Automatiseret overvågning/alarmering af patient

Thomas Axel

Kongens Lyngby 2012
IMM-M.Sc-2012-09

Technical University of Denmark
Informatics and Mathematical Modelling
Building 321, DK-2800 Kongens Lyngby, Denmark
Phone +45 45253351, Fax +45 45882673
reception@imm.dtu.dk
www.imm.dtu.dk

IMM-M.Sc: ISSN 0909-3192

Abstract

The Danish healthcare system is extremely expensive in its current form. Hence it is desirable to optimize the system in order to minimize the total costs. By implementing a tele-medical system with a well designed architecture, one can not only save time but also optimize workflow. Another advantage is the opportunity to collect data through data mining, which can be useful in future research projects, which could lead to better diagnoses and treatment of illnesses.

The architecture of the system is paramount in order to be able to improve the system in the future.

Therefore the objective of this project is the development of a system to assess and diagnose the patients health condition when the patients' health state changes from stable to critical. This is achieved by the use of a network of sensors which are placed on the patient. Data from these sensors are then sent to a mobile gateway, which transmits the information to a server. The server processes the information by a mathematical projection and evaluates whether the patients condition is stable or if there is a need for reaction.

The possibilities are promising when using a flexible structure allowing the diagnoses to be dealt with in parallel. This allows different health care professionals to access valuable information through the system so that they can properly diagnose and treat the patient. Multiple implementation schemes are possible, but flexibility and scalability are important in applying the right designed architecture scheme.

A proper architecture together with a vision from the health sector can realise this project.

Resumé

Det nuværende sundhedssystem er dyrt for samfundet og det er derfor ønskeligt, at der optimeres, hvor det er muligt. Ved indførslen af et telemedicinsk system foreligger der en reel mulighed for besparelser på mange af rutineundersøgelserne.
Ved hjælp af et telemedicinsk system med en veldesignet arkitektur, kan der opnås gode resultater for tidsbesparelser i form af ændrede arbejdsgange. Indsamlet data kan endvidere bruges til forskning og dermed på sigt give mulighed for bedre diagnosticering og behandling.

Arkitekturen for systemet er altafgørende ved muligheden for fremtidige systemforbedringer.

Fokus i dette projekt er, at muliggøre et system til evaluering og diagnosticering af patientens tilstand, når denne ændrer sig fra stabil mod værre.
Dette opnåes ved et netværk af sensorer på patienten, hvorfra data sendes til en mobil gateway, der videretransmitterer informationen til en server. Serveren behandler informationerne vha. matematiske fremskrivninger og kan derved evaluere om patientens tilstand er stabil, eller om der er behov for en reaktion.

Mulighederne er gode, såfremt der benyttes en fleksibel struktur, hvor diagnosticeringerne behandles parrallelt og der er mulighed for, at mange forskellige faggrupper kan tilgå systemet og bidrage med deres specialviden. Ved fra starten at tænke fleksibilitet og skalerbarhed ind i arkitekturdesignet, opnåes den mest langtidsholdbare løsning.

Arkitekturdesignet sammen med en vision fra sundhedssektoren kan realisere dette projekt.

Forord

Dette speciale er lavet ved 'Institut for Informatik og Matematisk Modellering, Danmarks Tekniske Universitet', som kandidatopgave på 30 ECTS point, hvormed der opnåes en IT kandidatgrad indenfor ingeniørfaget.

Projektet består af en rapport, der analyserer forskellige telemedicinske arkitekturer, med udgangspunkt i medicoindustrien. Derudover indeholder projektet et design af et system, der realiserer den valgte arkitektur.

For at opnå fuld forståelse af specialet, forventes det, at læseren har IT teknisk baggrund, og derved har kendskab til forskellige IT arkitekturer og netværks paradigmer.

Jeg vil gerne rette en tak til Christian Damsgaard Jensen fra IMM Danmarks Tekniske Universitet, Lyngby, der har været til hjælp når jeg har haft brug for kritiske øjne, eller bare havde lyst til at tale vidt og bredt om rapporten.
Samtidgt vil jeg gerne takke Rune Saaby Duschek fra Pallas Informatik A/S, Allerød, for at være min kontaktperson hos Pallas, samt har vejledt og fulgt mig gennem projektforløbet.

Kongens Lyngby
2. marts 2012
Thomas Axel

Indhold

Kapitel 1

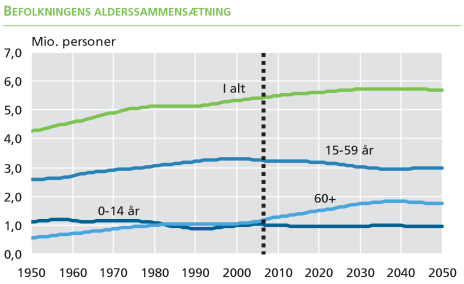
Introduktion

Læger og patienter med kroniske sygdomme bruger anseelig tid på rutineundersøgelser. Diabetespatienter skal eksempelvis have undersøgt om deres vitale værdier (f.eks blodsukkerniveau eller lignende) er på et niveau, der er acceptabelt i forhold til deres sygdom.
Grundet større tidspres ønskes det, at omfanget af disse undersøgelser minimeres eller forbedres, så de kan udføres hurtigere og mere effektivt for både patient og læge.
Med hensyn til vente- og undersøgelses tid, kan det nærmest direkte overføres til produktionstab for landet. Et tænkt eksempel med indlæggelselstal fra Danmarks Statistik viser, at der i 2009 var 1.157.172 indlægelser[17]. Det antages at patienten bruger 15 minutter (1/4 time) på ventetid, og ingen minutter på transport.

$$\frac{1.157.172}{4} = 289.293\text{timer}$$

Med en gennemsnitsløn på ca. 195 DKK[15] vil dette være ca. 56 millioner DKK om året i produktionstab.
Naturligvis skal disse tal tages med forbehold, da patienter udenfor arbejdsmarkedet, eksempelvis børn og pensionister, ikke er trukket fra. Modsat er der kun beregnet med 15 minutter hvilket er lavt sat, og uden transport. Tallet må formodes reelt at være endnu højere, derudover vil der også være situationer, hvor patienter udebliver og det derved er personalet der får spildtid.

Verdensbefolkningen vokser, og andelen af ældre i befolkningen bliver større. Dette betyder ikke at de ældre bliver svagere, men det vil stadig betyde, at vores sundhedsvæsen vil få en større arbejdsbelastning. Danmarks Statistik har lavet flere undersøgelser og prognoser, der påviser at den ældre del af befolknigen vil udgøre en større og større procentdel. Figurerne 1.1, 1.2 viser de omtalte fremskrivninger[38].



Figur 1.1: *Model fra Danmarks Statistik der viser grafisk befolknings statistik fra 1950-2050*

BEFOLKNINGENS ALDERSSAMMENSÆTNING

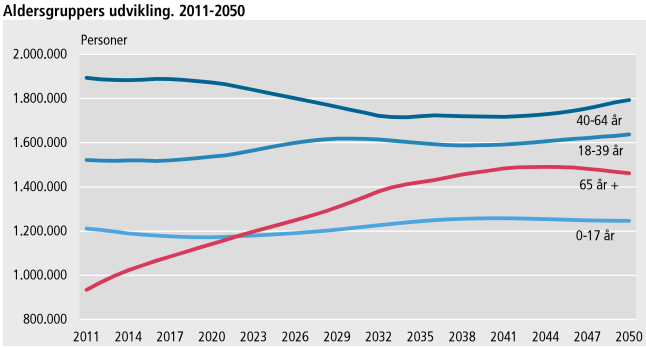
1. januar	1950	1975	2000	2025	2050
			mio.		
Befolkning, DK	4,3	5,1	5,3	5,7	5,8
			mia.		
Befolkning, verden	2,5	4,1	6,1	8,0	9,2
			pct. af befolkningen		
0-14-årige, DK	26,2	22,7	18,4	16,6	16,6
15-59-årige, DK	60,6	58,7	61,9	55,0	52,9
60 år og derover, DK	13,2	18,6	19,7	28,4	30,5
0-14-årige, verden	34,1	36,8	30,2	24,1	21,0
15-59-årige, verden	57,8	54,7	59,9	60,9	58,3
60 år og derover, verden	8,1	8,5	9,9	15,0	21,8

1950 1960 1970 1980 1990 2000 2010 2020 2030 2040 2050

Figur 1.2: *Model fra Danmarks Statistik der viser tabel over befolknings statistik fra 1950-2050.*

Ud fra figur 1.2 ses at personer over 60 år i 2000 udgjorde 19,7% af Danmarks befolkning og at den i 2050 estimeres til at, udgøre 30,5%. Altså ca. 50% større. På verdensplan bliver procentdelen af „ældre over 60 år“ mere end fordoblet fra 2000 hvor de udgjorde 9,9% til år 2025, hvor de udgør 21,8%.

Figur 1.3 viser, 1.491.000 personer fra den danske befolkning vil være fyldt 65 år i 2044, modsat de 934.000 personer der er i dag. Dette svarer til en stigning på 557.000 personer eller ca. 60%[16].



Figur 1.3: *Model fra Danmarks Statistik der viser, hvorledes den ældre befolkning vil øges over tid.*

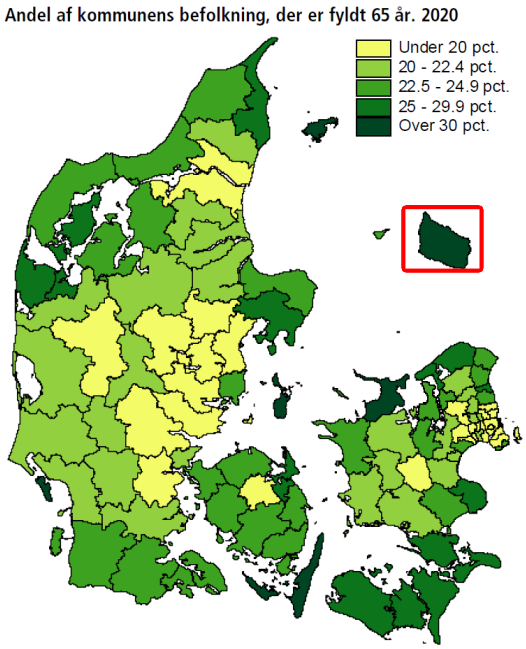
I følge Danmarks Statistik[16], samt figur 1.4, kan følgende aflæses.

> „For de 20 kommuner, der i 2011 havde den højeste andel af personer, der var fyldt 65 år, vil andelen i gennemsnit således stige med 5,6 procentpoint fra 2011 til 2020. Til sammenligning vil de 20 kommuner, som i 2020 havde den laveste andel af personer, der var fyldt 65 år, kun have en gennemsnitlig stigning i andelen på 3,1 procentpoint fra 2011 til 2020.“

En anden udfordring er, at fremskrivninger viser, at ældre flytter til udkantsdanmark. Figur 1.4 viser, hvilke dele af landet, der formodes at blive mest belastet. I Danmark er der samtidig en tendens, hvor de mindre hospitaler i udkantsområderne lukkes og der i stedet bygges store hospitaler (superhospitaler), hvor specialisterne samles under et tag.
Disse tendenser med flere ældre i udkantsområderne og få store hospitaler, kan betyde, at mange patienter får længere transporttid til undersøgelsesstedet. Hvis disse ældre alle har forladt arbejdsmarkedet, kan samfundet økonomisk set finde transporttiden irrelevant, men idet det antages at folk i fremtiden først pensioneres i en højere alder, vil der være et økonomisk incitament til at minimere transporttid.

Samtidig med, at der forventes flere indlæggelser, forventes prisen på den kommunale medfinansiering på en indlæggelse også at stige fra en DRG-takst (se afsnit 1.4) på maksimalt 4973 DKK[36] til maksimalt 14025 DKK[35] pr. indlæggelse i 2012 (Staten betalte mere før). Hvis kommunerne med et billigt telemedicinsk system kan undgå visse indlæggelser og rutineundersøgelser, vil de både spare penge, tid og ressourcer, uden at det forringer hverken sundheden eller tryghedsfølelsen hos patienten.

For at optimere lægernes arbejdsgang, arbejdes der på, at indføre mere og bedre telemedicin, som dækker hele landet.



Figur 1.4: *Model fra Danmarks Statistik, viser fordeling af ældre over 65 år i kommunerne i år 2020[16]*

Pallas Informatik har i projektet „Early Warning System“ (EWS) der omhandler obsersvationsskemaer på hospitaler for at optimere sygeplejerskers daglig dag og mindske dobbelt arbejde, dette kan f.eks være hvor der måles blodtryk, puls, iltmætning, respirationsfrekvens, temperatur og bevidsthed og dette registreres minimum én gang pr. vagt pr. patient, og op til fire gange i timen afhængig af patientens tilstand.

Følgenede observationstal er lånt med tilladelse fra læge, Lisbeth Schjerling (ansat på akut modtageafdeling ved Hillerød hospital). Beregningen er lavet i forbindelse med dette projekt.
Der er foretaget en tidsmåling på, hvor lang tid det tager at måle og indplotte disse data på et antal patienter, hvorefter følgende er blevet beregnet:
For 35 patienters tager det 18 timer at måle og registrere værdier, hvis alle patienter skal have målt værdier minimum én gang hver 8. time. Dette svarer til ca. til 30 minutter pr. patient pr. dag.

For Region Hovedstaden, der med 4500 sengepladser og derved bruges der op til 2.250 arbejdstimer dagligt til observationsskemaer, betyder det at med en sygeplejerskelønning på 180 DKK/t, at der dagligt bruges 405.000 DKK[41].
Det skal noteres, at denne halve million om dagen, er beregnet med et absolut minimum af oberservationer, og når der er fuld belægning på hospitalet.

Hvis et system til overvågning indføres, kan man formentlig halvere denne omkostning og på sigt spare millioner.

Andre firmaer undersøger også muligheder for telemedicin, deriblandt er danske DELTA. DELTA's Claus F. Nielsen, repræsentant i Statens Rådgivende udvalg for telemedicin udtaler[22]:

> „Det kræver valg af fælles infrastruktur og standarder.
>
> Hundredvis af forskellige it-systemer gør sundhedsvæsenet til et moderne babelstårn af mange sprog og kommunikationssnitflader. Og teknologier der ikke kan kommunikere indbyrdes eller anvende samme platform, er en alvorlig hindring for visionen om sammenhængende telemedicin.
>
> Forestil dig, at du skal skifte dit bluetooth-headset ud, hver gang du får en ny mobiltelefon, fordi hver producent er nødsaget til at bruge sin egen, proprietære platform. Sådan fungerer det langt hen ad vejen i sundhedsvæsenet i dag – systemerne taler ikke sammen. Derfor er en fælles infrastruktur og valg af standarder en nødvendig forudsætning - både for at skabe interoperabilitet mellem forskellige medico-produkter og softwaresystemer - og for en storskala, tværsektoriel udbredelse af telemedicin.“

Claus F. Nielsen taler implicit for en løsning på tværs af firmaer, og at denne skal være opbygget således, at der er et nemt tilgængeligt interface til systemerne. Dette vil man kunne løse ved at data og journaler kan tilgås via webservices. Brugen af webservices muliggør også, at der laves systemer, der grundlæggende laver datamining på nuværende systemer, ved hjælp af f.eks mashups.
Af samme årsag belyser denne rapport, hvordan en telemedicinsk arkitektur kan opbygges og gøre denne datatilgang mulig for forskellige faggrupper, som forskerer, læger og teknikerer.

Det ønskes at nedbringe den tid en læge skal bruge på rutineundersøgelser. Naturligvis vil sundhedspersonale stadig bruge tid på at skulle oprette en patient i et telemedicinsk system, men tiden det tager, vil være minimal i forhold til gentagne rutineundersøgelser.

Systemet skal være så nemt for patienten som muligt, og patienten skal på intet tidspunkt føle sig tynget af at være „overvåget“. Tværtimod ønskes det at patienten føler sig mere fri til, at gøre alt det normale i sin hverdag og ikke blive i hjemmet af angst for evt. at få et anfald i supermarkedet eller lign.

Et system kan naturligvis ikke overtage alle processer helt, da sundhedspersonale har et menneskeligt aspekt, som er nødvendigt for at vurdere helheden. Et system kan dog blive et godt værktøj til indsamling af data og derved minimere tiden hvori patient og sundhedspersonale mødes.

Der er mange faktorer, der skal gå op, og projektet kommer ikke rundt om alt, men lægger op til at faggrupperne bidrager til den del, de hver især er bedst til. Af samme grund skal diagnosticeringsarkitekturen, være så fleksibel som mulig.

De matematiske forskere skal kunne danne nye algoritmer og vælge, hvilke parametre denne algoritme skal bruge for at fungere.
En patient skal ikke kun have én algoritme til at analysere blodsukkertallene, men måske have fire, der analyserer på hver deres matematiske måde.
Det skal medregnes at nogle lidelser kan måles ud fra ét vital parameter (blodsukker, puls osv.), mens andre kræver flere, hvilket stiller krav til fleksibilitet i arkitekturen.

Hvis forskere kan få tilgang til automatisk genereret data om patienter, vil man ud fra datamining også kunne skabe ny indsigt i sygdomslære.
Det stiller krav til anonymisering af data, da udvedkommende ikke skal kunne hente oplysninger om patienter. Data kan misbruges og patienten skal derfor beskyttes.

Dette projekt vil opstille forskellige arkitekturer og foretage en diskussion af disse. Dette gøres med henblik på at finde det bedst mulige grundlag for en arkitektur, der er fleksibel og tillader at de forskellige faggrupper kan tilgå den ønskede data.

Der undersøges om brugen af intelligente sensorer, dvs. der selv kan sende data til en internetserver, har fordele frem for et mellemled der modtager, pakker ind og sender data videre til nettet. I dette område evalueres det på pris, men også på hvad der er bedst i forhold til regulativer og regler på området.

Projektet ender ud med et forslag til en arkitektur samt implementeringsforslag der kan realisere dette.

1.1 Problemstilling

Projektet beskæftiger sig med, at oprette en generel arkitektur for telemedicinske systemer, beregnet til indrapportering af data og analyse heraf.
Derudover undersøges det hvilke faggrupper, der ønsker tilgang til denne type system, samt hvilken data disse forskellige faggrupper skal kunne tilgå.

Projektet belyser problemer omhandlende arkitekturer, responstider og data/-patient genkendelse.

For at simplificere opstillingen beskrives arkitekturforslaget i to delafsnit; et, der beskriver et netværk, der er tæt på patienten og et andet med netværket på en server.

1.2 Antagelser

Der er ikke taget højde for udfordringen med, at få alle producenter til at indgå i et samarbejde om netop den beskrevne arkitektur. Det må formodes at de fleste producenter vil foretrække deres egen komplette løsning, da de derved har en bedre businesscase.

Samtidigt vil myndighederne fastsætte krav til, hvordan data skal placeres og anonymiseres. Eventuelt skal der udarbejdes en fælles dansk europærisk standard, for derved at sikre et bredere samarbejde til f.eks de diagnosticerende algoritmer.

Der tages derudover heller ikke forbehold for følgende punkter i rapporten, og de antages derfor som givet.

- Kommunikation mellem en eller flere dele af systemet foregår over en sikker protokol/linje.
- Lagring af data, er ikke begrænsning i håndtering eller mængde.
- Krav for personfølsom data.
- Patienter ønsker at have flere sensore på kroppen hver dag.
- Der findes en evt. plasterteknologi hvori en sensor kan monteres.
- Riskoen, hvis TDCs GSM dækning svigter er minmal.
- Forbehold ved fejlalarmering.
- Patienten opholder sig udelukkende i Danmark.
- Fagfolkene er trænede i at benytte systemet.

1.3 Rapport kapiteloversigt

En kort oversigt over hvad de respektive afsnit indeholder kan ses nedenfor.

- **Introduktion**
  Beskriver problemer, og hvorfor der er brug for en telemedicinsk arkitektur.
- **Telemedicinske projekter**
  Beskriver grundliggende telemedicin samt hvilke projekter, der har været indenfor området, som står frem i forhold til denne rapports tanker.
- **Telemedicinsk arkitektur**
  Omtaler telemedicinske arkitekturer hos patienten såvel som serverside.
- **Analyse**
  Viser hvilke overvejeler og valg der er gjort på baggrund af fordele og ulemper fra afsnittene Telemedicinske projekter & Telemedicin arkitektur.
- **Implementeringsforslag**
  Giver forslag på implementeringer, der kan være til gavn for det endelige produkt, hvor mediator, webservices og bus arkitekturer diskuteres.

- **Evaluering**
  Indeholder to forskellige brugsscenarier, samt Bluetooth undersøgelse af protokollen.

- **Konklusion**
  Fortæller hvad der blev fundet som bedst mulige arkitektur og hvorfor denne er valgt.

- **Fremtidige studier**
  Beskriver hvad der skal laves efter denne rapport, hvis det ønskes at dette projekt skal fuldføres.

- **Appendix**
  Bilagene indeholder den komplette XML fil, der er brugt til en transmissions beregning.
  En VDM model der blev oprettet i starten af projektet for at give et indblik i om projektet kunne lade sig gøre.
  Samt kildekode til Android Accelerometer programmet der sender sine data til en server.

1.4 Ordliste

En liste over nogle af de ord og forkortelser, benyttet i rapporten:

Pallas	Pallas Informatik A/S
DAO	Data Access Object
P2P	Peer to peer
DRG	Diagnose relaterede grupper
DRG takster	Er beregnet som landsgennemsnitlige omkostninger ved behandling af patienterne på de offentlige danske sygehuse.[43]
Mashup	Et program eller website dannet ud fra data opsamling fra forskellige kilder
BT	Bluetooth
Starvation	Hvis et program lider af datamangel, som hindre dens virke.
BLE	Bluetooth Low Energy
nPOCT	Net-enabled Point of Care Technology
RFID	Radio Frequency Identification
JSON	JavaScript Object Notation, et XML lignende format.
Heartbeat	Et signal der sendes fra punkt til punkt i en kommunikation, for at være sikre på begge ender „lever“
Cache	Midlertidig hukommelse i et system.

Kapitel 2

Telemedicinske projekter

Dette afsnit indeholder, hvad telemedicinbegrebet er, og hvad det skal kunne forbedre fremfor de de eksisterende løsninger.

Derudover nævnes nogle „State of the art“ projekter som viser at der findes nuværende projekter, der omhandler det samme emne, men griber det anderledes an, disse projekter vil være indgangsvinkel til nogle af problemstillingerne i dette projekt.

2.1 Telemedicin

Telemedicin er en fælles betegnelse for et system, der sammenfletter læger og patienters interaktion over nettet.[40][28]
Telemedicin er blevet udviklet gennem de sidste år, i kraft af man begyndte at få bedre IT systemer, samt hurtigere og mere stabile internetforbindelser.
Begrebet telemedicin dækker derved over, at patient og læge ikke behøver at være på samme geografiske lokalitet når en undersøgelse finder sted.

Et godt eksempel på dette er, hvis en patient bor i Danmark og specialisten til netop patientens sygdom er fra Japan. I sådanne tilfælde skal man enten have

patienten til lægen eller omvendt.
Ved at bruge første stadie af telemedicin vil patient og læge kunne tale sammen via en videokonference, og et dansk hospital kan evt. undersøge patienten. Resultaterne af undersøgelsen sendes gennem et system til Japan, hvor specialisten undersøger disse og svarer tilbage gennem systemet.
Derved sparer man at sende patienter jorden rundt for at lade dem undersøges af specialister.
En anden stor force er, at patienter med eksempelvis diabetes kan foretage målinger hjemme og et IT system sender målingerne til lægen, som efterfølgende evaluerer dem. Dette sparer både læge og patient for tid på transport, ventetid, introduktion, undersøgelse osv.

Formålet med telemedicin er simpelt: Der skal spares tid og penge samtidig med at der skabes tryghed for patienterne
Patienter der har en lidelse kan have tendens til at gemme sig i hjemmet og begrænse deres normale liv, da de er bange for der skal ske dem noget. Med telemedicin kan man lave en overvågning af patienten, så uanset hvor patienten er, kan denne føle sig „overvåget“ og tryg.

Der bliver flere og flere teknologier og ideer til telemedicin, og kan være svære at opsummere.
Dog er der nogle firmaer, der indenfor samme område som dette projekt, har udemærket sig , hvorved de vil blive betragtet nærmere i næste afsnit. Projekterne giver et indblik i hvordan andre har lavet telemedicinske løsninger og der kan evt. tages højde for de udfordringer de har mødt.

2.2 IctalCare

IctalCare er et dansk firma, der blev grundlagt i 2007, i samarbejde mellem DELTA og Coloplast.
IctalCare fokuserer på patienter med epileptiske anfald (toniske og tonisk- kloniske anfald). Epilepsi er en udbredt lidelse som mange i verden har, og anfald kan tydeligt genkendes ved, at patienten oftest vil falde om, og få et krampe anfald.
Disse kramper er ikke i sig selv farlige, men meget udmattende for denne og sammenlignes med at løbe et marathon. Det kan derimod blive farligt, hvis anfaldet opstår i en situation, hvor personen eksempelvis får et anfald mens der køres bil og vedkommende mister herredømmet over bilen. Det kan være farligt for patienten selv og evt. andre involverede i en ulykke. Af samme årsag, kan mange epileptikere ikke få kørekort.
Udover dette kan patienterne også være udsat for at falde og selve faldet kan

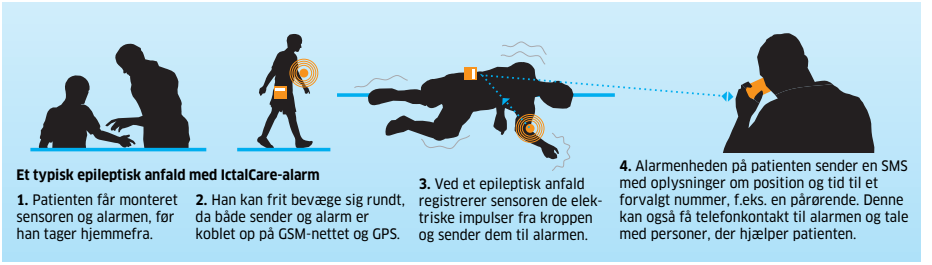
gøre stor skade på deres krop, da patientens reflekser lammes. Det kan sammenlignes med at patienten falder bevistløs om, og derfor ikke tager fra inden faldet.

Praktiserende læge Ib Koldbæk (Feburar 2012) udtaler, at epileptiske anfald ofte lukker for ilten til hjernen og den derfor kan tage skade af dette.
Han mener, at hvis man kan advare en patient bare sekunder før et anfald vil disse være til hjælp for patienten, der ville kunne lægge sig ned og forberede sig på et anfald.

Ved opslag på Epilepsiforeningens website den 14/2-2012[12] bekræftes det, at anfald kan være farlige og bør forebygges, hvis muligt.

> „**Er epileptiske anfald farlige eller skadelige?**
> Vi ved, at hjernen lider skade efter mange anfald. Et enkelt eller nogle få anfald giver ikke påviselige skader. Hvis man har hyppige anfald, vil de før eller siden skade hjernen, så f.eks. hukommelsen kan blive nedsat. Derfor er det vigtigt at forebygge anfaldene. „

Kim Gommesen, CEO IctalCare (Hørsholm, Dec-2011) informere at der på det nuværende markede, ikke eksisterer sensorer til epilepsi, der sidder på patienten. Der er dog produkter på markedet, der måler på rystelser i f.eks sengen, men denne løsning virker naturligvis kun når patienten er i sin seng og altså ikke befinder sig nede i f.eks indkøbscenteret eller på arbejde, hvor spontane anfald vil være farlige.
IctalCares løsning, der endnu ikke er kommet på markedet, vil kunne indrappotere anfald uafhængig af hvor patienten befinder sig.



Figur 2.1: *IctalCares koncept tegning (Billede udlånt af IctalCare)*

IctalCare sætter fokus på, at have et telemedicinsk system, hvor patienten bærer både sensor og gateway (som IctalCare kalder A-enhed). Sensoren indeholder både beregninger og logikken i systemet. Den sender kun alarmering ud til gatewayen, når patienten har et anfald. Gatewayen vil ved alarm, sende alarm

videre ud til en server via det mobile GSM netværk.
Ved at bruge GSM-netværket mener IctalCare, at de har dækning over stort set hele Danmark og der pt. ikke er et netværk, der er bedre dækkende og bedre testet end dette.

IctalCares nuværende løsning kan alarmere ved igangværende epileptiske anfald og altså ikke lave en fremskrivning.

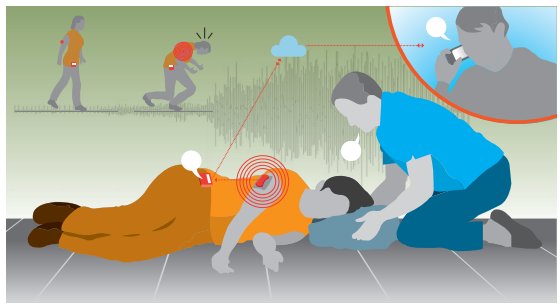
Figur 2.2: *Billedet viser IctalCares plaster, hvor sensoren og batteri er indbygget. (Billede udlånt af IctalCare)*

Alarmeringen er derfor ikke til patienten selv, men til pårørende, f.eks til forældre til et barn med epilepsi.

IctalCare har indbygget mikrofon og højtaler i gatewayen og udnytter at de allerede har GSM netværket. Disse features gør at pårørende kan ringe til gatewayen for at spørge patienten selv om de er ved bevisthed og derved afklare om det er en falsk positiv alarmering opstået ved f.eks voldsom leg eller lign. Ved et evt. anfald kan den pårørende, fortælle omkringstående folk, at patienten har et epileptisk anfald og at hjælp er på vej, derved oplyses folk der observere anfaldet, at patienten altså ikke er narkoman, alkoholiker eller lign.

Nogle af de features IctalCare løsningen har, er:

- 100% trådløs løsning.
- Lav båndbredde (sender kun at der er alarm).
- Lav GSM trafik.
- Algoritme på sensor (Fast Fourier transform).
- Keep alive (gateway sensor) (gateway server).



Figur 2.3: *Viser at pårørende kan tale med omkringværende gennem gatewayen. (Billede udlånt af IctalCare)*

- Data sendes ikke ud.
- GSM (ingen WiFi).
- Fanger 100% af anfaldene (baseret på IctalCares egne undersøgelser).
- Ingen fremskrivning, fanger kun når patienten har et anfald.
- Koster ca. 700kr pr. måned.
- Højtaler og mikrofon i gateway.
- Forsørger kan ringe til A-enheden (gateway).

IctalCare markedsfører deres løsning ved[23]:

> „Ca. 55.000 danskere har epilepsi. Hvert år får ca. 4.500 danskere konstateret epilepsi. De fleste får det enten som børn eller som ældre. Ca. 2/3 bliver anfaldsfri ved hjælp af medicin, men den sidste 1/3, der ikke bliver anfaldsfri, har brug for hjælp og pleje, i forbindelse med et anfald. IctalCare 365 Epilepsi Alarm er beregnet til netop
>
> denne gruppe. Den sender øjeblikkelig og pålidelig besked om et påbegyndende tonisk og tonisk – kloniske anfald til forældre eller plejepersonale. Den giver **ny frihed og tryghed** for alle.„

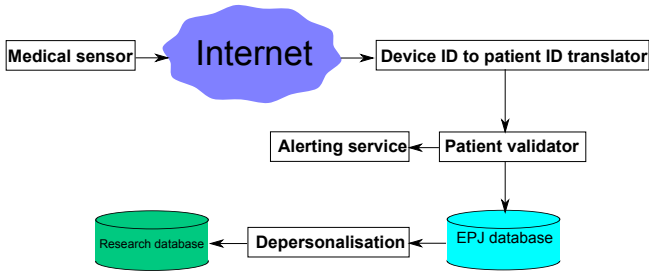
2.3 Secure Linking of Patients and Tele-medical Sensors

Et vigtigt aspekt i forbindelse med sikkerheden ved brugen af et telemedicinsk system beskrives af Michael Bojsen[4]. Her påpeges problematikken ved, at patienternes data kan blive forvekslet, og at dette kan ske både utilsigtet eller tilsigtet, for at snyde systemet til en evt. kraftigere dosering af medicin.

Der fremhæves flere begrundede faktorer for risikoen for patienter snyder, hvilket medfører en fejldiagnosticering, og dermed en forkert medicinsk behandling, kan finde sted. Hvis systemet står for medicineringen af patienterne kan disse risikere at dø, hvis der behandles med enten for lille eller for stor dosis.

Projektet fokuserer på at lave en analyse på indkommende data, for derved at kunne evaluere om denne data er sammenlignelig med patientens normale data. Der benyttes statistiske modeller samt neurale netværk for at vurdere om disse konklusioner er korrekte og om data tilhører patienten.
En anden faktor, der begrundes i projektet er, at sensor eller datasender (gateway) ikke, via CPR-nummer, IDnummer eller lign. må have kendskab til, hvem dataen tilhører. Dette gøres ud fra princippet om, at hvis sensoren kender dette, kan dette laves om, hvilket igen vil skabe risiko for fejl og snyd.

Figur 2.4 illustrerer hvor i en arkitektur det ønskes at denne validering skal placeres. Den placeres tidligt efter modtagelse, så man ikke risikerer, at dataen når så langt op i strukturen og bliver analyseret som personens egne.



Figur 2.4: *Oversigt over hvordan systemet er sat sammen (Billede udlånt af Michael Bojsen)*

I projektet benyttes data fra pulsure som beregningsgrundlag. Det viser sig dog, at en pulsmåling er svær at individualisere og derfor også svær at kunne differenciere og derved udelukke om det er den ene eller andens patients data, der modtages.

Tankegangen i projektet viser dog et stort potentiale og det vurderes, at hvis man brugte flere faktorer end kun pulsen, ville et lignende system forholdvis uden problemer kunne integreres i den løsning, der beskrives i denne rapport.

2.4 Opsummering

Det er sjældent, at man er den første, der får en idé til noget, dette beviser de to udvalgte projekter, der har tilknytning til telemedicin indenfor samme område som dette projekt også har.
Disse projekter har givet lidt indsigt i, hvordan andre anser mulighederne ved et telemedicinsk system.
Der findes mange flere projekter end disse to, men de fleste har været lukkede projekter, hvor det ikke har været muligt at få detaljer.

Kapitel 3

Telemedicinsk arkitektur

I dette afsnit undersøges forskellige arkitekturer, samt hvordan rækkefølgen af elementerne i en generel arkitektur kan gøre en væsentlig forskel for den datamængden, der sendes rundt i systemet.
Det har været svært at finde telemedicinske systemer, der bygger på samme problemgrundlag, som har ønsket at videregive hvordan deres arkitektur er opbygget. Dette har gjort at afsnittet har ført til mere grundforskning, end først antaget.
Arkitekturene opdeles i to sektioner, det adskilles ved om der er tale om serverdelen eller netværket af sensorer, der er tæt på patienten.

I en telemedicinsk arkitektur, som skal kunne analysere og alarmere patienter er der nogle kriterier, der skal overholdes for at funktionaliteten sikres:

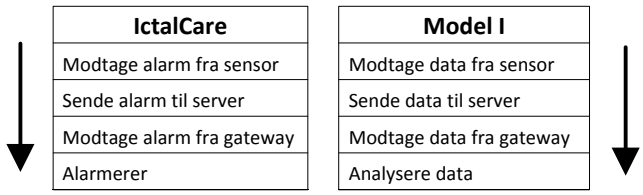
- Modtage data fra sensorer.
- Sende data til server.
- Analysere data.
- Alarmere patienten.

Hvis modellen skal udvides for yderligere funktionalitet, skal den endvidere indeholde:

- Kunne gemme modtaget data.
- Normalisere data, for at lave generelle analyser.
- At kunne benytte gemt data til at forbedre algoritmer.
- En analyse kan benytte data fra flere tilgængelige sensorer.

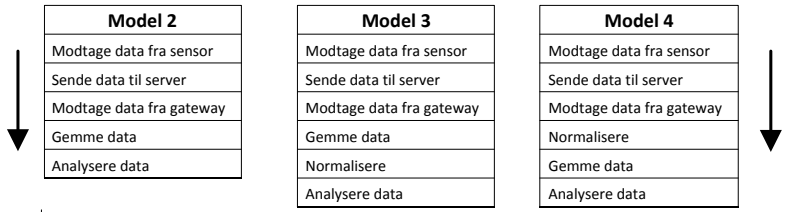
Det kan betragtes som Lego klodser, der skal stables i den bedst mulige rækkefølge, for at få den optimale funktionalitet.
Samtidigt skal rækkefølgen overvejes i forhold til om data- og sensorinformationer er nødvendige i alle „bokse“ eller denne data kan sorteres fra tidligere i forløbet, så man ikke bygger videre på et fundament med irrelevante oplysninger.

Nedenstående figurer skal ses som forskellige modeller. De skal læses oppefra og ned. Hvert felt repræsenterer et lag i den telemedicinske arkitektur.



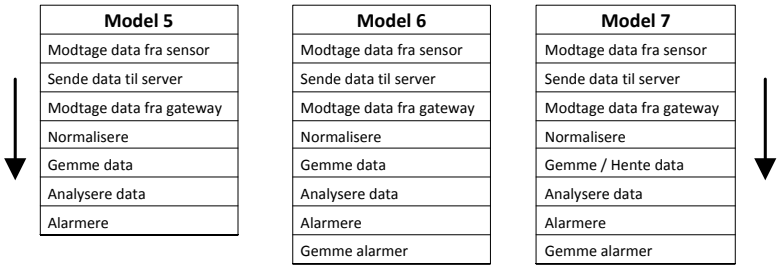
Figur 3.1: *IctalCare løsningen samt en simpel model.*

Figur 3.1 viser IctalCares løsningens lag, samt en simpel model, der byder på begrænset funktionalitet.
Begge har det til fælles, at data smides væk efter brug. Dette gør at forbedring af algoritmer på sigt ikke kan lade sig gøre, da man ikke vil have noget datagrundlag for dette.



Figur 3.2: *Viser tre modeller hvor det illustreres at normaliseringslaget kan indføres forskellige steder i strukturen.*

Derfor har Model 2 figur 3.2 fået introduceret et lag der gemmer data efter modtagelse.
Model 3 indføres normalisering efter data er gemt. Normaliseringen vil gøre det muligt at alle værdier er i samme enheder (KG, $^{mmol}/_{L}$ osv) og andre normaliserings krav. Model 4 normaliserer data først og gemmer den derefter, det kan diskuteres om dette skal gøres før eller efter. Det er fundet mest hensigtsmæssigt at data gemmes efter normaliseringen, som i model 4, da man derved har muligheden for at gemme både den normaliserede værdi samt rådata. Ved at have begge dele vil man kunne bruge data senere, til at finde fejl eller forbedre systemet.



Figur 3.3: *Viser tre modeller hvor alarmering kommer ind i billedet, Model 6 bidrager med et lag der gemmer disse alarmer for senere at kunne benyttes ved undersøgelser hos lægen eller lign. Model 7 ændrer „gemme" laget til også at kunne være et lag der kan hente lagret målinger til brug af træning af f.eks neurale netværk.*

Hvis systemet skal kunne alarmere, skal der naturligvis indføres et lag, der kan håndtere dette, figur 3.3
Model 5 placerer dette lag naturligt efter analysen, da alarmeringen skal bruge analyserne for at kunne alarmere.
I Model 6 ønskes der at gemme alarmerne; dette gøres blandt andet for at kunne træne algortimer bedre på sigt. Hvis man kun har data, kan man ikke træne algoritmerne, så alarmerne er nødvendige for at kunne undgå falsk positiv eller lign. fejl alarmeringer.
Netop ved træning af algoritmer, er der også behov for at kunne tage gammel data med ind i arkitekturen for at kunne træne netværk og teste diagnosticeringsalgoritmer. Derfor bliver lagring (save) laget til et save/load lag i Model 7.

Generelt skal man gøre sig overvejelser omkring hvornår data gemmes. Hvis der gemmes tidligt er data ikke normaliseret, mens man er nødsaget til at transportere potentielt unødvendigt data med op langs lagene i systemet, såfremt der gemmes til sidst.

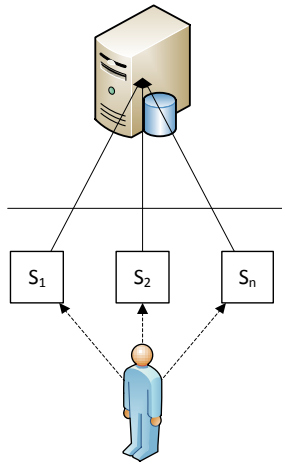
3.1 Opbygning

De første spørgsmål, der opstår i forbindelse med et telemedicinsk system, vil ofte være, hvordan aktikteturen skal opbygges, samt hvor i beregningen at analysen af data skal placeres i systemet?
Generelt kan det siges, at de specialiserede systemer, hvor beregningen placereres i yderste led, (tættest på patienten i dette tilfælde) ofte resulterer i dyrere enheder, men en billigere i datakommunikation. De dertilhørende problemer, som kontinuerlig sending af data kræver at enheden altid er online, hvorved der kræves mere end blot et WiFi modul.

For at simplificere billederne anvendes følgende forkortelser:

- **S** er en sensor.
- **M** modtagelses/dispatcher punkt (server side).
- **N** normalisering.
- **IO** dette er save/load laget som også stiller data tilrådighed for diagnosticering.
- **D** diagnosticeringer.
- **P** er en patient, på serverside.

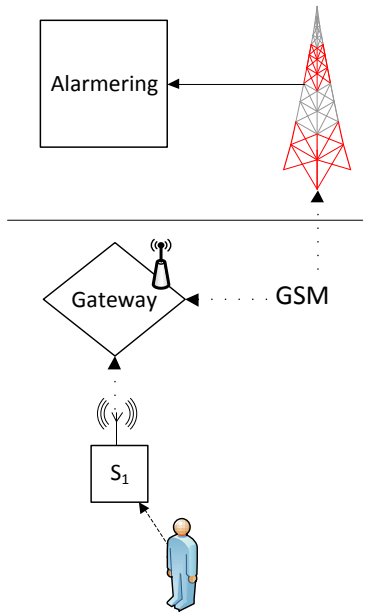
Det første eksempel er en simpel opbygning som vist på figur 3.4. Denne opbygning er baseret på, at en patient har et vilkårligt antal sensorer, der selvstændigt sender sensorenes data til en ekstern server, hvor data behandles. Dette er en typisk telemedicinsk tilgang, hvor data sendes på nettet til behandling eller alarmering. Mange telemedicinske systemer kan betragtes ud fra en simpel model, dog skal det overvejes at den indre serverarkitektur ikke analyseres i denne figur.



Figur 3.4: *Simpel opsætning, hvor tre sensorer selstændigt sender data til en server.*

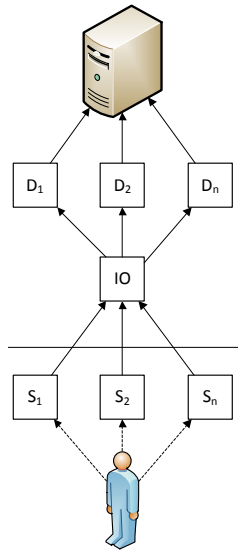
IctalCare, der beskrives i afsnit 2.2, har placeret størstedelen af intelligensen på sensoren og har derved lav kommunikation mellem enheder og server.
Grundet mindre datakommunikation kan dette give dem nogle fordele, men da de ikke sender data ud på nettet kan de have problemer med at lave nye algoritmer, da de mangler et datagrundlag som udgangspunkt.
Fordele og dlemper ved Ictalcares løsning kan læses i afsnit 2.2.

Kort kan det forklares og ses på figur 3.5 at IctalCare har én sensor der sender til én gateway, og denne gateway står for kommunikationen ud på nettet.
For at sikre at kommunikation opretholdes benytter Ictalcare heartbeat mellem server og gateway, samt mellem gateway og sensor, som sikre dem at de altid ved om der er forbindelse mellem enhederne.



Figur 3.5: *IctalCares arkitektur, hvor sensor sender til trådløst til gateway, og gatewayen sender alarmer over GSM.*

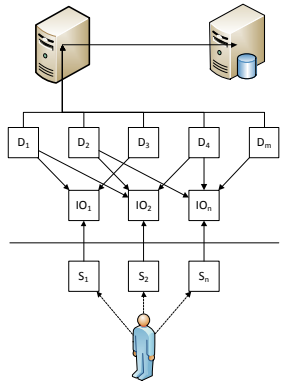
Figur 3.6, viser en model hvor man har flere sensorer pr. patient. Her sender hver sensor sin egen data ind til et centraliseret modtagelsespunkt, hvor den deles ud på sensor niveau igen, for til sidst at lave en enkeltstående analyse af sensorens data.
Dette vil f.eks være funktionelt med diabetespatienter, hvor man ofte kun måler blodsukker.



Figur 3.6: *Simpel løsning, med én patient med flere sensore, og oplysningerne sendes til en centraliseret modtagelse*

Hvis en model med en mere avanceret arkitektur ønskes, bringes nye overvejelser i betragtning.
Dataen fra sensoerer skal behandles, for at gøre det muligt at forudse, at en patients tilstand forværres eller er ved at blive kritisk. Derfor er det ofte svært kun at benytte en enkelt sensors data, hvilket „Secure Linking of Patients and Tele-medical Sensors“ afsnit 2.3 var et godt eksempel på, hvor genkendelse af en enkelt patient baseret udelukkende på pulsmålings dataene var besværlig..

Figur 3.7, side 26, viser et system, hvor tre sensorer sender data som modtages på serversiden, derefter analyseres på tværs af diagnosticeringer og til sidst gemmes i en database. Lagringen sker for at kunne genbruge eller bruge disse som bevis for ny funktionalitet for algoritmer er korrekt, eller som datagrundlag til forskning i specifikke sygdomme. Forskellen fra figur 3.6 er at en diagnosticering nu kan få data fra flere sensorer til analysen om patients tilstand.



Figur 3.7: *Arkitektur der gør det muligt at analysere på flere sensorer samtidig*

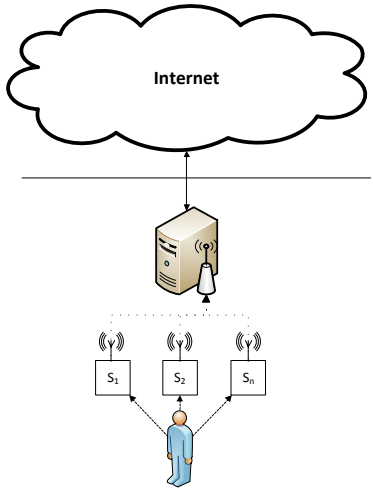
Der er flere måder hvorpå en telemedicinsk arkitektur skal defineres.
I nedestående afsnit vil flere arkitekturer samt deres fordele og ulemper beskrives, både inden for server- og klientdelen.

3.1.1 Patientnetværk

Det er valgt samtidigt at lave en arkitektur på patienten selv, kaldet et patientnetværk også kendt som Wireless Body Area Network (WBAN)[26] [42] Med et patientnetværk menes et netværk, der er i nær tilknytning til patienten. Netværket kan derfor bestå af et vilkårligt antal sensorer, som kan sende data til en server eller en patient tilknyttet en telemedicinsk PC.
Netværket er altså et lukket netværk, der enten er fast installeret i hjemmet eller følger med patienten rundt.

På figur 3.8, side 27, præsenteres en løsning med at have en patientcomputer i hjemmet. Hver enkelt sensor har trådløs forbindelse til en PC; denne PC i hjemmet vil være en del af det telemedicinske netværk og kan bruges til flere ting; f.eks kan al processering af patientdata foretages på denne PC. Derved sendes der kun data ud af patientens hus, når der sker en alarmering på baggrund af den processerede data. PCen kan alternativt bruges som et mellemled, der opsamler sensordata og sender dem i én samlet pakke, der sendes til serverne i det format den eksterne processesring kræver.

Denne løsning er brugbar, men omkostningsrig, da alle sensorer skal have WiFi eller anden forbindelsesteknologi til PCen og kræver, at alle patienter har en



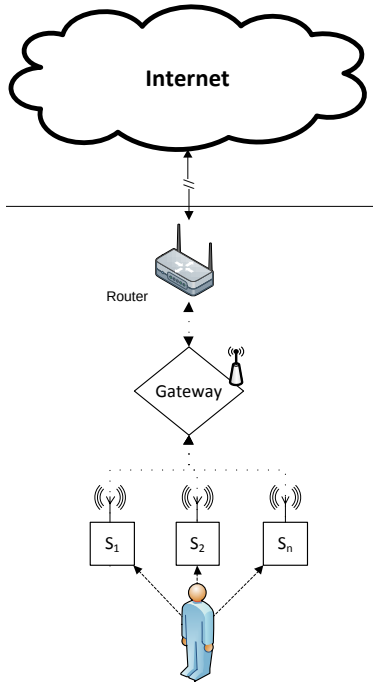
Figur 3.8: *Patienten har en patient PC hjemmet der står for størstedelen af systemet*

patient PC, som koster en del at etablere. Fordelen er, at datakommunikationen ud af huset er lille og vil derfor ikke have en nævneværdig indflydelse på omkostningerne.

Ved at placere processeringen hos patienten, opstår der problemer når processeringensalgoritmerne skal opdateres. Dette er muligt for udstyr installeret i patientens hjem, men vil ofte resultere i at algoritmerne, der bruges til analysen vil være de samme i flere år pga. manglende opdatering, eller indtil patienten bliver udstyret med en ny patient PC.
En anden problematik ved denne løsning er, at data fra patienten ikke sendes ud til en centraliseret placering, som derved kan benyttes ved fremtidig udvikling af forbedrede algoritmer.
Ved at gemme sensordata, er det muligt at bruge denne data til medicinsk forskning, der på sigt vil gøre telemedicinske systemer mere nøjagtige, samt gøre forsking mulig på data, der normalt ikke er tilgængelig.

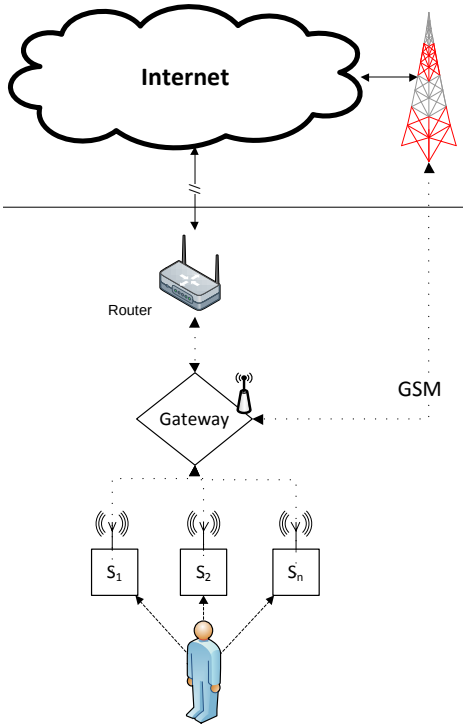
Figur 3.9 side 28, anvender til dels samme arkitektur som figur 3.8, dog er logikken flyttet længere ud i „skyen“.
Sensorerne sender data trådløst til en gateway, hvor disse opfanges og trådløst sendes videre til et accesspoint. Herefter sendes data ud på nettet, gennem en normal trådet forbindelse for at blive analyseret og behandlet. Ved dette skridt bliver den tunge processering lagt ud på nettet, hvilket medfører man derfor ikke længere har brug for en patient PC i hjemmet.
Man kan kritisere løsningen ved, at den er datatung da den sender alle data ud

af huset, hvilket vil stille krav til systemets internet båndbredde. Det kræver endvidere, at patientens gateway er inden for rækkevidde af et åbent WiFi netværk. Dette vil med andre ord medføre at patienten bliver tvunget til at være i hjemmet, hvis det ønskes der skal overvåges kontinuerligt.
Hvis dette ikke er tilfældet kan alle data fra sensorerne lagres på en cache i gatewayen og sendes når dette er muligt. Dette vil dog forhindre systemet i at alarmere i tide, da data kan være forsinket i en periode på ubegrænset tid.



Figur 3.9: *Patient med gateway, hvor gatewayen opsamler data og sender den videre.*

Problematikken ved figur 3.9 side 28 er den manglende garanti for dataforbindelse. Dette ændres på ændret på figur 3.10, hvor der introduceres trådløs GSM kommunikation fra gatewayen. WiFi kan bibeholdes i gatewayen for at gøre det muligt, at sende via WiFi når dette er muligt, og gennem GSM når patienten er uden for WiFi dækning.
Derved sikres kontinuerlig overvågning af data døgnet rundt.



Figur 3.10: *Patient der bærer en gateway hvor sensordata sendes til. Gatewayen har både WiFi og GSM for at kunne sende data til internettet.*

En gateway, der skal stå for at indsamle data fra sensorer, konstrueres på forskelligt vis. På langt sigt vil man foretrække en specialbygget enhed, der lever op regulatoriske krav. Hvis man starter med prototyping, er en ideel løsning en smartphone.

En smartphone har en masse indbyggede sensorer og funktionalitet. På figur 3.11 ses nogle af de features en typisk smartphone har, og de fremtrædende features for en gateway er markeret med fed skrift. Ved at bruge en smartphone opnås ekstra features og funktionalitet „gratis“, mens disse oftest er meget pålidelige og gennemtestet.
Smartphonen vil ikke være ideel rent regulatorisk (jf. afsnit 4.12), men rent prismæssigt er det svært at lave noget billigere end en smartphone, der allerede er i masseproduktion.

Android som er en af topspillerne på smartphonemarkedet har i deres nye version (Android 4.0) valgt at implementere Continua platformen som beskrevet i artiklen[3]. Dette gør det muligt for udviklere nemt, kan lave applikationer der benytter kommunikation med Continua kompatible sensorenheder.

Hvis man overvejer udtrykket „There is an app for that“ som oprindeligt blev introduceret af Apple i sammenhæng med deres iPhone smartphone, giver det god mening at man netop har en app (applikation) for at kunne blive monitoreret.

Samtidig opnås der en fordel ved udrulning og software opdateringer, da de fleste smartphone producenter understøtter et marked, hvorfra man kan hente apps og opdateringer.

Skærm	**GPS**
Bluetooth	Kompas
GSM/3G	Accelerometer
WiFi	**OS + API**
Mikrofon	**Market**
Højtaler	USB stik
Vibrator	LED lys
Kamera	**Cache/storage**

Figur 3.11: *Smartphone, med features. De fremhævede er de væsentlige for projektet.*

Man ville også ud fra de indbyggede sensorer kunne lave intelligente løsninger. Hvis man f.eks sammensætter ideen bag et faldsensorprojektet bygget til Android iFall[37], som bruger telefonens accelorometer til at diagnosticere om pa-

tienten er faldet eller laver noget andet. Faldet er defineret ved en kraftig acceleration efterfulgt af at der ingen bevægelse er. IFall aktiveres, hvis der detekteres et fald og programmet spørger om patientens tilstand.
Programmet tilbyder en faldsensor til ældre mennesker, hvilke ellers risikere at falde og ikke have mulighed for at rejse sig igen.
Vibrator og ringetoner kan bruges som alarmer, GPS position til lokation af patienten, og accelerometer til at se om patienten stadig går rundt eller om personen ligger helt stille. Naturligvis kan telefon ligge på et bord eller lignende, hvilket påviser at man ikke blindt skal stole på sensorerne, men at de imidlertid giver nogle fine retningslinier.

Derudover har man som udvikler også en stor API i Googles Android platform, hvilket gør det nemt at udvikle programmer og udvidelser.

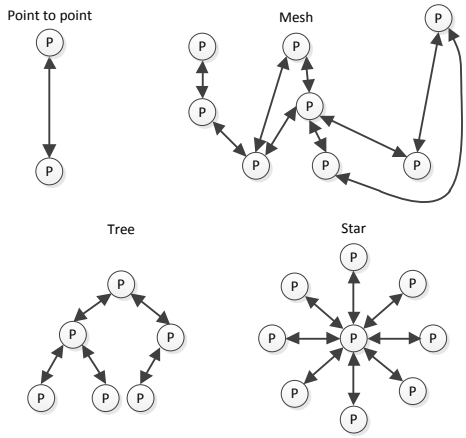
I starten af projektet blev der udviklet en lille nem applikation til en Android telefon, der modtager accelerometer data (hvordan telefonen holdes) disse data pakkes ind i et XML lignende dokument (JSON). Dokumentet indeholder lister af lister, så en datapakke der sendes fra telefonen, indeholder 1000 observationer fra accelerometerets tre akser. Data sendes ud til en server der gemmer data. Dette beviser at lister i lister fra sensorer kan lade sig gøre fra en smartphone.

Kildekoden kan ses i appendix C.

3.1.2 Kommunikation

Der findes flere netværkstopologier, der alle har nogle unikke egenskaber. Figur 3.12 side 32 viser fire kendte netværkstopologier: „Point to point“, „Star“, „Tree“ og „Mesh“. Det der adskiller dem fra hinanden, er hvordan forskellige enheder forbindes.

Et **point to point**, består altid kun af to enheder, der kan kommunikere med hinanden.
Star netværket er det simpleste netværk og bruges ofte mellem computere. Netværket består af et knudepunkt, eksempelvis en netværks switch.
Tree strukturen har et toplevel, som deles ud til en eller flere grene (point to point), dette betyder f.eks at kommunikationen fra niveau ét og tre skal gå gennem niveau to, da dette vil være den eneste kommunikationsvej.
Mesh netværk bruges ofte som selfhealing netværk, hvor noder kan sende til noder for at finde ruten til den akutelle destination. Mesh netværk kan f.eks bruges, hvis mange sensorer spredes ud over en mark, og man ønsker at kunne tilgå alle sensore. Sensorene vil alt efter implementering evt. oprette forbindelse til alle nærliggende sensore på marken, og derved oprette et mesh netværk af



Figur 3.12: *Fire forskellige netværkstopologier, Point to point, Mesh, Tree og Star*

forbindelser, hvorved det sikres at data kan sendes fra A til B. Ud over disse findes der også andre som f.eks „Bus“ og „Ring“ men de fire nævnte er de mest essentielle for projektet.

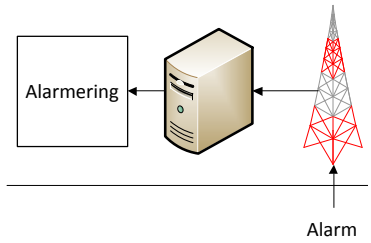
Netværkstopologier kan benyttes på både patientnetværket, samt på servernetværket til analyse af dataene.

3.1.3 Servernetværk

Det er svært at finde litteratur på hvordan andre laver telemediciniske serverarkitekturer, så dette afsnit kommer med nogle kvalificerede gæt af løsninger på serverarkitekturer.
Det der er altafgørende er, størrelse og kompleksiteten af serveren. Som omtalt i kapitel 3 kommer det an på, hvor processeringen placeres.
Hvis man gætter på, hvordan IctalCares serversideløsning ser ud, er den som det ses på figur 3.13 meget simpel. Det eneste der sendes til serveren er alarmering om, at patienten har en alarm. Ud fra denne alarm vil systemet slå patienten op i en database og underette pårørende til patienten.

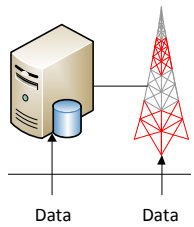
Denne løsning har en minimal serverside løsning og bruger nærmest ingen datatrafik.

Andre løsninger som f.eks bruges til forskning er, at patienter bærer en sensor. Denne sensor vil sende data ud på nettet, hvor de gemmes. Senere kan læger



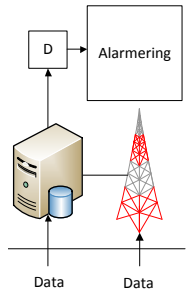
Figur 3.13: *Et gæt på hvordan IctalCares serverløsninger ser ud*

og forskere kigge på sensorens rapporterede værdier, og ud fra disse evt. lave en diagnosticering.



Figur 3.14: *Serveren modtager data på forskellige kanaler og gemmer disse til eventuel forskning.*

En serverside løsning ses på figur 3.14, men hvordan den implementeres er op til opgaven, og udover dette kan data også eksporteres til forskellige interessenter.



Figur 3.15: *Serveren modtager data, og analyserer denne, og ud fra analysens diagnosticering kan der alarmeres*

Denne løsning baseres på ideen fra figur 3.14, hvor data sendes til den eksterne server, men på figur 3.15 bliver data nu analyseret af matematiske algoritmer. Dette betyder i praksis, at patienter ville kunne få varsling på deres vitale værdier, selv uden at en læge overvåger dem. Dette kan sidestilles med at få foretaget

en sammenlignelig overvågning, som normalt på et hospital, men mens man forbliver i hjemmet og fortsætter sin dagligdag.

3.2 Opsummering

Det er teoretisk muligt at lave en telemedicinsk arkitektur, der kan bruges på generelle problemstillinger. IctalCares løsning binder sig meget op af et lignende system, dog med den undtagelse, at der kun sendes alarmer og ikke rå data ud på nettet. Den løsning er dog også fin, hvis det antages, at der er fremstillet en korrekt generel analyse af en sensors værdier.
Det kan dog være svært at bevise at man får patienters forskelligheder med i en generel diagnosticering, samt bevise, at en løsning er 100% korrekt. Derfor anses en løsning, hvor data sendes på nettet for at blive analyseret som optimal. Tæt på patienten kan sensorerne enten selv have mulighed for at sende data, eller ved hjælp af en gateway.
Gatewayen kan have nogle regulatoriske udfordringer, og dette bør undersøges nærmere.

Kapitel 4

Analyse

Analysen bygger videre på teorien i kapitel 2, Der ses nærmere på, hvordan der bedst muligt laves et WBAN og en server arkitektur til et telemedicinsk system, der skal kunne diagnosticere og alarmere.
Afsnittet introducerer hvilke funktionelle krav og problemer, som der kan være for et telemedicinsk system, samt nogle af de problemstillinger der er til et sådant.
Der beskrives kort, hvordan en databehandling kan opstilles og det vises grafisk med f.eks liniær fremskrivning.
Der begrundes, hvilke fordele der vil være ved at have dem med i systemet, hvad fagrupperne ønsker at kunne tilgå, samt hvorfor der skal være et normaliseringslag.
Der beskrives kort en evt. XML struktur, der kan benyttes i et system til netop dette, samt beregnes, hvor megen data der vil blive sendt fra en patient, hvis der benyttes XML. Derudover undersøges hvilke regulatoriske krav der skal opfyldes for at kunne realisere projektet.
Sidst i kapitlet i afsnit 4.13 side 73, beskrives hvordan der kan dannes et system ud fra analysens øvrige afsnit.

4.1 Funktionelle krav

For at kunne danne et system, er det nødvendigt at opstille nogle funktionelle krav til systemet. Disse er krav som systemet som mimimum burde kunne overholde.

For dette projekt er der opstillet følgende funktionelle krav, som projektet skal dække over. Disse krav er baseret på egen viden og behov, og ikke specificeret i samarbejde med sensorproducenter eller læger.

- Alarmere/reagere inden for en tidsramme på 5 min.
- Kunne lave diagnoser baseret på flere sensoreres data.
- Altid være i stand til at sende data til server.
- Lokal alarmering, hvis data ikke kan sendes.
- Kunne håndtere minimum otte sensorer pr. patient.
- Historik over data (Database).
- Dataforbindelse mellem server og patient skal være sikker.
- Server skal kunne håndtere minimum 500 patienter, og gerne op 5 mio. patienter.
- Systemet skal kunne differenciere mellem forskellige patientgrupper og hvor højt prioteret de er.
- Feedback fra brugere, nye analyser kræver gemt data.

Kravene dækker primært over systemets konnektivitet, samt at systemet skal være responsiv for at sikre hurtig alarmering af patienter.
Derudover er der nogle krav, som dækker over dataindsamling. Dataindsamlingen er netop vigtig på baggrund af at forskere ønsker så megen data som muligt til f.eks sygdomslære. Derudover er data også vigtig til senere, træning af neurale netværk.
Netop neurale netværk der skal trænes på baggrund af data, har behov for at kunne udtrække alarmeringer og evt. fejlalarmering. Fejlalarmeringer kan patienten selv melde tilbage om.

Disse krav bringer tankerne direkte videre til nogle af de problemstillinger projektet har.

4.2 Problemstillinger

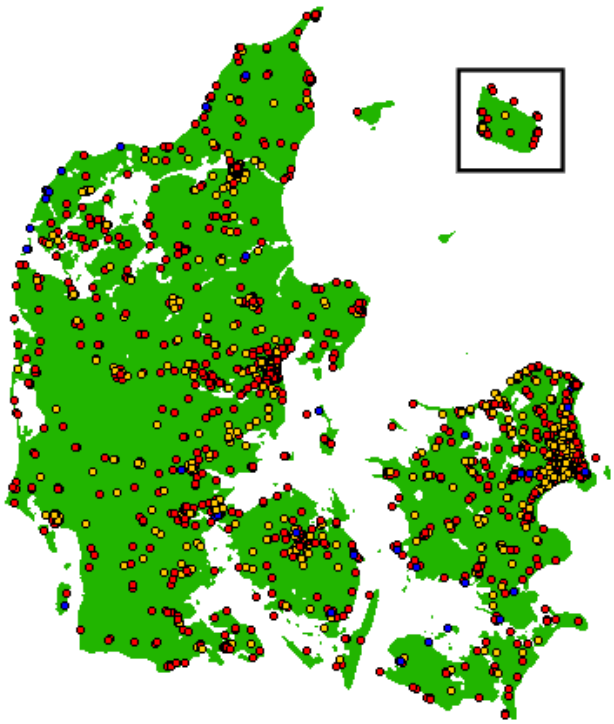
Der vil altid være problemer ved enhver løsning og til hver løsning kan der være nye problemer. Af samme grund belyser dette afsnit kort nogle af de elementer, der kan være problematiske for en funktionsdygtig implementering af projektet. Problemerne kan være større eller mindre, men vil alle have en afgørende betydning for, hvad der endeligt vil benyttes.

4.2.1 Gatewayhastighed

En af problemstillingerne, der kan opstå i systemet er, at gatewayhastigheden på patientsiden er for langsom.
Hvis der f.eks er 8 sensorer, der hver sender 4KB/s er der brug for 32KB/s. Hvis gatewayens datahastighed er 256Kbit/s svarende til 32KB/s er der risiko for at gatewayen er overbelastet og derved bliver gatewayen ude af stand til at sende hurtigt nok eller kan miste data. I begge scenarier kan være katastrofalt for systemets virke.

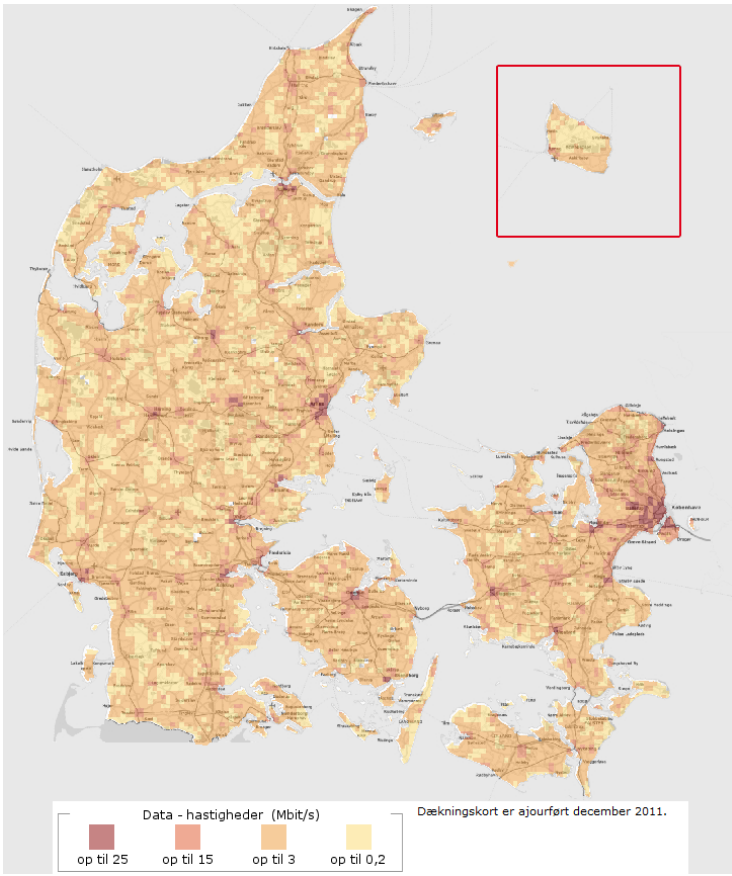
4.2.2 WiFi dækning

Netop konnektivitet kan være svært at overholde hvis, man ønsker 100% dækning indenfor de danske grænser.
Hvis det telemedicinske system skal kunne sikre, at man inden for få minutter kan diagnosticere patienten, skal data sendes kontinuerligt til serveren, der skal stå for analyse heraf.
Hvis det vælges at enheden kun skal have WiFi kan dette lade sig gøre hvis: personen er hjemme eller indenfor rækkevidde af et offentligt åbent WiFi. Denne mulighed er i følge figur 4.1 ikke særlig udbredt, da der kun er ialt 2271 åbne HotSpots.
Ved brug af WiFi skal det samtidig sikres, at enheden opnår en reel forbindelse til nettet, samt at den har mulighed for at skifte til GSM såfremt dataene ikke skal overføres.



Figur 4.1: *Åbne WiFi ifølge openwifi.dk[32], prikkerne viser hvor i Danmark der findes åbne WiFi placeringer.*

Da der ikke er god dækning i Danmark på åbne WiFi, anses dette for, at det er en nødvendighed med en anden metode, der kan sendes gennem. Her anses GSM som en god kandidat, da teknologien er forholdsvis gammel, samt Danmark allerede nu har et stabilt og meget dækkende datanet, Normalt estimeres det til at dække 98% af Danmark og de resterende 2% er ofte øerne eller udkantsdanmark. Dette hul i dækningen kan give problemer når figur1.4 viser at netop ældre flytter til udkantsdanmark. Ulempen er at det er omkostningstungt at sende data over mobilnettet. Dæknigen hos TDC for GSM er bedre end med åbne WiFi (se figur 4.2).
Af disse årsager anses det at sensor, gateway, eller anden slags mellemled har behov for at kunne have en kombination af WiFi og GSM eller GSM alene, da WiFi ikke lever op til kravet om, at data altid kan sendes til server.



Figur 4.2: *GSM dækning af TDC (lettere modificeret for at passe ind i rapportens layout)*

En ren GSM løsning kan dog være dyr på sigt, idet der sendes en del data (hvilket også ses i afsnit 4.6). Det ville evt. kunne løses ved at; anvende WiFi når dette er tilgængeligt, mens der i øvrige situationer benyttes GSM.. En illustration over et såsdan system ses på figur 3.10 side 29.
Ideen bag åbne WiFI kan sagtens bruges, hvis dataprotokolen ikke indeholder tegn på hvem patienten er. Åbne WiFi netværk kan være usikre og ofte uden kryptering og derfor risikere patienten at visse datapakker med vitale værdier, opfanges af en person denne data ikke vedrører.
Patienter på arbejdsmarkedet vil også bruge meget tid på arbejdet, hvilket vil sætte krav til arbejdsgiverens WiFi netværk.

4.3 Valg af opdeling

Hvor man placerer de forskellige elementer og procedurer i systemet, kan ændre systemet radikalt og samtidig få systemets pris til at stige markant. En anden detalje kan være om systemet tillader at nye algoritmer implementeres på gemt data eller om den allerede analyserede data smides ud.
Der indgår også vurderinger om batteriforbrug, samt hvad en sensor vil koste.

Der diskueres tre emner; hvordan IctalCare har lavet deres løsning, derudover et gateway- og til sidst analyseres en løsning, hvor man har intelligente sensorer, som indeholder datakommunikation og kendskab til eksterne servere.

Afsnittene kan sammenlignes ved figurer/lister i hvert afsnit, der beskriver hvad der generelt foregår i de respektive dele af systemet. Det som systemet ikke udfører i den givne arkitektur, er anført i grå og med overstregning. Denne visning er for nemmere at sammenligne figurene 4.3, 4.4 og 4.5, hvorved det tydeliggøres hvilke elementer der findes på tværs af systemerne.

4.3.1 IctalCare

IctalCares løsning som beskrevet i afsnit 2.2, side 12, kan anses som moduler. Hvis man ser på figur 4.3 skal man forestille sig at dette er hele systemet med sensor, gateway og „skyen“ (hvad der sker på serveren)

Sensor
Batteri
~~WiFi~~
~~GSM~~
Kommunikation (Gateway)
Indikatorer
~~nPoct (kendskab)~~
Alarmering
Intelligens
Cache
~~Processering (Lille)~~
Processering (Tung)
~~GPS~~
~~CPU (Lille)~~
CPU (Kraftig)

Gateway
Batteri
~~WiFi~~
GSM
~~Kommunikation (Sensor)~~
Indikatorer
~~nPoct (kendskab)~~
Alarmering
~~Intelligens~~
~~Cache~~
~~Processering (Lille)~~
GPS
CPU (Lille)
~~Højtaler~~
~~Microfon~~

Skyen
~~Processering~~
Alarmering
~~Intelligens~~

Figur 4.3: *IctalCare logik*

Kritik af IctalCares arkitektur:

- Ingen datalogning (ingen mulighed for at forbedre algoritmer).
- Begrænset logik på serversiden.
- Sensor bruger meget strøm.
- Sensoren er formentlig dyr da den er specialfremstillet.
- Understøtter ikke flere sensorer.

IctalCares fordele:

- Lav datamængde (der sendes kun alarmer)
- Mikrofon og højtaler i gateway

IctalCare har flere elementer, der er opbygget hensigtsmæssigt, dog synes begrænsningen i at data ikke sendes ud på nettet som et kæmpe mangel.
Hvis dette er gjort for at minimere datatrafik ressourcer, kunne det overvejes at når der detekteres et anfald, kunne den sidste halve time af data sendes ud på nettet, for måske på sigt at lave bedre algoritmer der kunne advare patienten før et anfald.
Ellers virker IctalCare opbygningen som en gateway løsning, dog kun med én sensor og uden at sende data.

4.3.2 Gateway

Hvis den omtalte gateway funktionalitet analyseres, ser disse placeringer helt anderledes ud. Figur 4.4 viser, at sensoreren bliver relativ uintelligent og derfor også langt billigere at producere; data sendes til gatewayen og der er kun envejskommunikation til gatewayen. Dette betyder også, at man sikrer sensoren for evt. regulatoriske høje klassificeringer, da gatewayen nu vil kunne anses som et kommunkationsled og derfor ikke som et medicinsk udstyr. (se afsnit 4.12).

Sensor
Batteri
~~WiFi~~
~~GSM~~
Kommunikation (Gateway)
~~Indikatorer~~
~~nPoct (kendskab)~~
~~Alarmering~~
~~Intelligens~~
~~Cache~~
~~Processering (Lille)~~
~~Processering (Tung)~~
~~GPS~~
CPU (Lille)
~~CPU (Kraftig)~~

Gateway
Batteri
WiFi
GSM
Kommunikation (Sensor)
Indikatorer
nPoct (kendskab)
~~Alarmering~~
Intelligens
Cache
Processering (Lille)
GPS
CPU (Lille)
Højtaler
Microfon

Skyen
Processering
Alarmering
Intelligens

Figur 4.4: *Gateway løsning hvor sensoren er meget begrænset og logikken er placeret på gatewayen.*

Anvedelsen af billige sensorer kræver dog også at man har en forholdsvis dyr gateway, som har en masse funktionalitet. Kan man få gatewayen til at være en billig smartphone, vil prisen for en af de billigste Android mobiler være ca. 1500 DKK (i 2011).
Det kan diskuteres om denne løsning er at foretrække, hvis man kun har én sensor tilknyttet til patienten. Måske er det billigere at benytte en dyrere sensor, der har flere features indbygget; dette bringer os til næste punkt.

Kritik:

- Gatewayen kan være dyr at producere.
- Hvis der kun er én sensor er det unødvendigt med en gateway løsning.

Fordele:

- Centraliseret data transmission.
- Kan benytte billigere sensorer.
- Gatewayen kan hvis sensorne sender batteriniveau, alarmere på dette, og bede patient skifte batteri en sensoren.
- Kendskab til serveren skal kun placeres på gateway og ikke sensorer.
- Ingen krav til sensorer om kendskab til den server benyttede protokol.
- Kravet til sensorer bliver mindre, eksempelvis er det ikke er nødvendigt med display, lysdioder osv, da gatawayen kan håndtere dette.
- Mulighed for flere sensorer pr. patient.
- Kan være en fordel regulatorisk (se afsnit 4.12 side 71).

En gateway løsning udmærker sig på mange punkter både i pris og regulatorisk. Samtidig sendes dataene ud på nettet, hvilket muliggøre personalisering af diagnoticeringerne. Endvidere vil det være muligt at udvikle bedre diagnosticeringsalgoritmer, på baggrund af de udsendte data.
Gatewayen kan i sig selv være dyr, især hvis man betragter et WBAN med kun én sensor og en gateway, men nærmest ekspotentiel billigere for hver ekstra sensor.

4.3.3 Smarte sensorer

Der kan ogsåDer kan også udvikles smarte sensorer, som indeholder alt den nødvendige funktionalitet til at kunne sende data ud på nettet gennem både WiFi og GSM. Dette vil imidlertid stille større krav til batteriet i sensoren, da strømforbruget derved øges markant.
Figur 4.5, viser at hver sensor næsten får alt funktionalitet som en gateway fra forrige afsnit, hvilket vil gøre den pågældende sensor dyr.

Sensor
Batteri
WiFi
GSM
~~Kommunikation (Gateway)~~
Indikatorer
nPoct (kendskab)
~~Alarmering~~
Intelligens
Cache
Processering (Lille)
~~Processering (Tung)~~
GPS
CPU (Lille)
~~CPU (Kraftig)~~

Gateway
~~Batteri~~
~~WiFi~~
~~GSM~~
~~Kommunikation (Sensor)~~
~~Indikatorer~~
~~nPoct (kendskab)~~
~~Alarmering~~
~~Intelligens~~
~~Cache~~
~~Processering (Lille)~~
~~GPS~~
~~CPU (Lille)~~
~~Højtaler~~
~~Microfon~~

Skyen
Processering
Alarmering
Intelligens

Figur 4.5: *Intelligente sensorers placering, de har mange ting indbyggede hvilket gør dem dyre men også istand til at fungere uden brug af gateway.*

Sensoren vil være en god løsning til den stationære sensor, som har en fastnets strømkilde og er fast placeret i hjemmet, dette kan f.eks være til monitorering af rum temperaturer, patientens dags vægt eller lignende.

Ud over dette skal sensoren også have kendskab til, hvor data skal sendes hen, samt hvordan data skal sendes. Dette vil kræve at sensoren kan opdateres hvis der kommer ændringer, hvorved sensoren skal opnå en højere medicinsk klassificering, hvilket beskrives nærmere i afsnit 4.12.

Kritik:

- Sensorer skal have kendskab til server.
- Mulighed for firmware opdatering.
- Regulatoriske udfordringer.
- Højt batteriforbrug.
- Kræver sensorer med indikatorer for at vise batteriniveau.
- Special produktion af en sensor, der netop understøtter den ønskede protokol.

Fordele:

- Selvstændige sensorer.
- Ingen behov for gateway.

Smarte sensorer, er generelt en dyr løsning, idet det vil kræve specielt udviklede sensorer, da der er brug for kendskab til datakommunikation, protokoler osv. Af disse årsager anses denne løsning ikke som optimal.

4.3.4 Opsummering

Hvis man skal finde frem til hvad der er bedst generelt, er dette stort set umuligt, da det afhænger af, hvilke behov der skal dækkes og hvad formålet med overvågnigen er.
Det optimale for et telemedicinsk system, der på sigt skal bidrage med informationssamling, er naturligvis at processeringen samt datamodtagelse finder sted på en server.
Om man benytter smarte sensorer eller en gateway løsning, er sværere at fastlægge. Hvis man kun benytter én sensor virker det voldsomt, at man også skal have en gateway, da gatewayfunktionaliteten kunne have været placeret i sensoren. Samtidig kan gatewayen understøtte flere forskellige sensorer, hvorved dens anvendelsesmuligheder kan udvides løbende. „Det anses derfor at gateway løsningen kunne anvendes som en standard platform for de telemedicinske systemer.“
Der er alligevel faktorer, der peger på at gateway løsningen er optimal.
Disse er f.eks de regulatoriske fordele ved at have adskilt tovejskommunikationen, men også at sensorerne ikke behøver være specialfremstillede til formålet. Specialfremstillede produkter vil alligevel altid være en dyrere løsning, frem for en „hylde“ sensor, som kan bruges til meget andet end et telemedicinsk system, og patienter kan jo risikere at tabe en sensor i toilettet eller opstå andre uheld. Så med en billigere sensor ville det ikke være et stort tab, ved uheldige episoder tilført sensoren.

Dette bringer tankegangen videre til, hvordan en gateway/WBAN løsning kan opsættes.

4.4 Patientnetværk

Netværket på patienten som man kan kalde et „wireless body area network“ (WBAN) eller et „personal area network“ (PAN), kan oprettes i mere eller mindre omfang, alt efter behov. Netop et WBAN findes der en del videnskabelige artikler om[33]. Artiklerne konkludere altid at et WBAN er løsningen, for et netværk tæt på patienten, og foreslår altid en gateway løsning, for at sikre stabilitet, strømforbrug og pris.
Hvis der sigtes efter en gateway løsning, skal enhederne/sensorerne kommunikere med gatewayen, hvilket betinger nogle faktorer.
Rækkevidden kan diskuteres, alt efter hvad der benyttes som gateway:
Hvis gatewayen er en mobil enhed man har med sig, på samme måde som et ur, og resten af sensorerne er på kroppen, vil en trådløs rækkevidde på to meter være tilstrækkelig.
En løsning, hvor rækkevidden er større og kan dække et hus ville være fordelagtigt, da man kan placere gatewayen et sted i huset og først når man forlader sit hjem, skal man bære den med sig. I dette tilfælde er en rækkevidde på 30-100 meter ideelt. Grunden til at der foreslås 100 meter er at f.eks armerede beton vægge vil mindske rækkevidden væsentligt.

For ikke at risikere, at to patienter begge med en gateway, begynder at modtage hinandens data, vil en form for parring mellem gateway og sensor være at foretrække.
Da man med paring kan sikre, at data ikke begynder at sendes til et andet knudepunkt blot fordi det er i nærheden.

For at få det bedst mulige overblik over de nuværende trådløse teknologier er der lavet en undersøgelse af nogle af de nuværende radioteknolgier, der er på markedet.
Der kigges på de faktorer der menes at være vigtige for et WBAN af sensorer. De vigtigste for et telemedicinsk overvågningssystem vil være batteriforbrug, rækkevidde, hastighed, sikkerhed samt parring af sensorer.

- **Z-Wave**
  Z-Wave er en trådløs kommunikationsprotokol dannet i samarbejde mellem danske Zensys og Z-Wave alliancen, Z-Wave er lavet med fokus på lavt batteriforbrug samt en lav båndbredde. Z-Wave bruges ofte til fjernbetjeninger til lys, kaffemaskiner og andet hjemmeelektronik.

- **ZigBee**
  ZigBee er en billig trådløs kommunikationsprotokol med fokus på pris, lavt batteriforbrug samt sikkerhed. ZigBee understøtter både 'Mesh', 'Star' samt 'Tree' topologierne i netværk. ZigBee bruges generelt i monitorerings udstyr og er på dette en konkurrent til Z-Wave. ZigBee har ydermere en afdeling med fokus på telemedicin.

- **Bluetooth**
  De fleste kender Bluetooth (BT), som blev udviklet som et alternativ til RS-232 (com port). BT findes i flere klasser, som får specifikationerne til at variere på hastighed og rækkevidde.
  Generelt ses BT som en teknologi med bedre hastighed og højere strømforbrug end andre lignende teknologier (ZigBee).
  BT bruges blandt andet til headsets, mus, keyboard og andet PC-udstyr.

- **Bluetooth LE**
  Bluetooth Low Power eller Bluetooth LE (BLE) er en videreudvikling af Bluetooth, og en del af Bluetooth 4.0.
  Forskellen fra BLE til BT er, at BLE sender ved en langt lavere hastighed, men strømforbruget er også formindsket markant. Derudover er opstarts tiden minimeret, hvilket vil gøre at man ofte ville kunne slukke helt for sit BLE hvor et BT apperat tager flere sekunder før den kan parres efter opstart. BLE er relativt nyt (2009), men forventes at blive brugt i telemedicinskeapperater såsom sensorer samt PC-tilbehør, der ikke kræver store datamængder.

- **Wireless Fidelity (WiFi)**
  WiFi, kræver ikke den store præsentation. WiFi kendes primært fra PC og mobiltelefonverdenen, og med WiFi kan der opnåes store rækkevidder, men det kræver også meget strøm.

- **Near field communication**
  Near field communicatoin (NFC) bygger videre på principperne bag RFID, NFC bruger primært til datatransmission fra telefon til telefon, kreditkort til betalingstandere, eller som betalingskort til f.eks busser.
  Den korte rækkevidde gør systemet yderst sikkert.
  NFC anses for at have en forholdsvis høj datatransmissionshastighed.

- **Propreritære**
  Der findes et utal af forskellige propreritære radioløsninger, hvor producenter laver deres egne, indenfor de åbne båndbredder. Disse løsninger har ikke nogen fælles betegnelse og kan ikke vægtes på nogen måde. De kan ligge meget tæt på nogle af de ovenstående standarder, men har spcialiseret protokollen til at kunne noget helt specifikt. Dette er en løsning der kan overvejes i et senere stadie af projektet, men set til prototype udvikling er dette ikke en mulighed.

- **Software radio**
  Dette er, hvor både radio frekvens og protokol kan skiftes i løbet af kort tid. Der benyttes ofte FPGA programmering til software radio.
  Men at se disse indbygget i relativ lille sensor lyder dog urimeligt, ud fra de løsninger der findes pt. Dog bør software radio overvejes til gatewayen. Ved brug af software radio bør gatewayen via software kunne kommunikere de protokoller der ønskes af forskellige sensor fabrikanter. Dermed behøver man ikke være afhængig af en teknologi.

For at danne et overblik over relevante features i teknologier er de vigtigste opstillet i nedenstående tabel 4.1 side 49. Dog har det været umuligt at finde et device der har indeholdt alle faktorerne, så tabellens data er opbygget over flere devices og kan derfor ikke anses for 100% generelle.

Ud fra tabel 4.1 og behovet i projektet ses det, at ZigBee og Bluetooth LE er de to konkurrenter på markedet der kan dække behovene. De resterende bruger generelt for meget strøm eller har for kort en rækkevidde. Rækkevidderne der er opstillet i tabellen er teoretiske rækkevidder og ikke en faktisk måling, hvor landskab og genstande kan ændre rækkevidden markant.

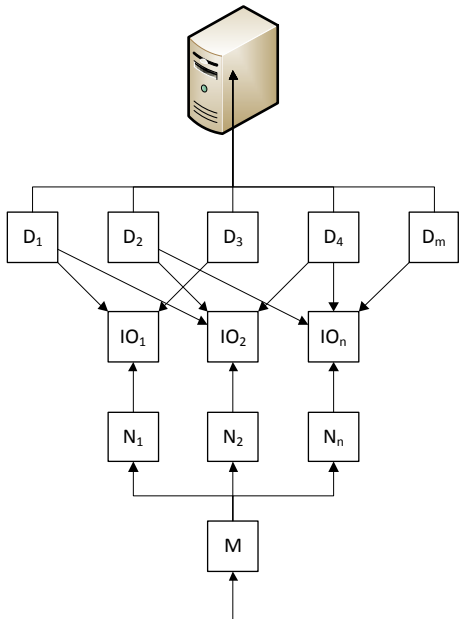
Udover et WBAN er der også brug for at få serversidens arkitektur på plads, da det er denne, der skal analysere og diagnosticere.

Teknologi	Kryptering	Hastighed	Rækkevidde	Strømforbrug	Båndbredde	Parring	Netværkstype
Z-Wave [44], [45]	3DES	9600 b/s	30 m	N/A	2.4 GHz	N/A	Mesh
ZigBee[27]	AES	250 kb/s	50 m	15 mA	2.4 GHz	Ja	All
Bluetooth (c. 2)[21], [25]	SAFER+	3 Mb/s	$<$ 10 m	2.5 mW	2.4GHz	Ja	Scattermet
Bluetooth LE [11]	AES 128	1 Mb/s	20-50 m	$<$15 mA	2.4 GHz	Ja	Star
WiFi 802.11n [1],[39]	WPA/WPA2	$<$ 600 Mb/s	$<$ 70 m	18 W	2.4 & 5 GHz	Ja	Star
NFC [34], [31]	Software	424 kb/s	20 cm	$<$ 15 mA	t 13.56 MHz	Nej	Point to point

Tabel 4.1: *Viser forskellige radio teknologier, Bluetooth LE og ZigBee ser ud til at være ideele til medicinsk udstyr, rækkevidden er teoretisk, og de steder der står N/A har det ikke været muligt at finde et resultat for dette.*

4.5 Servernetværk

Serversidens analyse og opbygning kan variere ud fra flere aspekter. På figur 4.6 er arkitekturen designet til at have en enkelt modtager. Denne modtager kan få en datapakke, der indeholder op til flere rådatamålinger fra flere apparater. Modtageren vil dele disse pakker op i segmenter ud fra hvilket type apparat data kommer fra og sendes videre til yderligere processering.



Figur 4.6: *Serverside*

Figuren viser et system, hvor patienten har tre sensorer, og derfor tre normaliseringsprocesser efter modtagelsen. Normaliseringen sker for at simplificere den senere dataanalyse. Dette er primært nødvendigt for at kunne udarbejde en ensartet analyse ud fra forskellige sensorer og apparater. Der ses nærmere på normaliseringen i afsnit 4.7 side 59.

IO laget er der, hvor den givne data gemmes/hentes og videreudbydes som resource ud fra hvilken apparattype, der sendes data fra, f.eks (blodsukkermåling). Diagnosticeringlaget er, hvor den egentlige analyse bliver udarbejdet. Figur 4.6 skal forståes sådan at en diagnosticering kan benytte data fra en eller flere målinger, f.eks kunne et sygdomsmønster både være afhængigt af data fra en temperatur- samt blodsukkermåling, da hverken temperatur- eller blodsuk-

kermålingen adskilt giver et entydigt resultat på en specifik sygdom. Af samme årsag skal det være muligt at kunne oprette et antal forskellige kombinationer af diagnosticeringsundersøgelser. Dette antal er givet ved:

$$\sum_{r=1}^{n} \left(\frac{n!}{r!(n-r)!} \right)$$

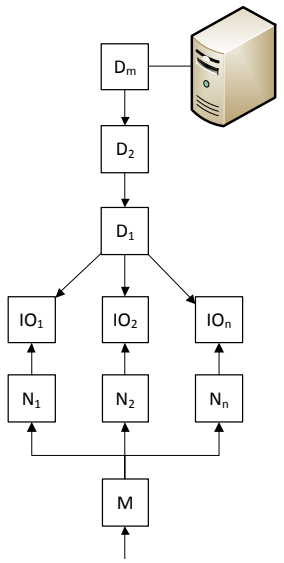
(n = antallet af sensorer) Dette antal er dog ikke nok, da man også ville kunne udvide med at have flere forskellige algoritmer inden for hver diagnosticeringsundersøgelse og bør betrages som om der findes uendeligt mange analysemetoder.

Disse diagnosticeringsprocesser skal kunne sætte et „flag“, der indikerer at de ønsker at alarmere.
Alarmeringen er nærmere omtalt i afsnit 4.10.

Hvordan, og i hvilket flow, diagnosticeringen skal laves er implementations specifikt. Men der er flere måder at tilgå problemet på.

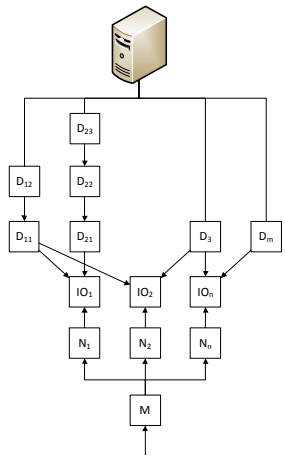
En simpel tilgang er at diagnosticeringen på en patient forløber serielt som vist på figur 4.7. Diagnosealgoritmerne får alt den patient pågældende data udbyder, og analyserer på de faktorer analysen har behov for. Når den er færdig med sin analyse videregiver den data til næste diagnosticering, og hvis den har en alarm, vil den også videregive denne.

Ideen bag dette kan være at spare på CPU ressourcer, men der vil være megen dataoverhead da diagnosticeringerne modtager alt data fra patientens sensorer, også selvom de ikke bruger data til noget. Derudover kan dette skabe problemer med svartider; hvis blot én diagnosticeringsproces tager lang tid forsinkes alle de andre. Forsinkelser kan være fatalt for svartider og derved forsinke en evt. alarmering på problemer.



Figur 4.7: *Seriel processering af data. Dette gør dog at alle diagnosticeringer modtager alt data, hvilket er overflødigt.*

En anden tilgang ses på figur 4.8, som bygger på et „Early out“ princip. Early out principperne bliver f.eks brugt i billedeanalyse (ansigts genkendelse) hvor man afgør om der findes et ansigt eller øjne i billedet, hvis dette ikke er tilfældet er der ikke nogen grund til at lave analyse på billedet og man kan gå videre til næste billede og derved spare beregnings ressourcer.
Dataanalysen fungerer stadig parallelt indenfor samme undersøgelse, men kun hvisde benytter samme vitale værdier som faktorer.
Med andre ord; hvis en patient har to algoritmer til diabetes, vil disse køres sekventielt. Dette kan gøres således men også udvides med at data sorteres fra. Hvis den første diagnosticering laver en grovsortering på data, og finder at værdierne ligger helt perfekt, behøves data ikke at sendes videre, da det er bevist at data ligger som det skal.
Dette giver umiddelbart god mening, men kan igen have uforudsete konsekvenser. Første problem er, hvordan og hvem kan afgøre om en diagnosticering er så effektiv, at den kan afgøre med sikkerhed om data er sikker eller ej?
Er det en læge eller forsker, der skal afgøre rækkefølgen?
Sidst men ikke mindst: Vil f.eks neurale netværk lide af starvation hvis de ikke modtager alle data. Ved sortering i data risikeres der, at intet kan afgøres. Netop denne problematik belyses i afsnittet om „Secure Linking of Patients and Tele-medical Sensors“ afsnit 2.3.

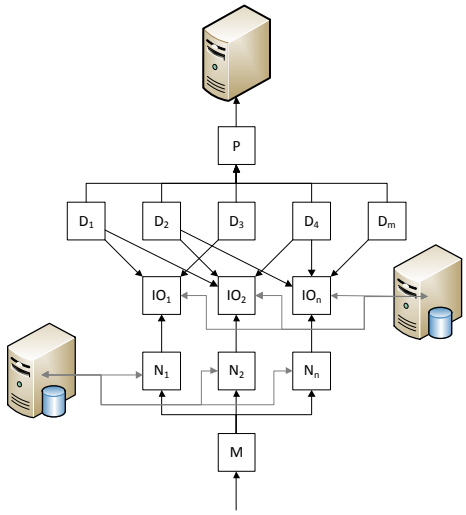


Figur 4.8: *Early out, hvor data ikke sendes videre, hvis den første algoritme ikke mener det er specielt.*

Den bedste løsning for en effektiv analyse samt korrekt data opsamling anses at være hvor diagnosticeringen kører 100% parallelt, og data gemmes i IO laget, hvor data allerede er normaliseret. Data gemmes normaliseret og som rå sensordata. Data skal indeholde så meget information som muligt, når det gemmes, så alt fra sensortype, serienummer og til hvilken normaliseringsalgoritme, der er blevet brugt. Figur 4.9 viser, hvordan serverlaget skal opbygges.
Ved at diagnosticeringen kører parallelt er der ikke nogen læge eller forsker der skal tage højde for, hvilke rækkefølge algoritmer skal køres, samt at der ikke er problemer med turnaround tider (svartider).

I patientlaget vil man kunne placere en mistænksomhedsvektor, så der ikke alarmeres ved den mindste lille detalje. Man ville derfor også kunne give diagnosticeringsalgoritmerne forskellige vægte alt efter hvad de analyserer samt hvor sikre de er på deres alarmering.

Som det ses er der mange rubrikker i arkitekturen, der skal kommunikere sammen, og dette kræver en fælles protokol for at kunne forløbe på bedst mulig vis.



Figur 4.9: *Forslag til serversiden med data logning samt normalisering*

4.6 Protokol

Der skal kommunikeres gennem et fastsat protokol, og der skal altid sendes information om hvem ejeren af data er, så der ikke opstår scenarier, hvor data fra forskellige patienter blandes, som omtalt i afsnit 2.3 side 16.

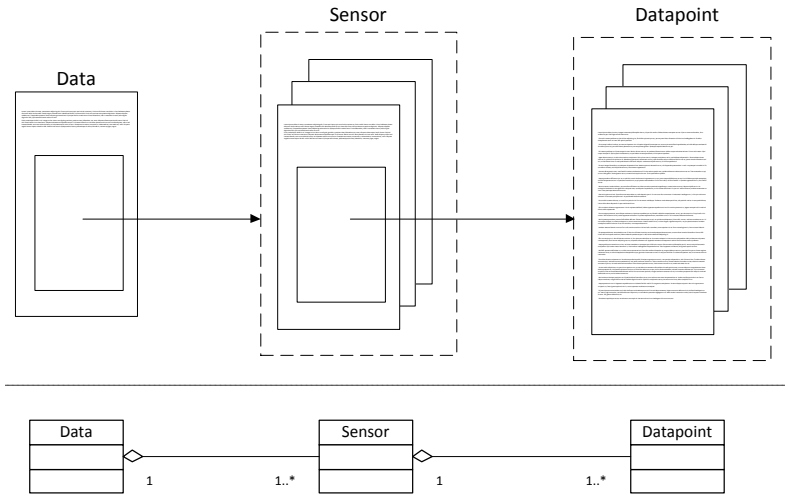
Ved undersøgelse af forskellige sensortyper ses det at ikke alle sensorernes data kan opsættes simpelt på punktform. Nogle kan have flere faktorer end et enkelt punkt i en pakke, og derfor skal der benyttes forskellige slags datapakker til sensorernes data.

For at kunne sikre rækkefølgen af data, samt at data når frem til modtageren benyttes TCP/IPs indbyggede funktionalitet. Hvis nogle pakker sendes over Wi-Fi og andre over GSM, vil TCP/IP imidlertid ikke være nok.
Hver gateway skal derfor kunne indentificeres sammen med et sekvensnummer. Derfor indføres det, at hver gateway har et unikt id (evt. IMEI nummer for en smartphone), og i sammenkobling med et sekvensnummer på pakkerne, er det muligt på serverside at sortere pakker i rigtig rækkefølge, så diagnosticeringen ikke laver uventede fejl, grundet dette fejlscenarie.

Der er fundet nogle elementer som er vigtige for datapakken.

- Heartbeat
- Timestart
- Timeend
- List of sensors
- UserId
- GatewayID (Serie nummer evt)
- Sekvensnummer på sendt pakke
- Package number
- Channel (WiFi GSM)
- Protocol nummer
- En til flere datapakker.

Heartbeat er ikke del af en datapakke, men bør dog understøttes af protokollen for derved at sikre kommunikation mellem gateway og server.
Datatider for, hvornår data er målt, eller modtaget på gateway, er også vigtige, da nogle algoritmer kan forvente data inden for et vist tidsinterval. Hvis andre algoritmer ønsker samme data sjældnere, må den pågældende algoritme selv tage højde for, hvad der skal gøres, hvis der er for mange eller få datapunkter.
Samtidig kan tidspunkterne bruges til at rejse en mistanke, hvis data ikke kommer ofte nok ind. Dette kan være et resultat på at en patient befinder sig i et område, hvor der ikke er netværksdækning eller gatewayen er løbet tør for strøm.
Dog skal man huske at tid i distribuerede systemer er problematisk, men så længe der ikke arbejdes i sekunder og deadlines ikke er afhængige af samtidighed anses dette ikke for et problem. Det bør i et implementeret system overvejes om der skal indføres en form synkronisering.



Figur 4.10: *En datapakke indeholder lister af sensorer, hvilke igen indeholder lister af datapoints*

Det vigtige i pakken er, at den skal kunne indeholde flere sensoreres data i en pakke samtidig med, at hver sensor kan have flere punkter. Med andre ord ønskes det, at der kan sendes lister af lister, og fletningen af pakker er vist i figur 4.10.

Samtidig er det vigtigt at forstå at denne dataopbygning ikke nødvendigvis behøver at være det yderste lag af datapakken, men kan evt. pakke data ind i andre dataformater.
Udadtil behøves der ikke tages højde for error checking af pakken, da dette er standard i TCP/IP, samt er der allerede er nævnt, at datapakkerne vil have et sekvensnummer.

Der er i XML dannet et data element for at symbolisere, hvordan dataen fra gatewayen kan sendes.
Først ses data elementet. Dette element er den yderste indpakning, der derfor ikke indeholder særligt meget ud over lister af sensorer.

```
<?xml version="1.0"?>
<data>
  <dataPackageVersion>1.5</dataPackageVersion>
  <patientID>1337</patientID>
  <gatewayID>abc43-5614</gatewayID>
  <dataPackageSequenceNumer>5678</dataPackageSequenceNumer> <!--
      Auto increment -->
  <timestampSend>2007-11-09 T 11:20</timestampSend>
  <sequenceNumber>53</sequenceNumber>
  <sensors>
    <sensor>
    ...
    </sensor>
    <sensor>
    ...
    </sensor>
  </sensors>
</data>
```

Listing 4.1: *Yderste element i pakken som bl.a. indeholder en liste af sensorelementer.*

Sensorelementet indeholder en del data om sensoren alene for at kunne identificere denne korrekt ude på serverne til normaliseringen.
Samtidig er det delt op som det ses i XML koden, med sensor datablokke, da denne måde at opdele det på gør det nemmere for dispatcheren at dele pakken op og sende dem individuelt til normalisering.

```
<sensor>
  <sensorPackageVersion>3.4</sensorPackageVersion>
  <sensorProductType>blood sugar</sensorProductType>
  <sensorProductTypeID>455</sensorProductTypeID>
  <sensorVendor>Siemens</sensorVendor>
  <sensorVendorID>55</sensorVendorID>
  <sensorSerial></sensorSerial>
  <timestampFirst>2007-11-09 T 11:20</timestampFirst>
  <timestampLast>2007-11-09 T 11:25</timestampLast>
  <patientID>1337</patientID>
  <datapoints>
    <datapoint>
    ..
    </datapoint>
    <datapoint>
    ..
    </datapoint>
  </datapoints>
</sensor>
```

Listing 4.2: *Hver sensorelement indeholder data om sensoren samt en liste af datapunkter.*

Datapoints er selve datapunktet, og det er derfor meget simpelt og indeholder det minimale. Der skal dog huskes, at et datapoint ikke er, hvad sensoren sender til gatewayen, men data gatewayen modtager fra sensoren, pakker ind korrekt format og sender videre ud til analysering.

```
1 <datapoint>
2   <datapointPackageVersion>1.0<datapointPackageVersion>
3     <value>27</value>
4     <unit>g/mol</unit>
5   <unitID>43</unitID>
6   <timestamp>2007-11-09 T 11:20</timestamp>
7 </datapoint>
```

Listing 4.3: *Et datapoint er den data sensoren har målt.*

Det blev regnet kort ud fra demodata i appendix listing A.
Filen skal forestille en patient med to sensorer og med hver to datapunkter i sendingen fra gatewayen. Filen indeholder 2178 bytes og hvis man ønsker at få en lignende pakke hvert 10 sekund i én måned vil regnestykket se sådan ud:

$$
\begin{aligned}
2178B \cdot 6\,{}^{\text{målinger}}/_{\text{m}} \cdot 60\,{}^{\text{m}}/_{\text{t}} \cdot 24\,{}^{\text{t}}/_{\text{dag}} &= 18817920\,{}^{\text{B}}/_{\text{dag}} \\
18817920\,{}^{\text{B}}/_{\text{dag}} \cdot 30\,{}^{\text{dage}}/_{\text{måned}} &= 564537600\,{}^{\text{B}}/_{\text{måned}} \\
((564537600\,{}^{\text{B}}/_{\text{måned}})/1024\,{}^{\text{B}}/_{\text{kB}})/1024\,{}^{\text{kB}}/_{\text{MB}} &\approx 540\,{}^{\text{MB}}/_{\text{måned}}
\end{aligned}
$$

540 MB pr måned for én patient må anses for at være meget, men at sende data hvert 10. sekund er måske også lidt overdrevet, derudover bør det overvejes ikke at sende i plaintext, men sende filerne binært og evt. komprimerede for at mindske dataoverførelsen. Angående pris har danske teleselskaber indført flatrate for mobil data trafik hvor 1GB/måned ca. koster 60 DKK.

Naturligvis kan man ikke uden videre fastlægge, hvilken protokol og hvordan data skal sendes, men nogle organisationer har forslag.

4.6.1 Continua

„Continua Health Alliance“ eller bare Continua[2] er en non-profit sammenslutning af mange interessenter i et system, der minder meget om det beskrevet i denne rapport.
Dette er en standard Pallas overvejer, såfremt et system skal implementeres og

vil derfor blive beskrevet kort i dette afsnit.
Continua har lavet en guideline til, hvordan de mener et telemedicinsk system skal sammensættes, og de har forslag på en protokol samt forslår at bruge Bluetooth som kommunkation til gatewayen fra sensorene.
Deres protokol ligger, ligesom XML forslaget op til, at alt data sendes med i en struktur så data kan genskabes. Alle producenter og sensortyper har keys i stedet for navne, for at sikre at data kan flettes sammen som det ønskes og at det f.eks ikke har indflydelse om teknikere skriver navne med stort eller småt.
En af de få faktorer de har med i datapakken er, hvor på kroppen sensoren sidder. Det er ikke fundet muligt at bekræfte eller udelukke om dette er en god idé, men som allerede nævnt er det optimale, at der vides så meget som muligt om målinger og omstændigheder. Så netop sensorplacering er en enhed man kan overveje om den også skal med i datapakken.
Udover dette anses Continua ikke som en fordel at implementere. Dog kunne det være noget man kunne implementere på gatewayen for at være kompatibel med Continua enablede sensorer og derved få et udvalg af sensorer „forærende“, især nu, hvor Android 4.0 har valgt at gøre Continua en oprindelig del af systemet.

4.7 Normalisering

Hvis der bruges et XML format, som det foreslåede er, det muligt at splitte data op tidligt i forløbet. Når data er splittet op kan vi normalisere den, og for at kunne lave mere generelle algoritmer til analyse, skal data være normaliseret. En normalisering kan f.eks være at omdanne sensorværdier til SI-enheder, så selvom sensoren er fra USA og benytter nogle andre skalaer og enheder end i EU, vil data blive normaliseret til det format der ønskes at arbejde i.
Dette vil betyde, at der ikke er behov for flere forskellige analyser, men kun forskellige normaliseringsalgoritmer.
Samtidig ønskes det at data gemmes i både rå sensordata samt i det normaliserede format. Man bør også gemme, hvilken normaliseringsalgoritme og version, der har normaliseret data, for på sigt at kunne normalisere det anderledes, hvis det skal bruges til forskning.

Normaliseringslaget skal ud fra modtaget datapakke kende: serienummer, produktnavn, produkttype og producent af sensoren. Ud fra disse faktorer er det muligt at lave et opslag og finde den korrekte normalisering til netop den givende sensor.
I og med at produktnavn og producent er med, kan man lave produktspecifik normalisering, der muliggør at tage højde for fejl, samt alarmere hvis man modtager værdier, der ligger uden for apparatets læsebare værdier.

Man ville kunne lave et lag i tilknytning enten før eller efter til normaliseringen, dog uden at have noget med normalisering at gøre, som kan bruges til fejlkorrigering; hvis der bruges billigere sensorer er forskellen fra den dyre ofte byggekvalitet, samt hvor præcis målingen er. Hvis man ved at en måling altid ligger 20% under den værdi den burde være, kan normaliseringen lave om i værdien og automatisk ligge 20% højere. På den måde vil man kunne ændre små fejl i sensorer (ud fra tidligere nævnt producent og produkttype). Dette vil gøre systemet billigere i indkøb af sensorer.
Naturligvis ønsker man at have korrekte sensorer, men det vigtigste for projektet er dog at indsamle data og få et bedst muligt godt patientindblik.

Når data er normaliseret er det hensigtsmæssigt at gemme data, hvordan dette kan gøres, diskuteres i næste afsnit.

4.8 Database og log/historik

Ønsket er, at man skal kunne flette data sammen med apparat og producent. Samtidig skal alarmeringen også kunne genkende hvilken patient data tilhører. Det skal ikke være muligt ved forskning at kunne se, hvem data tilhører, men kun apparat og producent.
Dette lægger op til en relational database, som bruger keys til at sammeflette dataen på tværs af tabeller.
Det, der er vigtigt er at have de rigtige felter med i tabelstrukturen. Fletningen til præsentationen kan være CPU belastende, men hvis tabellen indekseres korrekt på de felter (key/foreign key) der benyttes i fletningerne vil belastningen af datafletningen være minimal og nærmest ikke mærkbar, ligemeget hvor store datamængder der er i systemet.
Data skal ikke normaliseres overdrevet, men til et brugtbart niveau da en overnormalisering vil gøre SQL kald mere komplicerede end de egentlig bør være.

Især ved systemer som disse er det vigtigt, at alt gemmes i en database da systemets algoritmer, samt patienters personalisering og evt. neurale netværk bygger på disse værdier.

4.8.1 Normal 3NF database

Da data i databasen skal være så normaliseret som muligt er det naturligt, at det implementeres i 3 normal form (3NF).
Data kan ligge som tupler så man ved, hvem data eller hvilken sensor data

tilhører. På den måde er det altid muligt at rekonstruere data,, der blev lagt ind i databasen.
3NF lægger samtidigt op til der indexeres, hvilket som nævnt vil gøre databasen betydelig hurtigere ved læsninger.

4.8.2 Opsummering

Når data skal gemmes, er det vigtigste ikke, hvordan det gemmes, men at alt gemmes ned til mindste detalje. Ved at gemme alt altid kan rekonstruere den orginale data komplet, samt konvertere den til den standard man ønsker. En standard kan f.eks være Continua.

Efter at data gemmes stilles denne til rådighed til databehandling af diagnosticeringsalgoritmerne for at blive analyseret for tegn på kritiske fremskrivninger.

4.9 Databehandling

For at kunne behandle data og sikre at en sensor fungerer korrekt, skal der bruges forskellige matematiske modeller, alt efter hvilken sensor og datatype der måles på.
Umiddelbart kan **Lineær extrapolation** anvendes ved måling af f.eks sukkerindholdet i blod, da man kan lave en simpel fremskrivning, og systemet kan derved advare patienten om tilstanden og dermed give en forvarsel, så vedkommende kan undgå det forestående insulinchok.

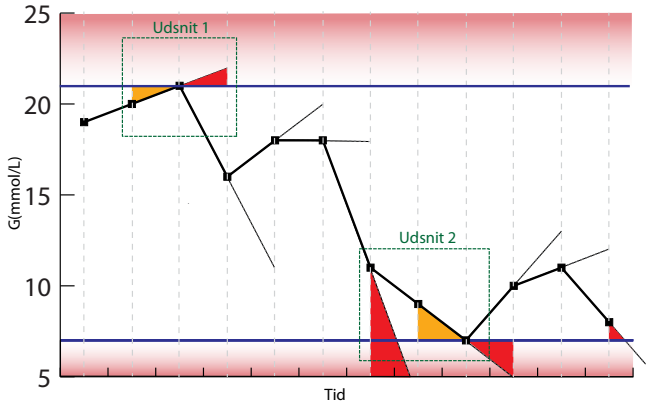
Derudover vil **Neurale netværk** og **Heuristik** også være muligheder til mønstergenkendelse.
Men der er ikke nogen grænse for hvad der kan bruges af matemastiske modeller. Dette vil være op til forskeren selv at analysere sig frem til netop den rigtige løsning ud fra vedkommendes synspunkt.
Mønstergenkendelse vil f.eks kunne hjælpe folk med lidelser, hvor deres anfald ofte opstår af samme grundlag gang på gang, hvilket kan være et udslag i temperatur samtidig med lavere blodsukker eller andre faktorer.

Derudover stilles der krav om at der skal tilknyttes historik (en database) og et feedbacksystem, hvor patienten selv skal kunne evaluere og bekræfte data. Hvis systemet ikke får disse, vil man ikke kunne udarbejde et mønster og derved ikke genkende, at patientens tilstand forværres.

Andre matematiske modeller skal overvejes, men disse bør diskuteres og forfines i samråd med læger og fagpersoner inden for den specifikke lidelse. Den vigtige pointe er at, man indser, at der er behov for et lag i arkitekturen, der skal stå for databehandlingen og analyse af data.

Det vigtige i databehandlingen er, at der tages højde for, at patienter er forskellige og derfor også vil have forskellige grænseværdier og forskellige mønstre, der viser, hvornår netop en specifik patient er ved at blive syg eller er ved at få et anfald.
Dette kræver at arkitekturen er fleksibel nok til at kunne være individuel for den enkelte patient. Af disse årsager er en generisk algoritme altid forbundet med en personaliseret datafil.

Der er opstillet en fiktiv diabetespatients data på figur 4.11 side 62 , værdierne for patienten kan være urealistiske, da det udelukkende er lavet for at illustrere nogle scenarier.
Hver lodret streg illustrerer hvornår man forventer at næste måling vil forekomme.
De blå streger ved y-aksen ved 21 og 7, er patientens personlig grænse for hvad blodsukkermåling skal ligge inden for, uden at målingen bliver kritisk.



Figur 4.11: *Fiktivt opstillet diabetespatient, der viser eksempler på simpel data fremskrivning og alarm.*

Diagnosticeringen er en simpel liniær fremskrivning, der baseres på to punkters liniær fremskrivning, hvor det nyeste punkt og det forrige punkt bruges til analyse af dataen, og disse fremskrives med én tidsendhed, som er markeret med stiplet linie. Dette kan opstilles matematisk ved:

$$a = \frac{y_2 - y_1}{x_2 - x_1} \tag{4.1}$$

$$b = y_1 - a \cdot x_1 \tag{4.2}$$

$$y_3 = a \cdot x_3 + b \tag{4.3}$$

Hvor y_3 er den fremskrevne værdi der skal sammenlignes med patientens maksimum og minimums værdier og alt efter, hvor meget over eller under den fremskrevne værdier er alarmerer systemet.
I udsnit 1 ses det, at grafen efter punkt 1 og 2 bliver orange. Dette skal indikerer at der er en mistanke om, at såfremt patienten ikke gør noget, kan det risikeres at patientens tilstand forværres.
Næste værdi er indsat liniært og viser at den forrige fremskrivning ville vise hvad der ville ske: Værdierne fortæller der nu er en kritisk situation og vil advare patienten om situationen.

I udsnit 2 fremstilles et hurtigt fald, over en kort periode. Dette betyder, at i tilfældet af patientens værdier vil ramme et kritisk niveau inden næste måling og at patienten bør reagere hurtigt på systemets alarm. Allerede her vil man advare patienten, da det nu er ved at nå et kritisk stadie.
Hvis der alarmeres kan det være, at tilstanden forbedres og bliver knap så kritisk som forventet og der vil nu være tale om en falsk positiv eller en fejlalarmering pga. forsigtighed.

Dette er dog en meget simplificeret matematisk måde at se disse værdier på og vil formentlig ikke være brugbar i et reelt system.

Indefor samme område er der skrevet en artikel [10] som benytter Model Predictive Control (MPC) og artiklens undersøgeleser basere netop også diabetes patienter. Her medtages i beregningen, hvornår folk spiser og hvor meget de ca. indtager i måltidet.

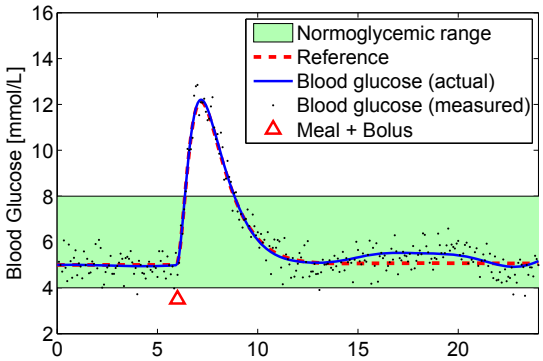
Deres analyse baseres på, at patienten har en automatisk insulinsprøjte som styrer patientens sukkersyge og ud fra hvad patienten fortæller, via feedforward-feedback, øges eller mindskes insulinindsprøjtningen. Figur 4.12 viser en illustration fra artiklen, som viser at de holder sig til den ønskede værdi. Billedet er fra en patient der har før måltidet har indberettet måltids mængden og isulin mængden justeres efter dette.

De skriver om dette:

> „These results show that reasonably good control can be obtained when a feedforward-feedback strategy is used, even for uncertain systems. However, the main limitation of this strategy is that a fairly good nonlinear model description of the patient must be available.“

Konceptet bag feedforward-feedback kan bruges til patienter, der er afhængige af medicin og at medicinen tages rettidigt for at holde dem på det rette niveau. Der ligger umiddelbart flere fordele i denne måde både for patienter såvel som lægen.

Praktiserende læge Ib Koldbæk udtalte (Feb-2012) at dette også vil kunne



Figur 4.12: *Billede fra ESCAPE-21 hvor en patient har opgivet korrekt afmåling af måltidets indhold[10].*

bruges ved patienter, der skifter til et kopipræparat, da det ikke er sikkert, at en patient optager kopipræparatet på tilsvarende måde som orginalproduktet. Dette vil ved hjælp af et feedforward-feedback system opdages, hvis patients vitale værdier ikke reagerer efter, at patienten har indmeldt, at medicinen er taget.

En diagnosticering kan kun bruges til forskning. Hvis den skal bruges i et alarmerende system, skal algoritmerne kunne videregive information til et system, der vurderer om der skal alarmeres eller ej.

4.10 Alarmering

For at kunne underrette patienten om faresituationer eller i vanskelige tilfælde underette alarmcentralen, skal der udarbejdes et lag i arkitekturen, der kan alarmere. Man bør dog overveje at en alarm skal have forskellige funktioner, da folk er forskellige samt hvilke lidelser eller symptomer patienten har. Af den grund skal der indføres triage.

4.10.1 Triage

Der skal indføres triage af alarmeringen. Dette skal både være i henhold til, hvordan der alarmeres, men også ved at prioritere nogle emner højere end andre. Eksempelvis skal en alarmering på en pacemaker prioriteres højere end en „lille“ alarm på, at patienten har feber.
Når dette er sagt, bør det også overvejes, at der kan laves grupper af alarmering. Derved menes der, at nogle alarmer ringer eller sender SMS til patienten, og i akutte tilfælde sender en besked til alarmcentralen, der underrettes, og dermed kan reagere med det samme.
Hvis eksemplet med diabetikeren anvendes, behøver man ikke sende en ambulance, fordi patienten 10 minutter senere gerne skulle have tage sin medicin; her kan SMS eller lignende notifikation være rigeligt. Modsat skal en patient med pacemaker, der har en puls på nul, eller hvor pacemakeren melder den giver stød, alarmere alarmcentralen at der er behov for direkte udrykning.[8]
Dette skal tages i betragtning, da der er forskel på patienterne og deres lidelser.

Det vil samtidig være fordelagtigt at sende GPS/adresse sammen med en alarmering, da alarmcentralens personale så har et fikseret lokation de skal besøge og ikke blot sidder med en formodning om patientens placering.

Gatewayen skal også kunne alarmere patienten lokalt. Dette er dog ikke en analyse alarmering, men en alarmering om lavt batteriniveau eller hvis den indsamlede data har problemer med at blive sendt ud til analyse.

Samtidig kan man godt benytte gatewayen som alarmeringsmodtager, hvis der skal sendes en lavprioritetsalarmering, som f.eks at diabetikeren skal huske at tage sin medicin eller anden information/rådgivning.

Man skal dog være opmærksom på at såfremt gatewayen bliver brugt til alarmering af patienten, vil den indgå som medicinsk udstyr som beskrevet i afsnit 4.12 side 71.

Der skal naturligvis ikke bare alarmeres så snart en diagnostisering mener der er en alarm. Det risikeres at få sekunder efter laver diagnosticeringsalgoritme en neutral analyse hvor patienten har det godt. Man kan evt. bygge noget logik op ved hjælp af en mistanke vektorer, et projekt „Hierarkisk distribueret IDS ved anvendelse af neural net“[5] der handler om „intrusion detection“, hvor der bruges neurale netværk til at afgøre om datapakker ser normale ud. Disse benytter en „suspicion vector“ hvor unormale hændelser øger mistanken og hvis der i en periode ikke er nogle trusler sænkes mistanken langsomt. Samme grundlag som disse ville kunne bruges til at detektere om der skal alarmeres eller ej.

4.10.2 Patientgrupper

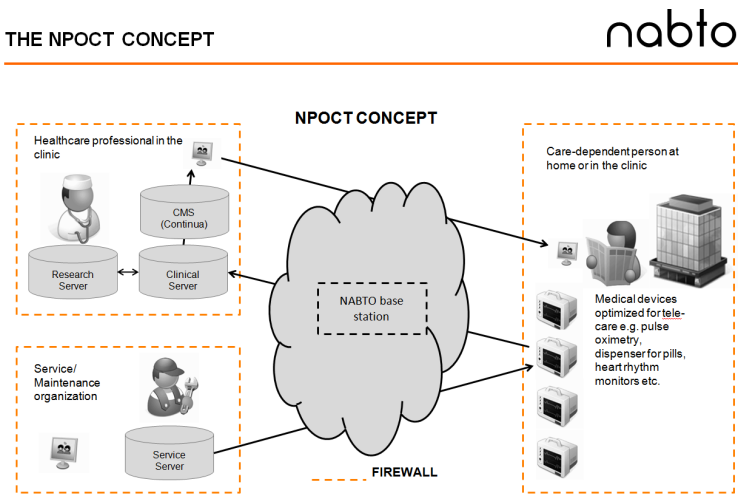
Ideen er at opdele folk i grupper. Det kan være lige fra børn, ældre eller særligt udsatte (fokuspatienter).
Man kan lave forskellige grupper, baseret på hvilken serviceaftale, der er lavet. Dette kan dog diskuteres om det er etisk korrekt at lave det sådan. Alarmcentralen kan også have et køsystem for at sikre, at alle får hjælp, derfor er det også vigtigt, at systemet kan se forskel på patienter og placere fokuspatienter samt folk med farlige lidelser forrest i køen hos alarmcentralen.

4.11 Faggruppers tilgang

Pallas er med i et projekt hvor flere firmaer, Nabto, DTU samt andre arbejder, på at få et framework for telemedicin kaldet nPoct. Ideen er at lave en telemedicinsk arkitektur til en „hyldevare“ og derved hjælpe producenterne af sensorer til at komme nemmere ud på markedet. nPoct er stadig i et meget tidligt stadie men jeg har alligevel valgt at benytte en af deres illustrationer, da den viser en god tankegang.
Ifølge den oprindelige nPoct ide, er der flere faggrupper der skal bidrage med fagspecifik viden, samt kunne tilgå og operere på forskellige dele af systemet. Nabto har patent på nogle teknologier, der gør det muligt altid at kunne tilgå f.eks en sensor; selvom den er bag firewalls og routere.

Figur 4.13 viser Nabtos egne teknologier er placeret på nettet og via disse teknologier kan forskellige interessenter tilgå nødvendige ressourcer.

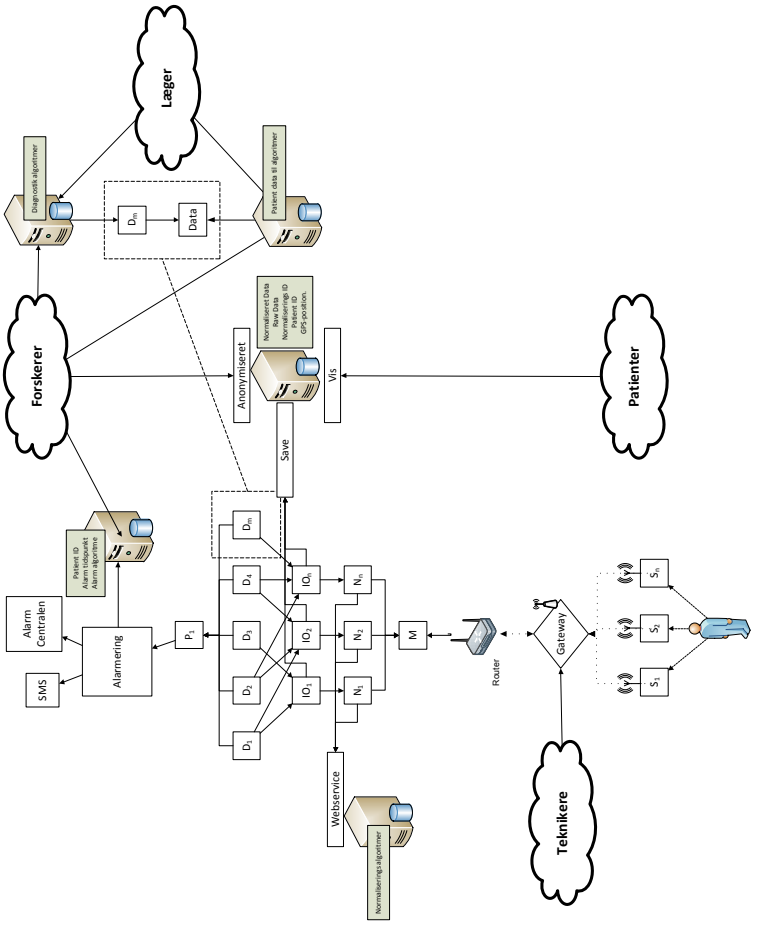
Enhver telemedicinsk løsning vil have forskellige interessenter, som her i rapporten kaldes faggrupper. Disse skal bidrage med fagspecifik viden, samt kunne tilgå og operere på forskellige dele af systemet.



Figur 4.13: *nPoct concept tegning der viser at kommunikationen over nettet benyttes med mange faggrupper som interessenter (Billede udlånt af Pallas)*

Faggrupperne er forskellige og har hver deres interesseområde, disse interesseområder vil forklares i nedenstående afsnit.

Figur 4.14 side 68 viser hvor i systemet fagrupperne ønsker at hente data. Hvor databaserne ligger skal ikke anses som faktisk da databaserne kan ligge på en og samme server i en evt. implementation.



Figur 4.14: *Viser arkitekturen og hvad faggrupperne ønsker at kunne tilgå. Stregerne symbolisere hvor i systemet denne data kan findes.*

4.11.1 Læger

Lægerne er den generelle bruger af dette system og derfor også gruppen der skal have flest muligheder i systemet.
Lægerne skal have et nemt overblik over algoritmer og være sygdomsopdelte, så det bliver så nemt som muligt at vælge en algoritme til en patient. Man kunne samtidigt vise, hvor mange patienter der bruger den pågældende algoritme, for at vise om det er en brugt algoritme eller ej; dette vil give lægen et indblik i „kvaliteten"af algoritmen'.
Lægen skal derefter brugervenligt kunne tilknytte en algoritme til en patient.
Af samme grund skal lægen vide, hvilke sensortyper patienten har tilknyttet.
Lægen skal, ikke mindst, også have mulighed for at se en given patients vitale værdier på datatypens bedst mulige måder (grafer:xy, violin, søjle osv diagrammer), hvor visningen skal kunne begrænses af tidsperioder.

Kort opsummeret:

- Se algoritmer.
- Se en specifik patients data.
- Se patients sensorer.
- Se liste og mulige algoritmer til en lidelse og tilknytte en algoritme til en patient.

4.11.2 Tekniker

En tekniker skal kunne tilgå gatewayen for at kunne opdatere en applikation eller parametre i en applikation.
Dette kan ske på grundlag af kommunikationsprotokol skifte eller hvis der er timiningsproblemer, samt reparation og vedligholdelse i sammenhæng med gatewayen.

Hvis dette skal kunne gøres uden at have enheden i hænderne skal man evt. have kendskab til enhedens egen IP, som skal hentes ud fra datalaget på serveren.

Teknikerens opgaver:

- Opdatere gatewayfunktionaliteten.
- Generel vedligeholdelse af systemet (algoritmer indføres osv.).

4.11.3 Forskere

Forskere ønsker mulighed for at kunne hente anonymiseret data til forskning af sygdomme og menneskets vitale værdier generelt, men også til fremtidig forbedring af algoritmer i systemet.
Hvis de skal kunne forbedre algoritmerne i diagnotiseringslaget, ønskes det at lagret data kan benyttes i arkitekturen.

Forskernes fokus punkter:

- Modtage anonymiseret data.
- Kunne benytte lagret data som var det sensordata.
- Oprette lignende netværk til træning af neurale netværk.

4.11.4 Patienten

Patienten har mulighed for at se egne data på grafer, for selv at se hvordan værdierne ser ud og opnå en følelse af, at der er overblik over egen situation.
Patienten skal derudover også kunne indmelde anfald, der ikke er blevet registreret af systemet og ligeledes kunne fortælle, at en alarmering har været en fejlalarmering (falsk positiv).

Patientens ønsker:

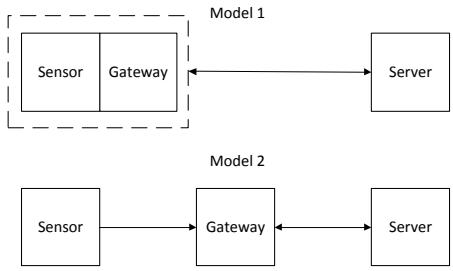
- Tilgå egne data f.eks på PC eller mobiltelefon.
- Rapportere anfald, der ikke blev fanget af systemet, selvom dette kan give anledning til bedrag samt forkerte indrapporteringer.

Da der netop er mange fagrupper og nogle informationer er personfølsomme er der nogle regler der bør overholdes.
I næste afsnit beskrives der, hvilke faktore, der er i spil for om et produkt er et medicinsk udstyr eller ej.

4.12 Regulatorisk og certificering

Danmark er et regelbaseret land, hvilket betyder at der naturligt også er regler for, hvordan telemedicinske systemer må fungere indenfor den gruppe de klassificeres.
Ved gennemgang af litteraturen om certificering af medico tekniske apps og medicinsk udstyr generelt, var det begrænset, hvad der blevet fundet om forskelle i certificering med hensyn til, hvor den eksterne kommunikation placeres.[29],[13],[20]

Idégrundlaget kan ses på figur 4.15, hvor to modeller opstilles.
Model 1 viser et system, hvor gateway og sensor er en og samme enhed.
Model 2 viser et system, hvor sensor og gateway er splittet om i to enheder.



Figur 4.15: *Viser placering af gateway, der kan gøre en regulatorisk forskel, da der er forskel i hvor den ekstern kommunikationen er placeret.*

Forskellen er primært i kommunikationen. I model 1 går forespørgslen fra internettet direkte til sensoren.
Dette kunne man i værste scenarie se sig hacket og at en evt. patient med pacemaker ville hackeren kunne giv stød eller anden vis kunne beskadige patienten.

Model 2 bygger i modsætning på at sensoren har envejskommunikation til gatewayen og gatewayen har tovejs kommunikation til serveren. Dette betyder i praksis, at en hacker stadig ville kunne tilgå gatewayen, som vil kunne ødelægge den kontinurlige datatransmission eller lignende, men vil altså ikke kunne beskadige patienten.

Rådets Direktiv 93/42/EØF[13] skriver i „Bilag IX, stk. 2.3“

> „Edb-programmel, som styrer en anordning eller påvirker anvendelsen af en anordning, klassificeres automatisk i samme klasse som den pågældende anordning.“

European Commission skriver i „Annex 1 section d“[20]:

> „The healthcare sector uses communication systems (e.g. email systems, mobile telecommunication systems, video communication systems, paging etc.) to transfer electronic information. Different types of messages are being sent such as prescriptions, referrals, images, patient records, etc.
>
> Most of the communication systems handle other types of messages other than medical information. This communication system is intended for general purposes, and is used for transferring both medical and non-medical information.
>
> Communication systems are normally based on software for general purposes, and do not fall within the definition of a medical device.
>
> Communication system modules might be used with other modules that might be qualified in their own right as medical devices.“

Disse to citater beskriver mere eller mindre forskellen mellem modellerne i figur 4.15. Da den ene model gør gatewayen til en del af et kommunikationssystem, vil derfor ikke indgå som en medicinsk enhed.
Og modellen „Model 1“ hvor gatewayen bliver en del af sensoren (smarte sensore), og gatewayen derfor kan interagere med sensoren. Dette vil betyde at gatewayen skal certificeres på samme niveau som sensoren.

Lovgivningen for medicinsk udstyr i EU beskrives i direktivet. Direktivet definerer i Artikel 1, stk 2a., at software der er beregnet til specifik anvendelse til diagnostiske formål anses som værende et medicinsk udstyr. Klassificeringen af softwaren afhænger af, hvilken klassificering det pågældende udstyr som softwaren kobles til, har. Der er stor forskel på om softwaren kobles til klasse I eller et klasse III udstyr og en regulatorisk strategi skal udarbejdes når det er undersøgt nærmere. Derefter bestemmes hvilket medicinsk udstyr den skal kobles til. Softwaren kan i princippet blive CE-mærket som et selvstændigt udstyr, hvor det så specificeres, hvilke typer medicinsk udstyr det kan benyttes med. En anden mulighed er at lade producenterne af medicinsk udstyr CE-mærke softwaren som en del af deres udstyr.

Overvejelsen med at kunne alarmere via gatewayen, der kunne være en telefon, skal man dog være varsom med, hvis man ikke ønsker at gatewayen skal ses som

et medicinsk udstyr: Det beskrives i reglementerne at et batteridrevet produkt, der kan alarmere eller diagnosticere en patient, er et medicinsk apparat.

Rådets Direktiv 93/42/EØF[13] skriver i artikel 1 punkt 2.

> „Medicinsk udstyr: ethvert instrument, apparat, udstyr, software, materiale eller anden genstand anvendt alene eller i kombination, herunder software, som af fabrikanten er beregnet til specifik anvendelse til diagnostiske og/eller terapeutiske formål, og som hører med til korrekt brug heraf, og som af fabrikanten er beregnet til anvendelse på mennesker med henblik på: — diagnosticering, forebyggelse, overvågning, behandling eller lindring af sygdomme
> — diagnosticering, overvågning, behandling, lindring af eller kompensation for skader eller handicap.“

Disse regler er umiddelbart kun tilknyttet Danmark og de andre EU-lande, men hvis man ønsker, at arkitekturen kan bruges i USA, bør dette undersøges hos det amerikanske U.S. Food and Drug Administration (FDA).

4.13 Eget forslag til telemedicinsk system

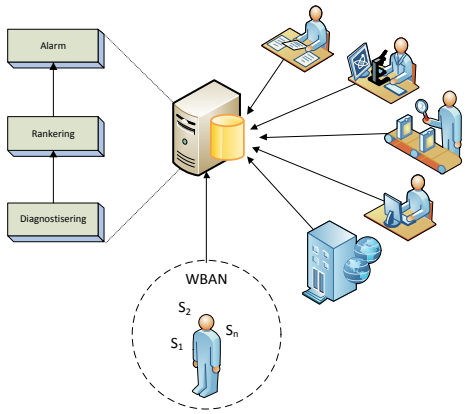
Analysen er gennemgået og ud fra denne vil følgende anbefaling gæld for at opbygge et system til telemedicinsk overvågning.
Til gatewaystrukturen anbefales WBAN og med Bluetooth LE som kommunikationslag, da dette vil medføre lavt batterikrav, understøtter parring af enheder og har en tilpas rækkevidde af data. Derudover er Bluetooth en gammel gennemtestet standard, som er lige til at gå til.
En simpel sensor er også at foretrække, hvis man vil gå op i certificering, da man så vil kunne købe en billigere sensor, hvor producenten har fået deres sensor certificeret på det rette niveau, så der ikke behøves at tage højde for dette.

På serversiden, er det vigtigt at inkludere normaliseringen da denne vil kunne rette fejl ved billigere sensorer, samt behøver man ikke købe dyre europæriske sensorer, men kan nøjes med billigere sensorer fra et andet land, da disse bare skal normaliseres til den rette enhed.

State of the art afsnitet „Secure Linking of Patients and Tele-medical Sensors“ afsnit 2.3 viser at det er svært at genkende folk ud fra kun en sensor. Dette vil

også være et problem til senere analyse af patientens vitale værdier, og derfor skal diagnosticeringen kunne benytte flere sensoreres data for at kunne have et optimalt virke.

Artikler, viser at fremskrivning og evt. regulering af data kan lade sig gøre automatisk[10].
Tanken med at det ønskes, at en patient godt kan have flere algoritmer/diagnosticeringer, der måler på samme værdier er et must, da der kan være forskel på forskeres tankegange og tilgangsvinkler. Dette betyder samtidigt, at ingen algoritmer skal forkastes uden lige, da netop en algoritme ikke er bedre end andre uden det ud fra den indsamlede data kan bevises, at den absolut ikke fanger nogle alarmer via fremskrivninger. Disse emner tydeliggør at en 100% parallel diagnosticeringsmodel er at foretrække.



Figur 4.16: *Forslag, til en ombygning, som indrapportere data, og interessenter har datatilgang til.*

Systemet vil have mange interessenter, såsom læger og forskere. Det er desuden vigtigt at overveje, hvordan data skal udbydes til disse; webservices ville være et godt forslag.
Det der skal forblive i tankerne er at hvis forskere og læger ikke har tilgang til systemet og data, bliver projektet aldrig godt, da de forskellige faggrupper spiller en stor rolle.
Det er derfor vigtigt, at både data og alarmeringer gemmes til senere brug.

4.14 Opsummering

Der er som ved alt andet fordele og ulemper ved netop denne måde at gøre det på.
Først og fremmest er der gateway løsningen. Denne kan være problematisk at få stabil, og samtidig være en dyr løsning, hvis patienten kun har én sensor der skal kommunikeres med.
I forhold til de andre nævnte løsninger ses denne dog stadig som den absolut mest brugbare.
Hvad angår gatewayens pris, skal man vurdere dette op imod en specialproduceret sensor, hvilket også hurtigt vil blive dyr.
Ser man på det regulatorisk, ses det, at gatewayen vil blive certificeret som en kommunikationenhed og ikke et medicinsk udstyr.
Det medicinske udstyr vil altså være server delen og sensoren vil være producentens pligt at få certificeret hvis den skal bruges til medicinsk brug.

Normaliseringen kan være en stor opgave at skulle oprette for at indføre en ny sensortype, men denne ses som nødvendig, da normaliseringen gør algoritmerne mere generiske, så i stedet for at have en algoritme pr. sensor er det hensigtsmæssigt at have en generisk algoritme og en simpel normalisering/konvertering i stedet.

Diagnosticeringen bør køre parallelt, for at sikre svartider, for et system der skal alarmere på lidelser.
Det at køre det parallelt medfører at alle algoritmer kører. Hvis der var blevet brugt early out sortering, vil der kunne spares på performance/server kræft, men samtidigt opstille dillema lignende scenarier, hvis der skulle prioteres mellem algoritmerne.

Sidst, men ikke mindst, skal data gemmes og hvilket format det gemmes i er mere eller mindre underordnet, så længe det er normaliseret.
Ud fra normaliseringen vil man altid kunne konvertere dataen til det format man ønsker, og dette betyder også, at projektet ikke behøver at implementere en protokol som f.eks Continua.
Hvis man senere finder ud af, at Continua er en standard man ønsker at protokollen følger, kan den opsamlede data fra databasen som sagt konverteres til dette uden problemer.

Kapitel 5

Implementeringsforslag

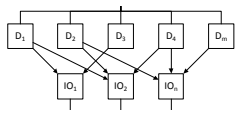
Da projektet har været et modelleringsprojekt, har det ikke inkluderet specifik kode.
I starten af projektet blev der oprettet en VDM model for at give et overblik over om det kunne lade sig gøre, VDM modellen blev dog nedprioriteret og lagt i bilag B. Modellen blev droppet da det hurtigt stod klar at ideerne kørte på teknologiske kendskaber, der tydeligt viser at det ikke er noget, problem at lave netop et telemedicinsk system som omtalt i nærværende rapport.

Grundet de mange overvejelser i projektets struktur er flere forslag skitseret, hvor det vises hvordan nogle af lagene i strukturen kan implementeres..

Dette afsnit gennemgår nogle af lagene, og derudover beskrives nogle ideer til implementeringer og begrunder, hvorfor disse måder findes favorable.

5.1 Mellem lagring og diagnosticering

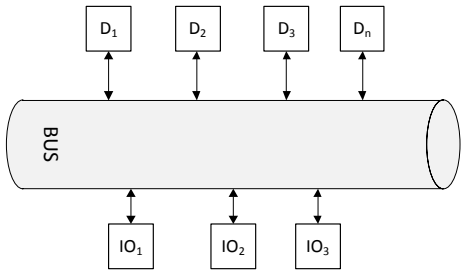
Leddet mellem det normaliserede data fra sensorerne og selve diagnosticeringsanalysen ses på figur 5.1. Det kan laves på flere måder. Det ønskes, at en diagnosticering kan benytte flere sensoreres data, ligemeget hvor data modtages fra.



Figur 5.1: *Serverside udsnit*

Af denne årsag vil man hurtigt kigge på et distribueret og evt. et centraliseret system med distribuerede klienter.
Det skal kunne modtage data på forskellige måder og via forskellige modtagelsespunkter, og samtidigt skal den diagnosticerende analyse udarbejdes et centraliseret sted.
Dette gør systemet til et system med centrale servere og distribuerede klienter, eller evt. et distribueret system med central styring.

Dette kan f.eks gøres ved at bygge et busbaseret system se figur 5.2. Dette bygger på provider og subscriber paradigme, hvor alle sensorer sender data ud på bussen. Derefter er det diagnosticeringsenhedens opgave at modtage den rette data. Dette lyder som en simpel og ligetil løsning, der er nem at implementere og gøre funktionel. Der er et stort problem ved den oprindelige databus; hvis man har 10.000 sensorer, der alle sender data ud på en samlet bus, skal f.eks 5.000 lidelsesanalyser undersøge, hver eneste pakke, der sendes ud og analysere om de skal bruge denne.



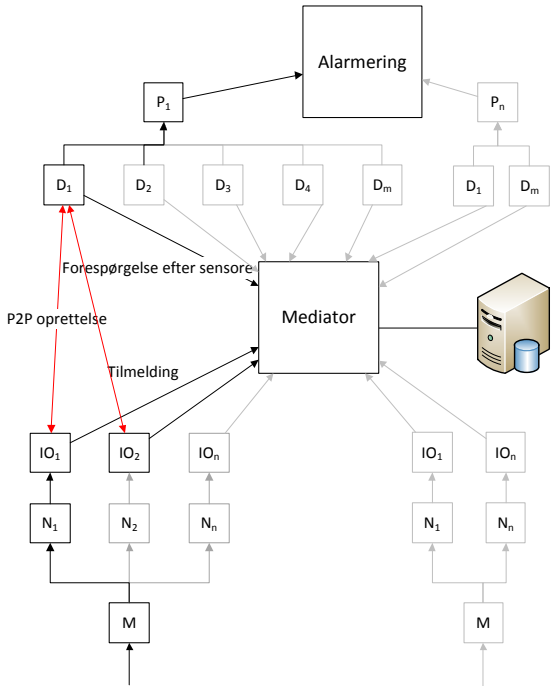
Figur 5.2: *Bus arkitektur*

Derved vil der bruges mange kræfter på at sortere og modtage data. Der findes forskellige måder at filtrere data på i selve bussen, men alle har dog tilfælles at noget skal overvåge bussen for at kunne sortere. Overvågningen vil kræve resourcer af systemet og belastningen vil kun stige i takt med at systemet bliver større. Derfor kan denne idé hurtigt forkastes, da den på sigt vil skabe en flaskehals.

Man vil kunne holde sig til mere moderne løsninger kaldet Enterprise Service bus (ESB) [14],[6] som er en software baseret bus og ikke rigtigt har nogle rammer

for hvornår man kalder det en bus.
Forskellen ligger i, at i en ESB lægges der logik i „bussen“ selv, dette kan være alt fra kryptering, forretnings logik såvel som, hvordan der forbindes mellem provider og subscriber. Dette vil altså sige at en sådan bus vil kunne minimere sensorernes egen lytning på pakker, da der sørges for dette i bussen. Generelt virker ESB som en stor udefinerbar enhed, hvor der ikke er nogle regler og restriktioner.
Det kan af samme grund virke som et overdrivelse til denne type system.

Grundprincipperne fra en busarktitektur kan overføres i en „peer to peer“ løsning, (frem efter kaldtet P2P) hvilket er en mere jordnær løsning end ESB, man ville dog naturligvis godt kunne lave denne type system inde i en ESB.
Hovedpunktet er, at man ikke ønsker, at der kommunikeres unødigt til elementer i systemet, ikke har behov for den pågældende data. En simpel måde at lave en P2P løsning er, ved at gøre som i det gamle Napster program, hvor man havde en centraliseret Mediator som klienten forbinder, og spørger efter, hvor en given ressource kan findes.



Figur 5.3: *Serverside mediator, hvor IO laget melder sig til ved den centraliserede mediator, og diagnosticeringen efterspørger, for senere at oprette forbindelse hvis den eftersøge datatype findes.*

Netop denne mediatoropstilling er ideel til dette led i systemet og kan ses på figur 5.3 side 79. En sensors normaliserede data forbindes til en mediator og informerer mediatoren, hvor den kan findes, hvis data den indeholder samt hvad den kan levere af services til analyselaget.

Diagnosticeringslaget vil efterspørge mediatoren, hvor den finder en givet sensortypes data på en specifik patient. Hvis data eksisterer, leveres en IP og port tilbage til diagnosticeringslaget som derefter vil oprette en client-server (P2P) forbindelse direkte til den normaliserede datastream. Mediatoren sørger altså for forbindelsen og så snart den har „hjulpet“ med, at de kan finde hinanden, sørger den ikke for mere, og derved belastes hele systemet ikke med unødvendig data.

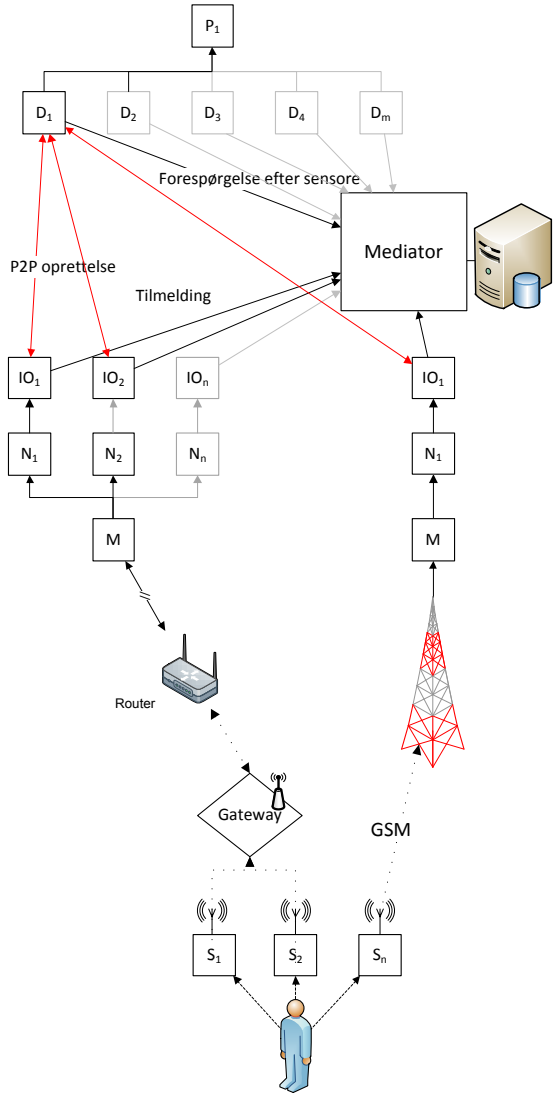
Hvis f.eks seks diagnosticeringer abonnerer på samme sensors data, vil dataen blive sendt seks gange. Dette kan virke lidt redundant, men det giver god mening at der også sendes TCP, da vi vil sikre os, at de der ønsker data har modtaget den.

En anden fordel er ligeledes, at denne måde at forbinde diagnosticering med data muliggøre, at data kan modtages på mange måder og forskellige steder i systemet, hvilket også betyder at systemet uden problemer kan modtage data fra GSM og WiFi samtidigt. Diagnosticeringslaget vil ikke se forskellen.

Figur 5.4 viser tre sensorer, der er forbundet til patientens gatewayløsning, derudover har patienten også en sensor med indbygget logik (tidligere kaldet en smart sensor). Denne kunne f.eks måle rumtemperatur i patientens soveværelse. Patienten bor i Jylland og rumtemperaturen indmeldes til den nærmeste server i Jylland ved at bruge GSM. Patientens gateway er dog blevet sat forkert op og sender derfor til en server på Sjælland. Normaliseringen sker normalt, og når data skal videre gives til diagnosticeringen oprettes dette via mediatoren.
Diagnosticeringsalgoritmen vil ikke bemærke, at noget data modtages i Jylland og andet på Sjælland.

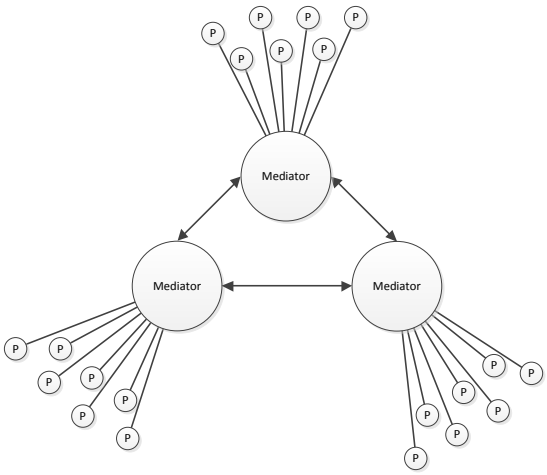
Mediatoren bliver et knudepunkt i systemet, ikke pga. datatrafik, men den er ansvarlig for, at de korrekte forbindelser bliver oprettet. Dette kan også resultere i et 'Single point of failure' og af samme årsag vil det give anledning til at oprette flere redundante mediatorer, hvis én skulle svigte. Til dette ville principper fra DNS opslag være en mulighed, da DNSen vil udføre loadbalancing mellem de givne mediatorer. Dette kræver at alle mediatorer er 100% opdaterede og ens, da man ellers risikerer, at en mediator svarer en diagnosticering, men at den givne sensor ikke findes, selvom en anden mediator kender den.

Hvis systemet bliver endnu større, vil man kunne benytte sig af en struktur meget lig den forrige med mediatoren som ses på figur 5.3, dog hvor der indgår flere



Figur 5.4: *Mediator brugt med data gennem forskellige kanaler eller modtage punkter, men hvor diagnosticeringen stadig får den korrekte data.*

mediatorer, derved vil systemet minde om et distrubueret system med superpeers [19] [18]. At introducere flere mediatorer (superpeers) på landsplan og lade disse mediatorer kende hinanden, gør det muligt at lade en mediator beskæftige sig alene med patienter fra deres egen landsregion. Hvis der forespørges på noget, den ikke selv har kendskab til, kan den spørge de andre mediatorer om det samme og hvis en af de mange mediatorer har indholdet, vil det blive fundet. Dette vil altså stresse mediatorernes belastning, dog kun hvis det forventes at der normalt spørges til den korrekte mediator fra start. Men ved at dele dem op i landsregioner, vil det ved et systemuheld kun ramme en landsregion og ikke hele systemet.



Figur 5.5: *Serverside mediator som superpeers*

Figur 5.5 viser at man kan have flere mediatorer og hver mediator har ansvar for nogle patienter. Hvis der ledes efter en patient, kan man blot spørge en mediator. Hvis denne ikke selv har patienten vil den spørge de andre mediatorer.
Funktionelt kan man se en mediator som en praktiserende læge, der kender sine patienter samt andre læger.
Et forslag vil være, at hver landsdel har sin egen mediator.

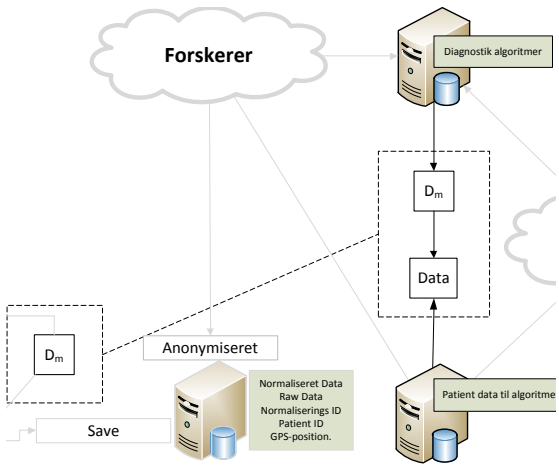
5.2 Normaliseringen

Normaliseringslaget skal kunne behandle data fra en sensor og normalisere denne, for at få ensartet normaliseret data.
Da der ikke er brug for database data eller andre dele, der kræver patientkendskab, kan dette med fordel implementeres som en webservice. Det essentielle

er dog, at både rå sensordata, normaliseret data, algoritmem og version for normaliseringen returneres, da denne skal gemmes senere. Ved at gemme alle disse faktorer muliggør man, at kunne genskabe andre scenarier ud fra andre normaliseringer, samt finde fejl i evt. normaliseringer.

5.3 Diagnosticering

I diagnosticeringen er det vigtigt, at beregningerne er så personspecifikke som muligt, derudover bør algoritmerne være generiske som muligt og have et fælles interface, der kan starte og alarmere på en standardiseret måde.
Ved at lave dem med fælles interface, kan det også lade sig gøre at have algoritmerne placeret i databaser og først når de skal bruges hentes de ud fra databasen. Algoritmen skal selv slå op i en anden database og hente patientens personliserede indstillinger (f.eks max og minimum værdier i patientens sukkersyge) til netop denne algoritme.



Figur 5.6: *Viser en loop effekt, for at vise at dette kan implementeres med generiske algoritmer, der bliver personliserede*

På den måde kan algoritmerne startes og stoppes så meget man ønsker. Figur 5.6 er et udsnit af figur 4.14 side 68, figuren, viser hvordan diagnosticeringen er opdelt i selve diagnosticeringen samt en data del.

Kapitel 6

Evaluering

Der kigges i denne evaluering nærmere på nogle af elementerne i projektet samt bruges scenarier, som det tænkes at det telemedicinske system kunne blive brugt.

Ideelt set, skulle nogle sensorer opkøbes og undersøges, men da dette ikke var muligt, blev dette gjort teoretisk i stedet. Da Bluetooth LE (BLE) anses for den bedst mulige kommunikationsvej, grundet f.eks parring med enheder, er evaluering lavet på grundlag af Bluetooth LE protokollen.

Når to BLE produkter parres, sendes informationer frem og tilbage mellem dem, deriblandt: vendorId og serialNumber, der skal være i datapakkerne[30]. Disse informationer sendes ikke med i den enkelte datapakke fra sensoren, derfor er det vigtigt, at disse gemmes ved parringen, så de kan benyttes fremover i datatransmissionen.

Dog kan man ikke sikre sig, at billige producenter opfylder protokollen for parring korrekt. Hvis der tages højde for dette på gatewayen (telefonen), vil man kunne parre og indtaste de nødvendige informationer i telefonens program, da disse ofte står på produktet selv.

BT lægger ikke op til at man sender batteriniveau med i datapakkerne, men ved undersøgelse af andre produkter blev et BT headset fundet, der netop forbindes til en telefon, og sender headsetets batteriniveau med over til telefonen gennem

BT kommunikationen[24].

Derudover blev der til sidst i projektet fundet en BLE sensor med 1-10 års batteri levetid på et knapcelle batteri, og denne sensor indeholder temperatur, accelerometer og sender batteriniveau med over i dataprotokolen, sensoren måler 15·22mm.[7]

Dette beviser, at det kan lade sig gøre.

Der er opstillet to fiktive scenarier, der ikke bygger på lægelige kvalifikationer. Dette betyder også, at der ikke vides om disse scenarier kan måles med de sensorer der foreslåes.

6.1 Scenarie 1 - Diabetes

En diabetes patient har haft problemer med at kontrollere sit blodsukkerniveau og har af samme grund søgt læge. Lægen anbefaler at han bliver oprettet i det projekt baserede system.
Lægen opretter patienten i systemet med CPR-nummer, benytter patientens telefon nummer til kontakt og tilføjer en matematisk simpel liniær (lig den beskrevede i figur 4.11) fremskrivning af patientens blodsukker værdier. Udfra samtale med patienten definerer de, at patientens egne grænser ligger på minimum 5 og max 21 G($^{mmol}/_{L}$).
Patienten bliver spurgt om han har en smartphone, hvilket han i dette tilfælde har. Lægen fortæller at patienten kan hente en app på telefonens software marked. Da dette er gjort finder lægen en sensor (lavet som et plaster) frem, som lægen sætter på patientens mave.
Ud fra en guide på telefonen bliver patientens egen smartphone over Bluetooth parret med blodsukker sensoren.
Patienten indtaster desuden sit CPR-nummer i den nyinstallerede applikation. Smartphonen viser nu, at der sendes data hvert minut til en server.
Inden patienten forlader lægehuset kontrollerer lægen om dataen bliver modtaget af systemet, for at kontrollere, at alt er sat korrekt op.
Der er nu ikke ikke andet at gøre end at vente på at se om systemet reagerer på en evt. alarmering.

Nogle uger er gået og lægen ringer nu tilbage til patienten. Lægen fortæller patienten, at hans værdier ligger glimrende og han bare skal fortsætte som han har gjort hidtil.
Denne episode har netop sparret patienten for at tage til lægen for en samtale og, en rutine undersøgelse.

Rent teknisk efter at applikationen var installeret på telefonen og blevet parret med sensoren, er telefonen, som har fungeret som gateway, begyndt at sende værdier ind til diagnosticering.
Data bliver først modtaget på serversiden, idet patienten her kun har én sensor bliver denne sensors data givet videre til normaliserings laget. Normalisering laget ved hvilken sensor det er og kan se det er fra et amerikansk mærke der opgiver i $^{mg}/_{dL}$ og da algoritmerne er standardiserede til $^{mmol}/_{L}$ dividerer normaliseringen tallet med 18.
Data gemmes herefter og der fortælles videre i systemet, at dette er diabetes målinger fra en patient.

Diagnosticeringslaget vil nu modtage en værdi som den kan bruge til at analysere på, 10 sekunder efter modtager diagnosticeringen endnu en værdi. Ud fra det nye punkt og det forrige laves der nu en analyse samt en fremskrivning af data. I dette tilfælde har hverken fremskrivning eller værdien i sig selv udløst en alarmering.

Lægen vil efter egen påmindelse kunne kontrollere data fra patienten.

6.2 Scenarie 2 - Ofte dehydrerende patient

Opsætningen og læge besøg er som forgående scenarie. I dette arbejdes der dog med en patient, hvor man ved, at patienten lider af dehydrering og er derfor blevet udstyret med en sensor der måler væskemængde i huden samt fået udleveret en vægt. Patienten skal veje sig selv morgen og aften.
Længen har oprettet patienten med tre forskellige diagnosticeringer.
Én der analyserer på væskemængden, én der analyserer på patientens vægt og sidst en der benytter begge faktorer.

Der er også en gateway løsning, men vægten er tilsluttet systemet som en smart sensor og har derfor egen datakommunikation med nettet.

Dagene går og intet sker. Hverken algoritme, der undersøger hydreringen, eller vægten opfanger problemer.
Pludselig, en dag alarmeres patienten om, at systemet har analyseret, at patienten over kort tid har tabt sig en del og i sammenhæng med dette begyndt at dehydrere.
Patienten vælger at tage alarmeringen til eftertanke og drikker en del vand og systemet alarmerer ikke patienten efterfølgende.

Teknisk har algoritmen, der bruger begge faktorer samtidigt, undersøgt på hydreringen og ved hver hydrerings måling brugt de sidste fem vægt målinger og set der var et fald i vægt samtidigt med at patienten blev mere dehydreret.

6.3 Opsummering

Ovenstående scenarier er fiktivt opsatte men giver en ide om at systemet bør kunne realiseres og bør være lige til at benytte for både læger og patienter, lige meget om systemet bruges til alarmering eller blot til undersøgelse af sygdomsforløb.

Kapitel 7

Konklusion

Rutineundersøgelser på patienter med kroniske sygdomme er en stor tidssluger. Patienterne bruger tid på transport til og fra syge- eller lægehus, og både sundhedspersonale og patienter bruger tid på undersøgelserne. For at kunne opnå besparelser på netop dette, kan der med fordel indføres telemedicinske systemer, da disse vil kunne bruges til at indrapportere data fra patinet til læge uden at patienten behøver forlade sit hjem.

I projektet er givet et arkitekturdesign for en generel telemedicinsk arkitektur for systemer, der indrapporterer vitale værdier fra patienter, og som ud fra disse data kan diagnosticere og alarmere på forskellige niveauer.
Et telemedicinsk diagnosticeringssystem vil lette arbejdet mærkbart i landets lægehuse og på hospitaler, da patienten bærer al informationen på sig.
Al denne data kan, ved at blive indrapporteret automatisk, give besparelser for patienter og læger, der bruger anselig tid og penge på introduktion og transport, hvilket indirekte giver et produktionstab for landet, da patienter ikke bruger tiden funktionelt, men venter på transport og på at blive undersøgt på lægehuse og hospitaler.

Det er blevet påvist, hvordan et netværk af sensorer på patienten (WBAN) kan gøre denne del af den komplette arkitektur mulig.
Det findes muligt at få startet på prototype udvikling af et sådan projekt, og det anbefales at dele det op i to grupper.

Der er en opgave i at få lavet et „wireless body area network“ WBAN med en ønsket radiokommunikation, og her anbefales Bluetooth LE, på med en rækkevidde på over 20 meter er fuldt tilstrækkeligt til brug i et WBAN, godt batteriforbrug og ligger op til parring af enheder (gateway$<->$sensor). Gatewayen for WBANet kan laves som en applikation på f.eks en smartphone, der indeholder mange af de features man ønsker at benytte i en gateway. Dette kan sågar lade sig gøre regulatorisk, set ud fra det nuværende direktiv.

På serversiden, hvor data modtages fra gatewayen, er det sværere at lave en generel struktur, da den er meget afhængig af, hvad data skal bruges til. Hvis systemet skal være et alarmeringssystem, anses den bedste løsning for at være med parallel diagnosticering, da dette vil effektivisere svartider.

Systemet er tilgængeligt for fagrupper som forskere, specialister og læger, der kan koncentrere sig om egen part i projektet, og fokusere på deres egne områder for at bidrage til en samlet bedre løsning.
Dataindsamlingsdelen i sig selv bruges allerede i andre projekter som Datafangst, og vil i sig selv være en gevinst for lægeuniverset til sygdomslære[9].

Netop ved at indsamle data, kan der på sigt laves mere teknologiske og generelle sensorer med selvstændige algoritmer udvundet af projektets system, da der vil være store datamængder til at perfektionere og optimere algoritmerne.

Ved at lave datasamling, kan det muligøres at patienten selv får adgang til egen data og derved vil have et indblik i sin egen tilstand.

Der er stort potentiale indenfor medicinsk overvågning af patienter, som tager medicin regelmæssigt.
Ud fra vitale målinger er det muligt at overvåge om medicinindtaget foretages, som det er ordineret, og om medicinen har den ønskede optagelse i blodet, da dette ikke altid er sikkert ved f.eks skift til kopipræparater.

Det skal dog overvejes, om der skal indføres mere telemedicin så længe sundhedsvæsenet ikke kan blive enige om, hvordan de vil have det. Først skal der ryddes op i de nuværende systemer og opnåes enighed omkring, hvilke fremtidsvisioner der er og derefter kan der designes et telemedicinsk, der dækker tilfredsstillende. En implementering af et telemedicinsk system på tværs af firmaer og lande, eller bare regioner i Danmark, har derfor lange udsigter.

Når først enigheden er opnået, og visionen er fundet, vil telemedicinske systemer som denne, kunne vinde frem. Mulighederne er, som påvist, utallige og med store besparelser og gevinster for samfundet.

Kapitel 8

Fremtidige studier

De matematiske modeller til fremskrivning af data og derved alarmering skal undersøges nærmere. Dette gøre med fokus på muligheden for at udvikle algoritmer, der kan fremskrive en diagnose. ;en en start vil være at have algoritmer, når der afviges fra patientens normale tilstand.

Det forekommer naturligt at der skal implementeres en prototype, og så længe projektet er i en opstarts periode er det oplagt at bruge en smartphone som gateway, da det er algoritmerne, der er det væsentlige.

Man kan undersøge om projektet kan bruges til overvågning generelt; dette kan være til industrielle køleskabe eller lignende, hvor et firma kan oprettes om en patient og hvert køleskab er en sensor, algoritmen er forholdvis enkel (kan sammenlignes med en simpel diabetes). Netop ved industrielle køleskabe til f.eks blodopbevaring, er der et krav om at blodet aldrig må have været over eller under et specifik temperatur. Ved at systemet har indbygget et lag, der gemmer data, kan dette forholdvis nemt dokumenteres.

Det kan undersøges, hvor store ændringer der er påkrævet, hvis systemet også skal kunne håndtere folk der tager medicin regelmæssigt. Disse patienter skal meget tidsnøjagtigt tage deres medicin for at opretholde et givet niveau af medicin i kroppen. Det kan derfor være ideelt, hvis patienten tog en mindre dosis flere gange om dagen.

Umiddelbart ses udvidelsen i form af et feed-forward samt feed-backward system, ikke som en voldsom udvidelse, men det vil kunne åbne helt nye muligheder for medicinering og overvågning, forudsat at medicinen tages korrekt.

Kapitel 9

Litteraturliste

Litteratur

[1] airtightnetworks.com. 802.11n frequently asked questions. http://www.airtightnetworks.com/home/solutions/80211n/80211n-faqs.html, Feb 2012. Set den 18 Feb 2012.

[2] Continua Heath Alliance. Continua design guidelines. *None*, 2009.

[3] Continua Heath Alliance. Continua health alliance releases free design guidelines; gains android support. http://www.continuaalliance.org/static/cms_workspace/Continua_Free_Guidelines_Release_1.11.12_Continua_Health_Alliance.pdf, Jan-2012. Set den 24 Jan 2012.

[4] M. H. Bojsen. Secure linking of patients and tele-medical sensors. Master's thesis, Technical University of Denmark, DTU Informatics, E-mail: reception@imm.dtu.dk, Asmussens Alle, Building 305, DK-2800 Kgs. Lyngby, Denmark, 2011. Supervised by Associate Professor Christian Damsgaard Jensen, cdj@imm.dtu.dk, DTU Informatics.

[5] M. Bornhøft and M. Carvalho. Hierarkisk distribueret IDS ved anvendelse af neural net. Master's thesis, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, 2004. Supervised by Prof. Robin Sharp.

[6] Rob Brooks-Bilson. The evolution of integration architecture: An introduction to the enterprise service bus (esb). http://rob.brooks-bilson.com/index.cfm/2007/12/14/, Dec 2007. Set den 5 Januar 2012.

[7] connectblue.com. Oem bluetooth low energy platform module olp425. http://www.connectblue.com/products/bluetooth-products/bluetooth-modules/

bluetooth-low-energy-platform-module-olp425/. Set den 28 Feb 2012.

[8] dagensmedicin.dk. Når pacemakeren tager magten. http://www.dagensmedicin.dk/nyheder/nar-pacemakeren-tager-magten/. Set den 21 Feb 2012.

[9] dak e.dk. Datafangst. http://www.dak-e.dk/flx/datafangst/datafangst/. Set den 28 Januar 2012.

[10] Henrik Madsen Dimitri Boiroux & Daniel Aaron Finan & Niels Kjølstad Poulsen and John Bagterp Jørgensen. Insulin administration for people with type 1 diabetes. *21st European Symposium on Computer-Aided Process Engineering : Elsevier Science, 2011*, 2011.

[11] John Donovan. Bluetooth low-energy: An introduction. http://low-powerwireless.com/blog/2010/07/08/bluetooth-low-energy-an-introduction/, Jun 2010. Set den 18 Feb 2012.

[12] epilepsiforeningen.dk. Ofte stillede spørgsmål. http://www.epilepsiforeningen.dk/epilepsi/spoergsmaal-og-svar/. Set den 28 Feb 2012.

[13] Rådet for de Europæiske fællesskaber. RÅdets direktiv 93/42/eØf. http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CONSLEG:1993L0042:20071011:DA:PDF, Jun-1993 / Oct 2007. Set den 12 Dec 2012.

[14] Jean-Louis MarÈchaux for IBM. Combining service-oriented architecture and event-driven architecture using an enterprise service bus. http://www.immagic.com/eLibrary/ARCHIVES/GENERAL/IBM/I060328M.pdf, Marts 2006. Set den 4 Januar 2012.

[15] NYT fra Danmarks Statistik. Vi tjener 233 kr. pr. time. http://www.dst.dk/pukora/epub/Nyt/2011/NR631.pdf. Set den 22 Feb 2012.

[16] NYT fra Danmarks Statistik. En halv million flere ældre i 2044. http://www.dst.dk/pukora/epub/Nyt/2011/NR205.pdf, May-2011. Set den 12 Okt 2011.

[17] NYT fra Danmarks Statistik. Sygehusbenyttelse 2009. http://www.dst.dk/pukora/epub/Nyt/2011/NR481.pdf, Oct-2011. Set den 22 Nov 2011.

[18] P. Garbacki, D.H.J. Epema, and M. van Steen. Optimizing peer relationships in a super-peer network. In *Distributed Computing Systems, 2007. ICDCS '07. 27th International Conference on*, Jun 2007.

[19] Beverly Yang Hector & Garcia-Molina. Designing a super-peer network. *None*, Mar-2003.

[20] EUROPEAN COMMISSION DG HEALTH and CONSUMER. Medical devices: Guidance document. `http://ec.europa.eu/health/medical-devices/files/meddev/2_1_6_ol_en.pdf`, Jan-2012. Set den 12 Feb 2012.

[21] HP. Understanding bluetooth. `http://www.hp.com/rnd/library/pdf/understandingBluetooth.pdf`, Jan 2002. Set den 28 Feb 2012.

[22] http://www.delta.dk. Delta bidrager til national strategi for telemedicin. `http://www.delta.dk/dk/Forr-omr/TC/medicinsk-udstyr/udvalg-telemedicin-okt-2011.page`, Okt-2011. Set den 1 Dev 2011.

[23] ictalcare.dk. Epilepsi. `http://ictalcare.dk/investors/`. Set den 12 Jan 2012.

[24] jawbone.com. Jawbone era tech specs. `http://jawbone.com/headsets/era/specs`, Feb-2012. Set den 21 Feb 2012.

[25] John Padgette from National Institute of Standards Karen Scarfone and Technology. Guide to bluetooth security. `http://csrc.nist.gov/publications/nistpubs/800-121/SP800-121.pdf`, Sep 2008. Set den 15 Feb 2012.

[26] Jamil. Y. Khan and Mehmet R. Yuce. Wireless body area network (wban) for medical applications. `http://www.intechopen.com/download/pdf/pdfs_id/9103`, Jan-2010. Set den 12 Jan 2012.

[27] Patrick Kinney. Zigbee technology: Wireless control that simply works. `http://www.itk.ntnu.no/fag/TTK4545/TTK2/PDF/ZigBee%20Technology%20-%20Wireless%20control%20that%20simply%20works.pdf`, Oct 2003. Set den 29 Feb 2012.

[28] MedCom. Hvad er telemedicin? `http://www.medcom.dk/wm111694`, Feb-2011. Set den 28 Feb 2012.

[29] Uri Duvald Andersen & Jens Peter Andersen & Martin Stenfeldt Medicoinnovation. Medico apps - a practitioner's guide. Aricle, Jan-2012.

[30] msdn.microsoft.com. Installing a bluetooth device. `http://msdn.microsoft.com/en-us/library/windows/hardware/ff536681(v=vs.85).aspx`. Set den 21 Feb 2012.

[31] Dan Nosowitz. Everything you need to know about near field communication. `http://www.popsci.com/gadgets/article/2011-02/near-field-communication-helping-your-smartphone-replace-your-wallet-2010`, Feb 2011. Set den 1 marts 2012.

[32] Openwifi.dk. Openwifi.dk - største danske oversigt over wifi hotspots... http://openwifi.dk/, Feb-2011. Set den 17 Jan 2012.

[33] CHRIS OTTO, ALEKSANDAR MILENKOVIC, COREY SANDERS, and EMIL JOVANOV. System architecture of a wireless body area sensor network for ubiquitous health monitoring. *Journal of Mobile Multimedia, Vol. 1, No.4 (2006) 307-326*, Jan-2006.

[34] C. Patauner, H. Witschnig, D. Rinner, A. Maier, E. Merlin, and E. Leitgeb. High speed rfid/nfc at the frequency of 13.56 mhz. http://www.eurasip.org/Proceedings/Ext/RFID2007/pdf/s1p4.pdf, Feb 2012. Set den 12 Feb 2012.

[35] Retsinformation. Cirkulære om aktivitetsbestemt, kommunal medfinansiering på sundhedsområdet. https://www.retsinformation.dk/Forms/R0710.aspx?id=139670, Dec-2011. Set den 16 Jan 2012.

[36] Retsinformation. Lov om ændring af lov om regionernes finansiering. https://www.retsinformation.dk/forms/R0710.aspx?id=136462, Mar-2011. Set den 17 Jan 2012.

[37] F. Sposaro and G. Tyson. ifall: An android application for fall monitoring and response. In *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*, pages 6119 –6122, sept. 2009.

[38] Danmarks Statistik. 60 år i tal – danmark siden 2. verdenskrig. http://www.dst.dk/pukora/epub/upload/12433/tresaar.pdf, Dec-2008. Set den 1 Okt 2011.

[39] TDC.dk. Rækkevidde med trådløst netværk (wi-fi). http://kundeservice.tdc.dk/privat/faq.php?id=20574. Set den 22 Feb 2012.

[40] Teknologirådet. Telemedicin - en vej til et bedre sundhedsvæsen. http://www.tekno.dk/subpage.php3?article=99&toppic=kategori6&language=dk, 06-09-2007. Set den 12 Jan 2012.

[41] ug.dk. Sygeplejerske. http://www.ug.dk/job/sundhedomsorgogpleje/sygepljordemoderarb/sygeplejerske.aspx#fold4. Set den 25 Feb 2012.

[42] Sana Ullah, Pervez Khan, Niamat Ullah, Shahnaz Saleem, Henry Higgins, and Kyung Sup Kwak. A review of wireless body area networks for medical applications. *CoRR*, abs/1001.0831, 2010.

[43] visualdrg.im.dk. Visualdrg 2012. http://visualdrg.im.dk/Help.aspx?PageID=1. Set den 2 Feb 2012.

[44] Z-Wave.com. Technical questions. `http://www.z-wave.com/modules/xoopsfaq/index.php?cat_id=2`, 2011. Set den 29 Feb 2012.

[45] Zen-sys.com. Zensys launches the first rf single chip portfolio for ir replacement in av control. `http://www.zen-sys.com/modules/iaCM-ZW-PR/readMore.php?id=16777216`, Jan 2008. Set den 29 Feb 2012.

Bilag A

XML protokol

```
<?xml version="1.0"?>
<data>
  <dataPackageVersion>1.5</dataPackageVersion>
  <patientID>1337</patientID>
  <gatewayID>abc43-5614</gatewayID>
  <dataPackageSequenceNumer>5678</dataPackageSequenceNumer> <!--
      Auto increment -->
  <timestampSend>2007-11-09 T 11:20</timestampSend>
  <sequenceNumber>53</sequenceNumber>
  <sensors>
    <sensor>
      <sensorPackageVersion>3.4</sensorPackageVersion>
      <sensorProductType>blood sugar</sensorProductType>
      <sensorProductTypeID>455</sensorProductTypeID>
      <sensorVendor>Siemens</sensorVendor>
      <sensorVendorID>55</sensorVendorID>
      <sensorSerial></sensorSerial>
      <timestampFirst>2007-11-09 T 11:20</timestampFirst>
      <timestampLast>2007-11-09 T 11:25</timestampLast>
      <patientID>1337</patientID>
      <datapoints>
        <datapoint>
          <datapointPackageVersion>1.0<datapointPackageVersion>
          <value>27</value>
          <unit>g/mol</unit>
          <unitID>43</unitID>
          <timestamp>2007-11-09 T 11:20</timestamp>
        </datapoint>
```

```
28          <datapoint>
29            <datapointPackageVersion>1.0<datapointPackageVersion>
30            <value>27</value>
31            <unit>g/mol</unit>
32            <unitID>43</unitID>
33            <timestamp>2007-11-09 T 11:20</timestamp>
34          </datapoint>
35        </datapoints>
36      </sensor>
37      <sensor>
38        <sensorPackageVersion>3.4</sensorPackageVersion>
39        <sensorProductType>blood sugar</sensorProductType>
40        <sensorProductTypeID>455</sensorProductTypeID>
41        <sensorVendor>Siemens</sensorVendor>
42        <sensorVendorID>55</sensorVendorID>
43        <sensorSerial></sensorSerial>
44        <timestampFirst>2007-11-09 T 11:20</timestampFirst>
45        <timestampLast>2007-11-09 T 11:25</timestampLast>
46        <patientID>1337</patientID>
47        <datapoints>
48          <datapoint>
49            <datapointPackageVersion>1.0<datapointPackageVersion>
50            <value>27</value>
51            <unit>g/mol</unit>
52            <unitID>43</unitID>
53            <timestamp>2007-11-09 T 11:20</timestamp>
54          </datapoint>
55          <datapoint>
56            <datapointPackageVersion>1.0<datapointPackageVersion>
57            <value>27</value>
58            <unit>g/mol</unit>
59            <unitID>43</unitID>
60            <timestamp>2007-11-09 T 11:20</timestamp>
61          </datapoint>
62        </datapoints>
63      </sensor>
64    </sensors>
65  </data>
```

Listing A.1: *XML fil der symboliserer en sending fra gatewayen, og indeholder to sensorpakker.*

Bilag B

VDM design

B.1 VDM

Vienna Development Method, også kaldet VDM, dækker over formelle dialekter til matematisk analyse af en model.
VDM benyttes ofte med pre og post condition, og er god til at analysere og afprøve en kravspecifikation, da VDM sproget er på et højt abstraktionsniveau, hvilket gør det muligt kun at gå i de detaljer der er vigtige for arkitekturen. Modellen kan udbygges ned til mindste detalje og gør det muligt at køre koden.

VDM er et god metode til brug før et evt. større projekt sættes igang, dan man kan bevise funktionalitet uden at implementere et helt system.

B.1.1 Pseudo Model opbygning med typer og nogle funktioner

I nedenstående ses de to modeller der blev lavet i VDM.

Først ses modellerne.

```
class Server
types
public ChannelLongDistance = <GSM> | <WiFi>;--??

operations

public recieve : set of Gateway`Measurement ==>()
recieve(measurements)==
for all m in set measurements do
cases m.type:
  <BloodPresure>-> (IO`println("Recieved␣blood␣preasure␣value:␣");
      IO`println(m)),
  <Accelerometer>-> (IO`println("Recieved␣Accelerometer␣value:␣");
      IO`println(m))
--  patternList2 -> expression2,
--  others -> expressionOthers
end;

end Server

class Gateway
types
String = seq of char;
public Value = int;
public SensorType = <BloodPresure> | <Accelerometer>;
public Time = int;
public Measurement ::
  timeStamp   : Time
  type    : SensorType
  val     : Value;

public Channel ::
  type : <ZigBee> | <Zwave>;--??

public Sensor ::
  val     : Value
  type    : SensorType
  manufactor  : String;

instance variables
  sensors   : map Channel to set of Sensor;
  comChannels : map Server`ChannelLongDistance to Server;
  cache   : set of Measurement := {};

operations

public Gateway : map Channel to set of Sensor * map Server`
    ChannelLongDistance to Server ==> Gateway
Gateway(s,c) ==
(
  sensors := s;
  comChannels := c;
);
```

```
51
52 public readSensors : () ==> ()
53 readSensors() ==
54   --BUG cache := cache union {mk_(1,s.val , s.type) |s in set
         dunion rng sensors};
55   cache := cache union {mk_Measurement(1, s.type, s.val) |s in set
         dunion rng sensors};
56
57 public send : () ==>()
58 send()==
59 (
60   --maybe only take a chunk of the cache based on the channel type
61   for all s in set rng comChannels do
62     s.recieve(cache);
63   cache := {};
64 )
65 pre cache <> {};
66
67
68 end Gateway
```

Listing B.1: *Model.vdmpp*

Her ses systemet som kan køres som var det et normalt program.

```
1  class System
2  values
3
4  s1 : Gateway`Sensor = mk_Gateway`Sensor(1,<BloodPresure>,"Siemens")
       ;
5  s2 : Gateway`Sensor = mk_Gateway`Sensor(1,<Accelerometer>,"Siemens"
       );
6
7  gw : Gateway = new Gateway( {mk_Gateway`Channel(<ZigBee>) |-> {s1,
       s2}},
8                {<GSM> |-> new Server()});
9
10 operations
11
12 public run : () ==> ()
13 run()==
14 (
15 --update sensor...
16
17
18 --gateway reads
19 gw.send();
20 gw.readSensors();
21 gw.send();
22
23
24
25 );
```

```
26
27 end System
```

Listing B.2: *System.vdmpp*

Bilag C

Android Accelerometer

```
1  package pi.acc;
2  import android.app.Activity;
3  import android.content.pm.ActivityInfo;
4  import android.content.res.Configuration;
5  import android.hardware.Sensor;
6  import android.hardware.SensorEvent;
7  import android.hardware.SensorEventListener;
8  import android.hardware.SensorManager;
9  import android.os.Bundle;
10 import android.util.Log;
11 import android.widget.TextView;
12
13 public class ACCActivity extends Activity implements
       SensorEventListener {
14
15   private TextView result;
16   private TextView xlabel;
17   private TextView ylabel;
18   private TextView zlabel;
19
20   private TextView packagenumberlabel;
21   private SensorManager sensorManager;
22   private Sensor sensor;
23   private long lastCurrentMillis;
24   private DataCollector dataCollector = null;
25
26     public ACCActivity()
27     {
28
```

```
29      }
30
31      @Override
32      public void onCreate(Bundle savedInstanceState) {
33          super.onCreate(savedInstanceState);
34          setContentView(R.layout.main);
35          setRequestedOrientation (ActivityInfo.
                SCREEN_ORIENTATION_PORTRAIT);
36          InitializeUI();
37
38          sensorManager = (SensorManager)getSystemService(
                SENSOR_SERVICE);
39          sensor = sensorManager.getDefaultSensor(Sensor.
                TYPE_ACCELEROMETER);
40          sensorManager.registerListener(this, sensor, SensorManager.
                SENSOR_DELAY_FASTEST);
41          dataCollector = new DataCollector();
42          Log.e("log_on....", "Create");
43      }
44
45      private void InitializeUI()
46      {
47         result = (TextView) findViewById(R.id.result);
48           xlabel = (TextView) findViewById(R.id.xlabel);
49           ylabel = (TextView) findViewById(R.id.ylabel);
50           zlabel = (TextView) findViewById(R.id.zlabel);
51           packagenumberlabel = (TextView) findViewById(R.id.
                 packagenumberlabel);
52           result.setText(""+lastCurrentMillis);
53  }
54
55  @Override
56  protected void onResume() {
57    super.onResume();
58    sensorManager.registerListener(this, sensor, SensorManager.
          SENSOR_DELAY_FASTEST);
59    Log.e("log_on....", "Resume:");
60  }
61
62  @Override
63  protected void onPause() {
64    super.onPause();
65    sensorManager.unregisterListener(this);
66    Log.e("log_on....", "Pause:");
67  }
68  @Override
69  protected void onStart()
70  {
71    super.onStart();
72    sensorManager.registerListener(this, sensor, SensorManager.
          SENSOR_DELAY_FASTEST);
73    Log.e("log_on....", "Start");
74  }
75
76  @Override
```

```
77    public void onBackPressed()
78    {
79      super.onBackPressed();
80      sensorManager.unregisterListener(this);
81      Log.e("log_on....", "BackPressed");
82    }
83
84    protected void onStop()
85    {
86      super.onStop();
87      sensorManager.unregisterListener(this);
88      Log.e("log_on....", "Stop");
89    }
90
91    @Override
92    public void onConfigurationChanged(Configuration newConfig)
93    {
94      super.onConfigurationChanged(newConfig);
95        setContentView(R.layout.main);
96
97        InitializeUI();
98        Log.e("log_on....", "ConfigurationChanged:");
99    }
100
101   private void refreshDisplay(float[] values)
102   {
103     result.setText("Milis␣from␣last␣update:␣"+(System.
          currentTimeMillis()-lastCurrentMillis));
104     lastCurrentMillis = System.currentTimeMillis();
105     xlabel.setText("x:␣"+Math.abs(values[0]));
106     ylabel.setText("y:␣"+Math.abs(values[1]));
107     zlabel.setText("z:␣"+Math.abs(values[2]));
108     packagenumberlabel.setText("package:␣"+ dataCollector.
          getPackageNumber() + "␣dataelement:␣"+ dataCollector.
          getDataElement());
109
110     dataCollector.collect(values);
111   }
112
113   @Override
114   public void onAccuracyChanged(Sensor sensor, int acc)
115   {
116     result.setText("accurancy␣changed␣"+System.currentTimeMillis())
          ;
117   }
118
119   @Override
120   public void onSensorChanged(SensorEvent event)  {
121
122     refreshDisplay(event.values);
123   }
124 }
```

Listing C.1: *ACCActivity.java*

```
1  package pi.acc;
2
3  public class Acceleration
4  {
5          private float X;
6          private float Y;
7          private float Z;
8      public float getX()
9      {
10       return X;
11     }
12     public void setX(float x) {
13       X = x;
14     }
15     public float getY() {
16       return Y;
17     }
18     public void setY(float y) {
19       Y = y;
20     }
21     public float getZ() {
22       return Z;
23     }
24     public void setZ(float z) {
25       Z = z;
26     }
27 }
```

Listing C.2: *Acceleration.java*

```
1  package pi.acc;
2
3  import java.io.BufferedReader;
4  import java.io.IOException;
5  import java.io.InputStream;
6  import java.io.InputStreamReader;
7  import java.io.Reader;
8  import java.io.StringWriter;
9  import java.io.UnsupportedEncodingException;
10 import java.io.Writer;
11 import java.util.ArrayList;
12 import java.util.List;
13 import org.apache.http.HttpEntity;
14 import org.apache.http.HttpResponse;
15 import org.apache.http.NameValuePair;
16 import org.apache.http.client.ClientProtocolException;
17 import org.apache.http.client.HttpClient;
18 import org.apache.http.client.entity.UrlEncodedFormEntity;
19 import org.apache.http.client.methods.HttpPost;
20 import org.apache.http.entity.ByteArrayEntity;
21 import org.apache.http.impl.client.DefaultHttpClient;
22 import org.apache.http.message.BasicNameValuePair;
23 import org.apache.http.params.BasicHttpParams;
24 import org.apache.http.params.HttpConnectionParams;
```

```
25 import org.apache.http.params.HttpParams;
26 import org.apache.http.protocol.HTTP;
27 import org.json.JSONException;
28 import android.util.Log;
29 import com.google.gson.Gson;
30 import com.google.gson.JsonArray;
31
32 public class DataSender  extends Thread
33 {
34    HttpClient httpclient;
35
36   public static void sendData(List<Report> Data)
37   {
38     MyThread senderThread = new MyThread(Data);
39       senderThread.start();
40   }
41
42 static public class MyThread extends Thread {
43
44     private List<Report> data;
45
46     public MyThread(List<Report> data)
47     {
48       this.data = data;
49     }
50
51     public void run() {
52       try
53         {
54         sendDataToDotNetServer(data);
55         }
56         catch(Exception e)
57         {
58             Log.e("log_tag", "Error:␣␣"+e.toString()+"␣"+e.
                   getStackTrace());
59         }
60       try
61         {
62         sendDataToPHPServer(data);
63         }
64         catch(Exception e)
65         {
66             Log.e("log_tag", "Error:␣␣"+e.toString()+"␣"+e.
                   getStackTrace());
67         }
68     }
69   }
70
71   public static void sendDataToPHPServer(List<Report> data) throws
         JSONException, ClientProtocolException, IOException
72   {
73         String urlServer = "http://www.tweak.dk/axeltest/
               AddAccelerationData.php";
74             DefaultHttpClient httpClient = new DefaultHttpClient();
75             HttpPost postMethod = new HttpPost(urlServer);
```

```
76              List<NameValuePair> nameValuePairs = new ArrayList<
                    NameValuePair>(2);
77
78              Gson gson = new Gson();
79          JsonArray ja = new JsonArray();
80              for (Report value : data)
81              {
82            ja.add(gson.toJsonTree(value));
83          }
84
85              nameValuePairs.add(new BasicNameValuePair("jsonString",
                     ja.toString()));
86              postMethod.setEntity(new UrlEncodedFormEntity(
                    nameValuePairs));
87
88              HttpResponse response = httpClient.execute(postMethod);
89              Log.i("postData", response.getStatusLine().toString());
90      }
91
92  public static void sendDataToDotNetServer(List<Report> data)
        throws JSONException, ClientProtocolException, IOException
93  {
94
95    String urlServer = "http://193.163.101.88:8081/json/postReports
          ";
96
97    HttpParams httpParams = new BasicHttpParams();
98    HttpConnectionParams.setConnectionTimeout(httpParams, 10000);
99    HttpConnectionParams.setSoTimeout(httpParams, 10000);
100   HttpClient client = new DefaultHttpClient(httpParams);
101
102   HttpPost request = new HttpPost(urlServer);
103
104   Gson gson = new Gson();
105   JsonArray ja = new JsonArray();
106       for (Report value : data)
107       {
108     ja.add(gson.toJsonTree(value));   }
109
110       ByteArrayEntity entity=  new ByteArrayEntity(ja.toString().
              getBytes("UTF8"));
111
112   entity.setContentEncoding(HTTP.UTF_8);
113     entity.setContentType("application/json");
114   request.setEntity(entity);
115
116   HttpResponse response = client.execute(request);
117   HttpEntity entityResponse = response.getEntity();
118   if (entityResponse != null)
119   {
120     InputStream instream = entityResponse.getContent();
121     String result = convertStreamToString(instream);
122     Log.i("Content␣from␣server", result);
123     Log.i("Read␣from␣server", response.getStatusLine().toString()
          );
```

```
124     }
125   }
126
127   private static String convertStreamToString(InputStream instream)
128   {
129     if (instream != null)
130     {
131       Writer writer = new StringWriter();
132       char[] buffer = new char[1024];
133
134       try
135       {
136         Reader reader = new BufferedReader(new InputStreamReader(
              instream, "UTF-8"));
137         int n;
138         while ((n = reader.read(buffer)) != -1)
139         {
140           writer.write(buffer, 0, n);
141         }
142       } catch (UnsupportedEncodingException e)
143       {
144         e.printStackTrace();
145       } catch (IOException e)
146       {
147         e.printStackTrace();
148       }
149       finally {
150         try
151         {
152           instream.close();
153         } catch (IOException e) {
154           e.printStackTrace();
155         }
156       }
157       return writer.toString();
158     }
159     else
160     {
161       return "NO␣STRING!";
162     }
163   }
164 }
```

Listing C.3: *DataSender.java*

```
1  package pi.acc;
2
3  import java.util.LinkedList;
4  import java.util.List;
5
6  public class DataCollector
7  {
8    private final int ELEMENTPERPACKAGE = 1000;
9    private int numberOfDataElementsSend = 0;
10   private  int packageNumber = 0;
```

```
11    private int dataElement = 0;
12    private List<Report> Data;
13
14    public DataCollector()
15    {
16      Data = new LinkedList<Report>();
17    }
18
19    public  int getPackageNumber()
20    {
21      return this.packageNumber;
22    }
23    public  int getDataElement()
24    {
25      return this.dataElement;
26    }
27
28    public void collect(float[] values)
29    {
30      if(numberOfDataElementsSend < ELEMENTPERPACKAGE)
31      {
32          Report report = new Report();
33          report.setDataElement(dataElement);
34          report.setDataPackage(packageNumber);
35
36          report.setUserid(42);
37          Acceleration acc = new Acceleration();
38          acc.setX(values[0]);
39          acc.setY(values[1]);
40          acc.setZ(values[2]);
41          report.setAcceleration(acc);
42
43          Data.add(report);
44
45          dataElement++;
46          numberOfDataElementsSend++;
47      }
48      else if (numberOfDataElementsSend == ELEMENTPERPACKAGE)
49      {
50        numberOfDataElementsSend = 0;
51        dataElement = 0;
52
53        DataSender.sendData(Data);
54        Data = new LinkedList<Report>();
55        packageNumber++;
56      }
57    }
58  }
```

Listing C.4: *DataCollector.java*

```
1  package pi.acc;
2
3
4  public class Report
```

```
5  {
6
7          private int Userid;
8          private int DataPackage;
9          private int DataElement;
10         private Acceleration Acceleration;
11
12     public int getUserid() {
13       return Userid;
14     }
15
16     public void setUserid(int userid) {
17       Userid = userid;
18     }
19
20     public int getDataPackage() {
21       return DataPackage;
22     }
23
24     public void setDataPackage(int dataPackage) {
25       DataPackage = dataPackage;
26     }
27
28     public int getDataElement() {
29       return DataElement;
30     }
31
32     public void setDataElement(int dataElement) {
33       DataElement = dataElement;
34     }
35
36     public Acceleration getAcceleration() {
37       return Acceleration;
38     }
39
40     public void setAcceleration(Acceleration acceleration) {
41       Acceleration = acceleration;
42     }
43 }
```

Listing C.5: *Report.java*