneholder informasjon hentet fra offentlige registre og fastlege, samt egenregistrert informasjon. Tjenesten kjernejournal er i dag kun tilgjengelig som utprøvning i Trondheims- og Stavangerområdet, men i mars 2015 vil flere områder inkulderes i utprøvingen. Kjernejornalen gir informasjon om legemidler, besøkshistorikk fra sykehus, kritisk informasjon og logg over oppslag i kjernejournalen til pasienten. Egenregistrert informasjon kan være informasjon om nærmeste pårørende, spesielle kommunikasjonsbehov og egen sykehistorie. I egenjournalen vises historikk på inntil 3 år av brukte legemidler. Kritisk informasjon er informasjon som kan være avgjørende i en akuttsituasjon. Det kan for eksempel være alvolige allerigier eller overfølsomhetsreaksjoner [269].

Kapittel 7

Oppsummering

Legemiddelhåndteringsprosessen går hovedsakelig ut på å kommunisere og dokumentere handlinger, og å forflytte legemidler. Det er fare for at det oppstår legemiddelskader på grunn av dårlig informasjonsflyt i legemiddelhåndteringsprosessen. Legevakten og hjemmetjenesten mener at tilgangen på nødvendige legemiddelopplysninger og informasjon om legemiddelbruk hemmer arbeidet deres. Den manglende informasjonen om pasientens legemiddelbruk fører til feil behandlinger og uønskede legemiddelreaksjoner.

I dag har ulike institusjoner hver sine lister over hvilke legemidler pasientene står på. Det er påvist at disse listene ofte ikke stemmer overens. Kommunikasjonen mellom ulike institusjoner har vist seg å være dårlig og dermed utfordres pasientsikkerheten.

Det skjer mange feil fordi legemiddellistene til ulike institusjoner ikke er samstemt. Pasientsikkerhetsprogrammet "I trygge hender" har et mål om å løse samstemmingsproblematikken. Deres forslag til løsning er at pasientene skal bære med seg en papirutgave av legemiddellisten.

Legemiddelhåndteringsprosessen involverer mange aktører. Lover og retningslinjer regulerer ansvarsfordelingen i prosessen. Det er allikevel ofte uklart hvilke aktører som har ansvar for hva, og hvor langt de ulike aktørenes ansvar strekker seg. Det kan være vanskelig for hver enkelt aktør å vite om de tar ansvar for det de skal, og når de beveger seg mot andres ansvarsområder.

Det er omfattende for leger og annen helsepersonell å sette seg inn i hver enkelt pasient sin situasjon, spesielt når det er flere behandlere involvert i et sykdomsforløp. For en lege med ansvar for å vurdere om legemidlene til en pasient passer sammen, kan det være vanskelig å holde oversikt over hvorfor legemidlene er forskrevet og hvilken effekt de er ment å ha. I noen tilfeller har legene tilgang til systemer som tydeliggjør uheldige interaksjoner mellom legemidler.

Pasienter utenfor helseinstitusjoner og pasienter uten hjemmesykepleie har selv ansvar for at legemidlene tas som legen har bestemt. Pasienter og pårørende mangler informasjon om hvorfor legemidler er rekvirert, hvilken effekt de er ment å ha, samt mulige bivirkninger. Den eneste tilgjengelige informasjonen er pakningsvedlegg og informasjon tilgjengelig på nett. Også hjemmesykepleien savner mer informasjon om pasientenes legemidler.

Pasienten oppfordres til alltid å lese pakningsvedlegget. Pakningsvedlegget inneholder svært mye informasjon og kan være vanskelig å få oversikt over. Eldre pasienter har vanskeligheter med å oppfatte informasjon og brukerveiledninger. Mer tilgjengelig informasjon om legemidler, som er lettere for eldre å forstå, kan bidra til økt etterlevelse blant pasienter som tar flere legemidler samtidig.

65% av befolkningen på over 65 år har to eller flere kroniske lidelser. Antallet diagnoser er økende med alder. Mange eldre bruker derfor flere legemidler samtidig. For en bruker som tar flere forskjellige legemidler kan det være problematisk å forholde seg til mange pakningsvedlegg. Informasjon om sammensetning av legemidler er vanskelig å finne. Fastlegen blir derfor en naturlig informasjonskanal om virkning, tenkt effekt, mulige bivirkninger og interaksjoner mellom legemidler.

Eksisterende tjenester for å finne informasjon om interaksjoner og effekt av legemidler er i dag rettet mot klinikere. Pasienter kan ikke forventes å ha samme kunnskapsnivå og forståelse for domenespesifikke begreper. For å gjøre informasjon om effekter og interaksjoner mellom legemidler tilgjengelig for pasienter, bør den formidles på en måte som ikke stiller krav til opplæring eller medisinskfaglig kunnskap.

Informasjon tilgjengelig for pasienter om interaksjoner fokuserer på farlige interaksjoner. Det finnes interaksjoner mellom legemidler som ikke er farlige, men som likevel er plagsomme for pasienten. Det kan derfor være interessant både for pasienten og klinikere å se hvilken kombinasjon av legemidler som fører til plagene. Slik kan informerte avgjørelser tas om hvorvidt effekten av legemidlene er verdt plagene eller ei.

Teknologier med hensikt å fremme helse er mye omtalt i politikken. Det legges vekt på at teknologi skal legge til rette for å gi pasienten mulighet til medvirkning i egen behandling. Pasienten skal gis mer makt ved å få tilgang til relevant og oppdatert informasjon. Ved å tilby pasienter elektronisk tilgang til helsejournalen sin flyttes makt og medbestemmelsesrett fra det offentlige til den enkelte bruker.

Studier har vist at det er varierende hvor aktiv pasienter ønsker å være i sin egen behandling, men det er enighet om at pasienter i stor grad ønsker å være informert om sin egen helsetilstand. Noen pasienter ønsker allikevel ikke

å være informert om alt. At det finnes et tilbud om tilgang til informasjon betyr ikke at man må benytte seg av tilbudet.

En studie fra WHO viser at en måte å motivere pasienter til å ta legemidlene sine er "self-management"-innovasjoner som for eksempel helseapper. Pasienter som er involvert i sin egen behandling har bedre etterlevelse. Et system som viser informasjon om legemidler kan gjøre pasienten aktiv i sin egen behandling. Det kan også føre til at pasienten tar en mer aktiv rolle i diskusjoner med legen angående legemiddelbruk.

Samvalg omhandler det å involvere pasienten i beslutningsprosessen i sin egen behandling. Individuell plan er et virkemiddel som bidrar til samvalg. Det er et tjenestetilbud for samarbeid mellom pasient og heletjeneste. Tjenestetilbudet gir mulighet for en aktiv involvering i egen helse hvor helsepersonell og pasienter kan dra nytte av hverandres erfaringer og kompetanse. Behandlinger uten deltagelse fra pasienten tar ikke hensyn til det som er viktigst for pasienten. Tjenestetilbud for å involvere pasienten og elektronisk helsejournal kan gjøre pasienten mer bevist på sin egen situasjon.

I dag kan man finne sin egen pasientinformasjon ved hjelp av elektroniske pasientjournalsystemer. I Skandinavia er Norge sist ute med å tilby innbygerene innblikk i sin egen helsejournal. Ved bruk av elektroniske pasientjournaler kan det legges til rette for at pasienter skal få innsyn i egen journal. Det er også mulig å gi pasientene mulighet til å redigere data i egen journal. Tiltross for at journalen inneholder mye informasjon, mangler den informasjon om legemidelinteraksjoner. Hvis et interaksjonssystem kommer inn på markedet vil det være naturlig at den integres med pasientjournalen.

Besluttningsstøttesystemer er systemer ment for å hjelpe helsepersonell å ta beslutninger. Det kan tas mye lærdom fra forskning om besluttningsstøttesystemer når det skal utvikles nye systemer i helseomsorgen. Semantisk Web gir muligheten for en detaljert semantisk representasjon av data. Det er gjort forskning på hvordan Semantisk Web kan brukes i beslutningsstøttesystemer.

Eksisterende besluttningsstøttesystemer og andre elektroniske helsetjenester er med på å forebygge flere typer feil. Mange systemer stiller krav om ulike sjekker ved valg av legemidler og dosering. Slike krav fungerer som ekstra ledd i forebyggingen av feilmedisinering. Dagens løsning med elektronisk behandling av resepter har ført til en betydelig reduksjon i antall forskrivnings- og håndteringsfeil. Redusering av feil i helsetjenesten er med på å øke kvaliteten i helsetilbudet.

Selv om teknologi kan være et verdifult hjelpemiddel kan det også være en kilde til frustrasjon og feil. Digitaliserte tjenester kan erstatte personlig kontakt og føre til at pasienten ikke føler seg trygg eller sett. Selv om dagens digitaliserte tjenester i mange tilfeller klarer å utføre helseomsorgsoppgaver,

mislykkes de noen ganger. En av årsakene til at systemer feiler kan være mangel på regelmessige oppdateringer og vedlikehold.

Bruk av IKT gjør det mulig å effektivisere prosesser, og gir brukerne mer uavhengighet og kontroll. I dag eksisterer det flere systemer ment for å effektiviserer dagliglivets oppgaver. Disse systemene flytter kontrollen og medbestemmelsesretten fra fagfolk til enkelt brukere. Eksempler på slike systemer innebærer nettbank for betaling av regninger, eller bestilling av reiser på nett. Kanskje kan helsesektoren hente erfaringer fra andre bransjer for, og ta del i trenden med å gi brukerne mer kontroll.

En måte man kan gi pasienter mer kontroll over egen helse er ved å gi dem tilgang på bedre informasjon om egne legemidler. For å lage et system for legemiddelinformasjon kan det være nyttig å ha en kunnskapsrepresentasjon for legemiddelinformasjon. Det er vanskelig lage en kunnskapsrepresentasjon for legemidler som har riktig informasjon om virkningen av legemidlene. For å lage en god kunnskapsrepresentasjon for legemidler kan det være nyttig å bygge på eksisterende kunnskapsrepresentasjoner, og ta lærdom av den forskningen som er gjort ved utviklingen av disse kunnskapsrepresentasjonene.

Kapittel 8

Videre arbeid

8.1 Problemstilling

Bruk av legemidler er et viktig tiltak i helsevesenet, men er ofte lite koordinert og ikke underlagt samlet kontroll. Pasienten, pårørende og eventuelt institusjon eller hjemmesykepleie savner informasjon om hva legemidlene skal behandle, kortfattet oversikt over mulige bivirkninger og forventet virkning (jf. Motivasjon).

Problemstillingen for masteroppgaven er: **Hvordan kan nyttig legemiddelinformasjon presenteres for pasienter?**

Informasjon om legemidler er av interesse for både pasienter, pårørende og ansatte i helsetjenesten, men i masteroppgaven er det valgt å fokusere på pasienter. Det skal særlig fokuseres på pasienter med flere kroniske sykdommer som tar flere legemidler.

Problemstillingen fokuserer på at det er "nyttig" informasjon som skal vises for brukerne. For å kunne svare på problemstillingen må følgende spørsmål besvares: **Hvilken informasjon er nyttig for pasienter?**

Det er viktig at informasjonen presenteres på en forståelig og brukervennlig måte. På nåværende tidspunkt er informasjon om legemidler tilgjengelig, men presenteres på en lite forståelig måte i form av blant annet pakningsvedlegg. For å kunne svare på problemstillingen må følgende spørsmål besvares: **Hvordan kan informasjon om legemidler presenteres på en forståelig og brukervennlig måte for pasienter?**

Det finnes i dag mange kilder til informasjon om legemidler, men dataene er dårlig strukturert og er på forskjellige formater. For å kunne svare på problemstillingen må følgende spørsmål besvares: **Hvordan kan eksisterende**

informasjonskilder integreres for å presentere legemiddelinformasjon for pasienter?

8.2 Plan for våren

For å svare på problemstillingen ønsker vi å undersøke muligheten for å lage et system for å fremstille nyttig informasjon om legemidler. Målet med et slikt system er å gjøre informasjon om legemiddelbruk, bivirkninger og interaksjoner ved bruk av flere medikamenter lettere tilgjengelig for pasienter.

Implementering er krevende, tar mye tid, og ikke er direkte nødvendig for å kunne svare på problemstillingen. Vi velger imidlertid å gjøre noe implementasjon da vi mener det er en god måte å vise at det er mulig å vise relevant informasjon til pasienter. Vi ønsker ikke å implementere et ferdig system som kan brukes av alle brukere, men velger å avgrense implementasjonen til å bare omfatte et system som kan vise relevant informasjon til noen utvalgte fiktive personer.

Vi velger å avgrense implementasjonen til å ikke omfatte lagring eller overføring av reelle personlige data. En slik avgrensning kan gjøres ved å enten lage fiktive data, eller ved å gå ut i fra et system der brukerne allerede har alle de nødvenige persondataene. Systemet vårt er ikke en kilde til personlig informasjonen, men et system som kobler den personlige informasjonen med generell legemiddelinformasjon. Systemet gjør informasjonen om egne legemidler mer forståelig for pasienter. Siden vi avgrenser implementasjonen til å ikke omfatte lagring eller overføring av personlige data, kan vi se bort i fra problemstillinger knyttet til informasjonssikkerhet.

Papirprototyping er en effektiv og enkel måte å undersøke nytteverdi og brukervennlighet på. Vi velger å gjennomføre papirprotyping i to faser for å undersøke ulike aspekter. I første omgang ønsker vi å finne ut hvilken informasjon pasienter anser som nyttig. I andre omgang ønsker vi å se på ulike måter å presentere informasjonen på, og undersøke hva som gjør at brukere finner systemet forståelig.

For finne svar på hvilken informasjon som er nyttig for pasienter skal vi:

RQ1.1
- intervjue pasienter for å undersøke hvilken informasjon de opplever som nyttig

.2
- intervjue helsepersonell for å undersøke hvilken informasjon de mener er nyttig

.3
- lese litteratur om legemiddelinformasjon til pasienter

- lage et system med informasjon og sjekke om pasienter opplever informasjonen i systemet som nyttig

For å finne svar på hvordan informasjon om legemidler kan presenteres på en brukervennlig måte for pasienter skal vi:

- gjøre brukerundersøkelser på papirprototyper i forkant av kravspesifikasjon
- vurdere brukervennlighet ved å gjøre brukertester på det implementerte systemet
- lese litteratur om brukervennlighet

For å finne svar på hvordan eksisterende informasjonskilder kan brukes til å presentere legemiddelinformasjon for pasienter skal vi:

- undersøke eksisterende informasjonskilder
- sjekke om informasjonskilder kan sammenstilles
- lese litteratur om det formatet datakildene er på
- prøve å benytte eksisterende informasjonskilder i systemet vårt

8.2.1 Hovedaktiviteter

- Litteraturstudium
- Rapport
  - Dokumentere
- Kartlegge krav
  - Brukerundersøkelse
  - Prototyping
- Valg av datakilder
- Valg av teknologier
- Brukergrensesnitt
- Implementere
  - Kode
  - Dokumentere
- Teste
- Evaluere

Identifiserte risikoer:

- Gruppemedlem blir syk
- Veileder blir syk/utilgjengelig
- Uenighet innad i teamet
- Manglende motivasjon
- Mangel på tekniske ferdigheter
- Arbeid går tapt
- Datakildene kan ikke brukes
- Ikke tilgang på pasienter
- Velge feil teknologi
- Velge feil datakilder
- Sviktende kommunikasjon med kontakter i helsesektoren
- Bruke for mye tid på/ ikke få til integrasjon med eksisterende systemer
- Bruker tid på feil ting (Bruker for mye tid på implementasjon)
- Miste fokus på forskningsmål (Utvikle teknologi som ikke gir kreditt karaktermessig)

Konsekvens → / ↓ Sannsynlighet	Lav	Medium	Høy
Lav	Ikke tilgang på pasienter	Arbeid går tapt	
Medium	Sviktende kommunikasjon med kontakter i helsesektoren	Gruppemedlem blir syk Veileder blir syk/utilgjengelig	Bruker tid på feil ting Bruke for mye tid på/ ikke få til integrasjon med eksisterende systemer
Høy		Datakildene kan ikke brukes Manglende motivasjon	Uenighet innad i teamet Mangel på tekniske ferdigheter Miste fokus på forskningsmål

Figur 8.1: Risikomatrise for arbeidet med masteroppgaven

8.2.2 Milepæler

15. januar
Ferdigstillelse av masterkontrakter. Dette inkluderer både kontrakt mellom gruppen og institutt for datateknologi og informatikk ved NTNU, samt kontrakt med den eksterne veilederen Capgemini.

1. februar
Bestemme struktur av oppgaven, og bestemme alle overskrifter.

10. februar
Ha skrevet ferdig masteroppgavens problemstilling.

15. februar
Være ferdig med innsamling av kasustikker.

20. februar
Gjennomføre første papirprototypetest.

20. februar
Ha avklart hvilken datakilder vi skal benytte oss av.

1. mars
Ha startet med implementasjon.

1. april
Gjennomføre andre papirprototypetest.

15. april
Ferdig med all implementasjon.

20. april
Brukertest av implementert system.

1. mai
Levere førsteutkast av masteroppgave til alle veiledere.

1. juni
Levere ferdig masteroppgave.

8.3 Risikoanalyse

Gjennomføring av risikoanalyse gjøres for å kjenne til risikonivå, kunne iverksette tiltak for å hindre uønskede hendelser og for å kunne håndtere det dersom uønskede hendelser skjer. Under presenteres identifiserte risikoer for prosjektet, og en vurdering alvorlighetsgrad. For de mest alvorlige risikoene foreslås tiltak for å begrense sannsynlighet og konsekvens.

Under presenteres tiltak som kan gjøres for å senke sannsynligheten for, og redusere konsekvensen av, risikoer som er identifisert å ha medium eller høy konsekvens og sannsynlighet i Figur 8.1.

Tiltak for å redusere sannsynlighet og konsekvens av å bruke tid på feil ting:

- tenk igjennom hva problemstillingen er, og sørg for at alt man gjør er viktig for å svare på problemstillingen
- lag plan med milepæler, og pass på å følge den
- ha realistiske mål
- sjekk regelmessig om man jobber med riktige ting, og om det er andre ting som har blitt viktigere
- unngå å planlegge mye implementasjon. Implementasjon tar ofte lenger tid enn planlagt, og gir lite karakterutbytte i forhold til hvor mye tid det tar
- lage enkle use case slik at implementasjon blir enkelt

Tiltak for å redusere sannsynlighet og konsekvens av å bruke for mye tid på/ ikke få til integrasjon med eksisterende systemer:

- forsøke å lage et system som ikke er veldig avhengig av integrasjon med andre systemer
- velg fritekst som datakilde om mulig for å gjøre arbeidet med integrasjon lettere
- sjekk arbeidsmengden integrasjon krever for å vurdere om integrering er nødvendig eller gunstig karaktermessig

Tiltak for å redusere sannsynlighet og konsekvens av å miste fokus på forskningsmål:

- sørg for at alle milepæler man bestemmer seg for er viktige for å nå forskningsmål
- tenk igjennom hva problemstillingen er, og sørg for at alt man gjør er viktig for å svare på problemstillingen
- unngå å bli for ivrig med implementasjoner som ikke bidrar nok til å svare på forskningsmål
- sjekk regelmessig om alt man jobber med er viktig for å nå forskningsmålene, og om det er andre ting man kunne gjort som er viktigere for å nå forskningsmål

Tiltak for å redusere sannsynlighet og konsekvens av uenighet innad i teamet:

- forventningsavklaring i forkant

- samarbeidskontrakt
- god kommunikasjon
  - ukentlige møter med gjennomgang av "sjekkliste" om samarbeid
    * hva har fungert og hva kan bli bedre?
- gi hverandre ros
- være åpen for kritikk
- gjøre ting sammen utenom oppgaven
- si ifra før problemer blir for store

Tiltak for å redusere sannsynlighet og konsekvens av mangel på tekniske ferdigheter:

- søke hjelp
- ta seg nok tid til å lese og lære før man gjør
- senke ambisjonsnivået
- benytte oss av teknologier vi kan fra før

Tiltak for å redusere sannsynlighet og konsekvens av at gruppemedlem eller veileder blir utilgjengelig: Det er umulig å hindre at sykdom eller skade oppstår. Det kan imidlertid vurderes hva som kan gjøres for å senke konesekvensen av at det skjer. Gruppen må sørge for å ikke være for avhengig av enkeltpersoner i teamet ved at alle er oppdatert på hva den enkelte jobber med. Vi må også sørge for å ikke være for avhengig av en enkelt veileder, slik at vi kan fortsette arbeidet uavhengig av veileder.

kap.8:
Viktige pkt., men vel mye pkt.-lister?