NTNU
Det skapende universitet

Fingerprintløsning for lokalisering ved bruk av WLAN

Gerhard Christoffer Wiese

Master i kommunikasjonsteknologi
Oppgaven levert: Juni 2010
Hovedveileder: Steinar Andresen, ITEM
Biveileder(e): Thomas Jelle, ITEM

Norges teknisk-naturvitenskapelige universitet
Institutt for telematikk

Oppgavetekst

Oppgaven består i å designe, implementere og teste en nettsentrert posisjoneringsløsning for WLAN. Løsningen skal basere seg på innsamling av posisjonsdata fra netteier og innsamlet erfaringsdata. Løsningen skal kunne brukes i ulike WLAN nett dvs være dynamisk mhp muligheten for å benytte forskjellige kart. Posisjon skal kunne presenteres til ulike terminaler, spesielt mobile terminaler, uten vesentlig forringelse i presisjon. Posisjon skal presenteres til bruker/ terminal i form av koordinater og i kartgrensesnitt.

Basert på tidligere erfaringer og lignende arbeid er følgende hovedutfordringer identifisert og så langt uløst:
- Ulike mobile terminaler har forskjellige antenner, og vil oppfatte signalstyrker fra omliggende trådløs infrastruktur forskjellig. Dette skaper utfordringer for et system basert på fingerprinting, og må håndteres for ikke å forringe nøyaktigheten til posisjoneringen.
- Innhenting av signalstyrker fra infrastrukturen er identifisert som en flaskehals for å kunne implementere anvendbare applikasjoner. Det må derfor etableres metoder for å optimalisere tiden det tar å gjennomføre lokasjonsoppslag.
- I enkelte omgivelser er det vanskelig å posisjonere brukeren nøyaktig og det oppstår rom- og etasjefeil. Det må derfor utvikles metoder for å håndtere slike feil. Det er ønskelig at systemet skal kunne oppgi riktig rom og etasje selv for områder med minimal fysisk adskillelse.
Dette vil være hovedutfordringene som skal adresseres i oppgaven.

Oppgaven gitt: 21. januar 2010
Hovedveileder: Steinar Andresen, ITEM

Sammendrag

I forbindelse med denne masteroppgaven har det blitt designet, implementert og testet en nettsentrert lokaliseringsløsning for WLAN. Løsningen baserer seg på innsamling av lokasjonsdata i form av lokasjonsfingeravtrykk. Ved å benytte lokasjonsfingeravtrykk har det blitt oppnådd en universell lokaliseringsløsning som vil kunne fungere i alle trådløse nettverk, og som baserer seg på allerede eksisterende trådløs infrastruktur.

I tidligere forskning på lokaliseringsløsninger basert på fingeravtrykk, har det blitt tatt i bruk relativt begrensede dekningsområder. Samtidig har innsamlingen av lokasjonsdata og testingen av lokaliseringsløsningen blitt gjennomført med den samme mobile enheten. Det har heller ikke blitt fokusert på tiden lokasjonssystemet behøver for å gjennomføre et lokasjonsestimat. Lokaliseringsløsningen utviklet i forbindelse med denne masteroppgaven skiller seg fra tidligere forskning ved at det er studert et mer helhetlig lokasjonssystem. Det har blitt identifisert hvordan en slik lokaliseringsløsning vil fungere i et omfattende dekningsområde, og det har blitt tilrettelagt for bruk av ulike mobile enheter. Samtidig har det blitt fokusert på hvordan en bruker av lokasjonstjenesten vil oppfatte systemets ytelse. Det har derfor blitt vektlagt at et lokasjonsestimat skal gjennomføres så raskt som mulig.

Hensikten med oppgaven har vært å undersøke hvorvidt en slik lokaliseringsløsning vil være en fornuftig måte å implementere et innendørs lokasjonssystem i store bygninger, slik som ved NTNU campus. Lokasjonsløsningen har et omfattende dekningsområde i EL-bygget ved NTNU Gløshaugen på 856 kvadratmeter, fordelt over to etasjer.

Kjernen av lokaliseringsløsningen er en server som estimerer lokasjoner basert på lokasjonsdata mottatt fra klienter. For klientene er lokaliseringsløsningen utelukkende programvare basert. Alle klienter med WLAN-støtte, og mulighet for å kjøre programvare, vil kunne benytte seg av lokasjonstjenesten. På denne måten har det blitt skapt et lokasjonssystem som vil fungere for ulike mobile enheter, uavhengig av underliggende hardware og plattform. Dette har blitt bevist ved at det er implementert klientløsninger for to ulike smarttelefon-plattformer, iPhone OS og Android.

Ulike mobile enheter vil oppfatte signalstyrker fra WLAN-aksesspunkt forskjellig for samme lokasjon. Dette gjør det til en krevende oppgave for lokasjonsserveren å estimere riktig lokasjon, uavhengig av hva slags mobil enhet som benyttes. Euklids avstand har tidligere blitt ansett som en svært egnet lokasjonsbestemmende algoritme i lokaliseringsløsninger basert på fingerav-

trykk. I denne oppgaven har det blitt bevist at når lokaliseringsløsningen skal gi støtte for bruk av ulike mobile enheter, gir Euklids avstand ikke lenger tilfredsstillende resultater med hensyn på nøyaktighet. For å øke nøyaktigheten av lokasjonssystemet har det derfor blitt utviklet en alternativ lokasjonsbestemmende algoritme. Denne algoritmen benytter et vektingssystem, basert på forholdet mellom mottatt signalstyrke fra ulike aksesspunkt, for å estimere klientenes lokasjon.

I etterkant av implementasjonen har lokaliseringsløsningen blitt testet for å identifisere ytelsen med hensyn nøyaktighet og tidsbruk. Testene ble utført ved hjelp av en iPhone og den Android-baserte HTC Hero. Den gjennomsnittlige tiden det tar å estimere en lokasjon for Android-telefonen er 1104 millisekunder, mens en det i gjennomsnitt tar 1343 millisekunder å estimere en lokasjon for iPhone.

Nøyaktigheten av et lokasjonsestimat har blitt funnet ved å måle avvik i luftlinje mellom estimert og reell lokasjon. Det gjennomsnittlige avviket mellom estimert og reell lokasjon ble for iPhonen funnet til å være 3,46 meter, mens for Android-telefonen var dette 4,62 meter. Selv om vektingsalgoritmen økte lokasjonssystemets ytelse med hensyn på nøyaktighet, anses nøyaktigheten fortsatt ikke som tilfredsstillende. Det har derfor blitt forsøkt å identifisere en måte en vil kunne øke nøyaktigheten av lokaliseringsløsningen ytterligere. Det vil oppleves en større fluktuering i mottatt signalstyrke for sterke signaler, sammenliknet med svake signaler. Dette tilsier at det er mer egnet å benytte svake signalstyrker i en lokaliseringsløsning basert på fingeravtrykk. Det har derfor blitt forsøkt å identifisere et intervall for signalstyrker som vil være ideelle å benytte i en lokaliseringsløsning. Intervallet ble identifisert til å strekke seg fra -75 dBm til -65 dBm. Ved å benytte flere fingeravtrykk for å definere en lokasjon i serverens database, vil det være flere signalstyrkeforhold å sammenlikne for vektingsalgoritmen. Det antas derfor at ved å øke antallet fingeravtrykk, og samtidig danne fingeravtrykk ut i fra signalstyrker som befinner seg innenfor det identifiserte signalstyrkeintervallet, vil en kunne øke nøyaktigheten av lokaliseringsløsningen.

Innhold

Figurer

Tabeller

Kapittel 1

Introduksjon

1.1 Motivasjon

Kontekstsensitive tjenester er tjenester som tilpasser seg brukerens omgivelser. Omgivelser kan for eksempel være temperatur, bevegelse, lys eller lokasjon [3]. Det finnes en rekke artikler som presiserer at dette er tjenester for fremtiden [3, 5, 17], allikevel finnes det et begrenset antall kommersielle realiseringer av slike tjenester. En viktig faktor for å bestemme brukerens omgivelser, er brukerens lokasjon. Tjenester som tilpasses i forhold til brukerens lokasjon kalles lokasjonsbaserte tjenester. Om programvare har mulighet til å vite brukerens lokasjon, vil det åpne nye muligheter for hva programvaren vil kunne utrette.

Det finnes flere forskjellige implementasjoner av lokasjonsbestemmende systemer [5, 10]. Utendørs er GPS allerede en velutviklet og godt testet løsning, men GPS har allikevel en essensiell begrensning i at det sjeldent vil kunne fungere innendørs [16]. Samtidig dukker det stadig opp flere trådløse nettverk med vel utbygget infrastruktur. Ved å benytte trådløse nettverk til å bestemme brukerens lokasjon vil en kunne skape et lokasjonssystem som vil fungere både innendørs og utendørs.

Det finnes ulike tilnærminger til hvordan en kan benytte trådløse nettverk for å bestemme brukerens lokasjon. Det er utviklet servere samt annen hardware netteier kan implementere i deres nettverk for å oppnå en lokaliseringsløsning. Slik hardware er ofte proprietær og svært kostbar. Blant annet har Cisco og Radionor utviklet slike løsninger (se avsnitt 1.5.3 og 1.5.4).

Denne masteroppgaven er en fortsettelse på et fordypningsprosjekt utført

høsten 2009 [32], og vil undersøke om det er mulig å utvikle en innendørs lokaliseringsløsning som vil fungere i alle trådløse nettverk. Dette er gjort ved å implementere en prototype på et lokaliseringssystem som tar i bruk en teknikk kalt lokasjonsfingeravtrykk. Løsningen vil basere seg på allerede eksisterende trådløs infrastruktur, og kreve minimalt med ekstra hardware. Samtidig vil det være mulig å benytte seg av lokasjonssystemet ved hjelp av ulike mobile terminaler, med fokus på smarttelefoner.

Det har tidligere blitt forsket og eksperimentert på lokasjonssystemer basert på lokasjonsfingeravtrykk. Prototypen utviklet i forbindelse med denne masteroppgaven skiller seg fra annen forskning på følgene punkter:

1. Systemet har et dekningsområde som er representativt for store bygninger slik som ved campus på NTNU Gløshaugen. Dette er et vesentlig større dekningsområde enn tatt i bruk i annen forskning.
2. Systemet støtter bruk av forskjellige mobile terminaler. Det er ikke funnet annen forskning hvor dette er gjennomført.
3. Hvor raskt systemet vil kunne gjennomføre et lokasjonsestimat vil være en viktig faktor for brukernes opplevelse av systemets ytelse. Under implementasjonen av lokasjonssystemet har det blitt fokusert på at et lokasjonsestimat skal bli gjennomført så raskt som mulig. Annen forskning har ikke sett på dette aspektet ved lokaliseringsløsninger basert på fingeravtrykk.
4. Systemets ytelse med hensyn på gjengivelse av den reelle lokasjonen vil, for en bruker, avhenge av andre faktorer i tillegg til avstand fra reell til estimert lokasjon. For eksempel vil det være viktig at systemet estimerer brukerens lokasjon til riktig rom. Slike utfordringer er adressert i denne oppgaven.

Hvordan prototypen utviklet i forbindelse med denne masteroppgaven skiller seg fra annen forskning er grundigere beskrevet i avsnitt 1.6.3. Dette avsnittet vil også vise til annen forskning, og dermed begrunne relevansen av dette arbeidet.

1.2 Begrepsbruk

Dette avsnittet vil kort forklare sentrale begreper, nødvendige for å forstå den resterende delen av rapporten.

1.2.1 Posisjonering eller lokalisering

Det å posisjonere noe betyr å gi noe en posisjon, det vil si å plassere noe et sted. Samtidig kan posisjonering også bety å finne posisjonen til noe, dette er overlappende med utrykket lokalisering. Denne oppgaven beskriver hvordan en kan lokalisere mobile enheter (jf. avsnitt 1.2.4) ved hjelp av trådløse nettverk. Begge begrepene vil derfor i denne oppgaven beskrive det å finne lokasjonen til noe.

1.2.2 WLAN eller Wi-Fi

Begrepene WLAN og Wi-Fi blir vanligvis brukt om hverandre. Det ligger dermed et behov for å klargjøre disse begrepene slik at det kommer tydelig frem hva som menes. WLAN er en IEEE industristandard, og refererer dermed til selve teknologien som benyttes i trådløs kommunikasjon. Wi-Fi er et varemerke tilhørende The Wi-Fi Alliance som baserer seg på WLAN teknologien. Det at det er et varemerke innbærer blant annet at leverandørene garanterer av Wi-Fi produkter kan kommunisere med hverandre[1]. I denne oppgaven vil det utelukkende diskuteres rundt selve teknologien, dermed vil utstyr brukt til trådløs kommunikasjon omtales som WLAN utstyr, selv om det også vil kunne kategoriseres som Wi-Fi.

1.2.3 Lokasjonsfingeravtrykk

Lokasjonsfingeravtrykk refererer til det engelske begrepet ”location fingerprints”. Et lokasjonsfingeravtrykk er et sett med målbare verdier som kan benyttes til å identifisere en lokasjon. Utrykket lokasjonsfingeravtrykk benyttes også ofte om en lokaliseringsteknikk som forbinder fysiske lokasjoner til lokasjonsdata mottatt fra radiosignaler. Lokasjonsdata er beskrevet i avsnitt 1.2.9.

[1] http://www.wi-fi.org/discover_and_learn.php, 5. mai 2010

1.2.4 Mobil enhet

Med en mobil enhet menes i denne oppgaven en bærbar enhet med sender og mottaker for radio signaler. Utrykket brukes gjerne om enheter som vil kunne benytte seg av omliggende trådløse nettverk mens de er bevegelse, slik som bærbare datamaskiner, mobiltelefoner, PDAer og liknende. Begrepet mobil terminal har en tilsvarende betydning som mobil enhet.

1.2.5 Referansemaskin

Begrepet referansemaskin vil i denne oppgaven tas i bruk for å beskrive den konkrete mobile enheten benyttet under innsamlingen av lokasjonsfingeravtrykk.

1.2.6 Rom- og etasjefeil

Rom- og etasjefeil beskriver to typer feil som kan oppstå når et lokasjonssystem estimerer en brukers lokasjon i en innendørs omgivelse.

- **Romfeil:** Beskriver et lokasjonsestimat der brukeren får oppgitt sin lokasjon i et rom hun eller han ikke befinner seg i.
- **Etasjefeil:** Beskriver et lokasjonsestimat der brukeren får oppgitt sin lokasjon i en etasje hun eller han ikke befinner seg i.

1.2.7 Klientløsning og serverløsning

Uttrykket klientløsning vil i denne oppgaven vise til programvare på de mobile enhetene som benytter seg av lokasjonssystemet. Uttrykket serverløsning refererer til programvaren og databasen serveren benytter for å estimere brukernes lokasjoner. Klientløsninger vil kommunisere med serverløsningen.

1.2.8 Funksjonell komponent

Begrepet funksjonell komponent brukes om et utsnitt av lokasjonssystemets programvare som inneholder en spesifikk funksjonalitet. Dette kan for eksempel være en klasse, eller en sentral metode innenfor en klasse. Systemets

helhetlige virkemåte er et resultat av en sammensetning av flere funksjonelle komponenter. Det er fokusert på å bygge opp lokasjonssystemet ut i fra funksjonelle komponenter, da dette vil lede til at det er enklere å gjenbruke deler av systemet i nye løsninger.

1.2.9 Lokasjonsdata

Lokasjonsdata vil si data en kan registrere fra omliggende trådløse nettverk. Dette er gjerne: Nettverk identifisert ved et navn, aksesspunkt identifisert ved en unik adresse innefor et nettverk, mottatt signalstyrke fra aksesspunkt og hvilken kanal aksesspunkt benytter.

1.3 Bruksområder

Dette avsnittet vil presentere noen eksempler på lokasjonsbaserte tjenester. Hensikten er ikke å liste opp tidligere arbeid, men å belyse hvorfor et innendørs lokasjonssystem er av stor nytte.

1.3.1 Navigasjon

Mange biler kommer i dag levert med integrert GPS-utstyr slik at brukeren lettere skal kunne navigere seg rundt på veiene, eventuelt kan en kjøpe frittstående GPS-tilleggsutstyr. Etter GPS for bil ble lansert, har flere og flere sett nytten av et slikt produkt, noe som har resultert i en vidstrakt bruk. Det er ikke bare utendørs en kan dra nytte av et navigasjonssystem. En kan se for seg et senario der en befinner seg på en stor ukjent flyplass, kanskje har en dårlig tid mellom to avganger. I et slik senario vil det kunne være hensiktsmessig å ha en tjeneste, for eksempel på mobiltelefonen, som forklarte korteste vei til gaten basert på brukerens lokasjon. En slik tjeneste vil også kunne oppgi annen lokasjonsbestemt informasjon, som hvor er nærmeste toalett eller kiosk.

1.3.2 Lokalisering av personell og utstyr

Det har vært mye forskning på hvordan lokasjonssystemer vil kunne øke arbeidseffektiviteten samt sikkerheten ved sykehus [23, 8, 31]. Medisinsk utstyr

er ofte svært kostbart og det er derfor gjerne et lite antall av en type utstyr, sett i sammenheng med størrelsen av sykehuset. Et lokasjonssystem vil kunne korte ned tiden nødvendig for å identifisere hvor utstyr befinner seg, og dermed effektivisere sykehuspersonellets arbeid ved at mindre tid går med til å lete etter utstyr. Om lokasjonssystemet også identifiserer hvor sykehuspersonell befinner seg vil en kunne avgjøre om utstyr er i bruk. Hvis medisinsk utstyr er lokalisert i nærheten av for eksempel en lege, på et rom hvor det befinner seg pasienter, er det naturlig å konkludere med at legen benytter dette utstyret. I et nødstilfellet vil et slikt lokasjonssystem også kunne øke effektiviteten og dermed sikkerheten ved sykehuset, ved at systemet identifiserer hvor utstyr og personell nødvendig for å takle situasjon befinner seg. For eksempel kan det være nødvendig å kontakte en spesialist, lokasjonssystemet vil da kunne fortelle hvilken spesialist som befinner seg nærmest stedet hvor nødssituasjonen har oppstått.

Et lokasjonssystem vil også kunne effektivisere andre arbeidsplasser ved å lokalisere utstyr, som for eksempel på en byggeplass eller i en lagerbygning.

1.3.3 Ulykker

Om en ulykke skulle inntreffe, vil gjerne nødpersonell som politi, ambulanse eller brannredskap bli kontaktet. Personen som kontakter nødpersonell vil ikke alltid kunne redegjøre for hvor han eller hun befinner seg. Vanligvis vil det å vite hvilken GSM basestasjon, eller hvilket trådløse aksesspunkt, den innkommende nødsamtalen er rutet fra gi en tilstrekkelig geografisk avgrensing. En kan allikevel se for seg senarioer hvor en mer nøyaktig lokasjonsbestemmelse vil kunne forenkle arbeidet for nødpersonell. Om en relativt stor bygning skulle brenne, som for eksempel en skole, og en får et nødanrop fra en person inne i bygningen vil det være hensiktmessig å kunne vite nøyaktig hvor denne personen befinner seg. I et slikt senario vil et lokasjonssystem, som utviklet i forbindelse med denne oppgaven, kunne være nyttig. Senarioet går ut i fra at nettverksinfrastruktur nødvendig for at lokasjonstjenesten skal fungere ikke er brent opp.

1.3.4 Lokasjonsavhengig reklame

For at netteiere skal implementere ny funksjonalitet i deres nett, er det gjerne ønskelig at implementasjonen leder til nye inntektskilder. Det er lett å se at en lokasjonstjeneste kan benyttes til reklame. Ved å skreddersy reklame i

forhold til brukerens lokasjon, vil reklamen sannsynligvis være mer effektiv. Med det menes at brukere vil kun få reklame som er relevant i forhold til hvor de befinner seg, dermed øker sannsynligheten for at reklamen gir inntekter hos de som averterer. Netteier vil da kunne øke prisen for å avertere gjennom deres nettverk.

1.3.5 Lokasjonsbaserte tjenester

Ved at netteier har en lokasjonsbestemmende tjeneste i sitt nett, kan det åpnes for at tredjeparts utviklere implementerer tjenester som bygger på denne funksjonaliteten. Det blir dermed ikke opp til netteier å finne ut hva tjenesten skal benyttes til, men de sitter på en funksjonalitet i nettet kun de kan tilby. Det blir opp til andre å bestemme hvilke tjenester som skal baseres rundt denne funksjonaliteten.

1.4 Teknikker for lokasjonsbestemmelse

For å kunne lokalisere noe, et objekt eller en person, må det som skal lokaliseres inneha en mobil enhet det er mulig å identifisere. Samtidig må det finnes stasjonære enheter, enheter med en fast posisjon, som kan identifisere de mobile enhetene avhengig av hvor de befinner seg. En stasjonær enhet kan for eksempel være et aksesspunkt eller en basestasjon. I grove trekk finnes det to måter å lokalisere en mobil enhet.

1. En kan benytte matematiske algoritmer for å regne ut mest mulig nøyaktig lokasjon ut i fra informasjonen som er tilgjengelig. Dette krever at den mobile enheten befinner seg innen rekkevidde av flere stasjonære enheter. Om det er ønskelig med et lokaliseringssystem hvor en mobil enhet blir lokalisert på koordinatform, må det benyttes et system som faller innenfor denne kategorien.

2. En kan benytte et sonebasert system, her vil en stasjonær enhet ha et bestemt dekningsområde. Lokasjonen til den mobile enheten bestemmes avhengig av hvilket dekningsområde den befinner seg innenfor. Dermed kan man oppgi hvilken sone den mobile enheten befinner seg i. En sone kan typisk være et rom, en etasje og liknende.

Dette avsnittet vil forsøke å kartlegge de mest grunnleggende teknikkene for lokasjonsbestemmelse.

1.4.1 Triangulering og trilaterasjon

I dagligtale brukes gjerne ordet triangulering om både triangulering og det som heter trilaterasjon. Innen lokasjonsbestemmende systemer er det i all hovedsak trilaterasjon som benyttes. Trilaterasjon ligger til grunn i for eksempel GPS (se avsnitt 1.5.1). Den etterfølgende teksten vil derfor adskille disse begrepene. Triangulering og trilaterasjon er metoder for å finne lokasjonen til et punkt basert på trigonometri. Metodene kan benyttes i to, tre eller flerdimensjonale rom. Trigonometri gjøres mulig ved at en har punkter med kjent lokasjon. Metodene skilles ved forskjellige matematiske tilnærminger til hvordan lokasjonen til det ukjente punktet estimeres.

For å enkelt beskrive hvordan disse metodene fungerer vil det gås ut i fra et todimensjonalt rom. Begge metodene vil da kreve to punkter med kjente lokasjoner. Det vil også være nødvendig å vite avstanden mellom disse to punktene. Denne avstanden vil danne grunnlinjen i en trekant. Ved triangulering vil en danne imaginære linjer fra de to kjente punktene og til det ukjente punktet. Deretter vil vinklene mellom grunnlinjen og de to imaginære linjene måles. Dette er nok informasjon for å enkelt kunne estimere avstanden til det ukjente punktet. I forbindelse med lokasjonssystemer omtales denne metoden gjerne som "angle-of-arriving (AOA)", og vinkelen mellom grunnlinjen og de imaginære linjene i trekanten finnes ved å studere vinkelen for innkommende radiosignaler hos mottaker.

Ved trilaterasjon forsøker en å måle avstanden mellom de to kjente punktene og det ukjente punktet. Det dannes altså et estimat av lengden av de to ukjente sidene i trekanten. I et lokasjonssystem benyttes radiosignaler for å estimere avstanden mellom de to kjente punktene og det ukjente punktet. GPS estimerer avstanden fra punktet på jorden (med ukjent lokasjon) til satellittene (punktene med kjent lokasjon) ved å måle tiden radiosignaler bruker fra sender til mottaker. Ved hjelp av trigonometri kan en da estimere lokasjonen til det ukjente punktet på jorden. Denne metoden omtales ofte som "time-of-arrival (TOA)" i forbindelse med lokasjonssystemer.

Forklaringene over er noe forenklet. I virkeligheten vil en ikke nødvendigvis vite avstanden av grunnlinjen i trekanten. Det vil derfor i de fleste tilfeller benyttes tre eller flere kjente punkter, for å estimere en ukjent lokasjon i et todimensjonalt rom. For en utdypende forklaring se [22, 12]. Triangulering og trilaterasjon vil i prinsippet kunne tas i bruk for alle typer mobile enheter med sender og mottaker for radiosignaler.

1.4.2 Lokasjonsfingeravtrykk

Et lokasjonsbestemmende system som baserer seg på lokasjonsfingeravtrykk, vil kreve at det samles inn lokasjonsdata for hver lokasjon det ønskes at systemet skal kunne identifisere. Systemets dekningsområde vil være bestemt av hvilke lokasjoner det har blitt gjennomført en slik innsamling for. Ut i fra lokasjonsdataene blir det dannet lokasjonsfingeravtrykk. Lokasjonsfingeravtrykk er en delmengde av lokasjonsdataene registrert for en spesifikk lokasjon. Lokasjonsfingeravtrykkene lagres sentralt, for eksempel i en database, og knyttes opp mot den tilhørende lokasjonen. Når systemet benyttes vil en mobil enhet lese inn lokasjonsdata fra omliggende nettverk, og dannet fingeravtrykk ut i fra denne informasjonen. Disse fingeravtrykkene vil sammenliknes med fingeravtrykkene i databasen, for å estimere hvor brukeren befinner seg [25]. Metoden er i de fleste anledninger implementert i WLAN. En kan allikevel se for seg at det er mulig å benytte metoden sammen med andre teknologier som Bluetooth eller Rfid.

Systemet denne oppgaven baserer seg på tar i bruk lokasjonsfingeravtrykk. Systemet er også laget for og benyttes i WLAN. Fingeravtrykkene består av en målt signalstyrkeverdi, samt en link til hvilket aksesspunkt signalstyrken er generert fra. Fingeravtrykkene linkes også opp mot konkrete lokasjoner, en lokasjon er definert av tre fingeravtrykk.

1.4.3 Ultralyd

Selskapet Sonitor har benyttet ultralyd for å skape et velfungerende innendørs lokasjonssystem. Systemet er sonebasert, og er i hovedsak utviklet for å benyttes på sykehus. Systemets nøyaktighet er avgrenset til romlokasjon, og fungerer på følgende måte: Mennesker og utstyr som skal lokaliseres er utstyrt med en ultralydsender. I hvert rom er det installert en mikrofon som kommuniserer med en sentral enhet. Ultralydsenderne sender kontinuerlig ut et unik identifiserbar ultralyd. Det vil si mønster eller melodier som er unike for denne senderen. Mikrofonene vil plukke opp denne lyden, og videreformidler dette til den sentrale enheten. Den sentrale enheten vil dermed kunne holde oversikt over hvilken sone ultralydsenderne befinner seg i til en hver tid. Systemet tar i bruk ultralyds særegne egenskap i at den ikke penetrerer vegger og tak. Det er derfor ikke mulig for mikrofoner å fange opp ultralyd fra sendere i nærliggende rom/soner.[2]

[2]http://www.sonitor.com/, 25. februar 2010

Lokasjonssystemet Cricket benytter også ultralyd som en sentral del av deres innendørs lokasjonssystem. Cricket har valgt å la stasjonære enheter kalt "beacons" sende ut ultralyd samt meldinger som inneholder deres lokasjon. Mobile enheter registrerer ultralyd sendt ut fra stasjonære enheter, og beregner ved hjelp av dette avstanden til de stasjonære enhetene. Ved hjelp av meldingene som inneholder de stasjonære enhetenes lokasjon, vil de mobile enhetene kunne kalkulere sin egen lokasjon. En mobil enhet som ønsker å benytte systemet må være koblet til en Cricket "listener", dette er en hardware enhet som vil kunne lytte på meldinger og ultralyd fra "beacons". Cricket oppgir den gjennomsnittlige nøyaktigheten av systemet til og være 10 cm. Denne nøyaktigheten er funnet ved å teste systemet i et 9 kvadratmeters området med hele 5 stasjonære enheter [26].

Begge disse ultralyd lokasjonssystemene er proprietære, da de vil kreve spesiallaget hardware sendere og mottakere for å kunne fungere.

1.5 Fungerende lokaliseringsløsninger

Dette avsnittet vil kartlegge sentrale kommersielt lanserte lokaliseringsløsninger som er tilgjengelig i dag. Utendørs er GPS den dominerende løsningen, mens det innendørs finnes en flere ulike løsninger. Hensikten med avsnittet er ikke å beskrive alle lanserte løsninger, men heller gi et innblikk i ulike tilnærminger for å skape lokasjonssystemer.

1.5.1 Global Positioning System

Global Positioning System (GPS) består av et nettverk av satellitter som går i bane rundt jorden. Dette satellittnettverket gjør det mulig å bestemme lokasjonen til GPS-utstyr på jorden med en nøyaktighet på ca. fem til ti meter. Satellittene beveger seg rundt jorden med en syklus på om lag et halvt døgn, i svært nøyaktige baner. Det er GPS-utstyret på jorden som kalkulerer sin lokasjon ved hjelp av informasjon som blir sendt ut av satellittene. Det kreves at en mottar signaler fra tre satellitter for at en skal kunne finne en todimensjonal posisjon. For å kunne estimere en tredimensjonal posisjon må det mottas signaler fra fire eller flere satellitter. Avstanden til satellittene benyttes for å bestemme brukerens lokasjon, disse avstandene blir estimert ut i fra hvor lang til signalet bruker fra satellitt til mottaker. For å kunne øke systemets nøyaktighet finnes det GPS-stasjoner på jorden, som vil kunne sende

korreksjonssignaler. Dette blir sjeldent brukt blant vanlige private brukere.[3] GPS vil normalt ikke kunne fungere innendørs da signalene vil ha problemer med å trenge gjennom bygningsmasse, og de vil bli utsatt for signalforurensning som refleksjoner og multipath (se avsnitt 2.1.4). Det ser allikevel ut som det er mulig og benytte GPS innendørs om brukerutstyret forbedres [30].

EU har ansvar for utviklingen av et nytt satellittnettverk kalt Galileo. Ved blant annet å benytte seg av flere satellitter skal Galileo kunne gi mer nøyaktige lokasjonsberegninger enn GPS.[4] Det er mulig Galileo vil kunne fungere innendørs.

1.5.2 Navizon

Navizon er et programvarebasert lokasjonssystem som tar i bruk informasjon fra GPS, trådløse nettverk og GSM basestasjoner. Hensikten med systemet er å utvikle et alternativt lokasjonssystemet til GPS, for brukerutstyr som ikke inneholder GPS-mottakere. Systemet er fellesskapsbasert (community based), det vil si at systemet er avhengig av interaksjon fra brukermassen. Navizon-brukere med GPS registrerer informasjon om trådløse aksesspunkt, samt GSM basestasjoner, for lokasjonen de befinner seg ved. Denne informasjonen samt tilhørende GPS-koordinater blir formidlet til en sentral server. Brukere uten GPS kan laste ned en applikasjon til deres mobile enhet, som vil kunne fungere som en virtuell GPS. Applikasjonen vil samle inn informasjon om nærliggende aksesspunkter og GSM basestasjoner. Denne informasjonen blir videreformidlet til Navizon Central Server som basert på informasjonen vil kalkulere et estimat for hvor brukeren befinner seg, og formidle denne lokasjonen til brukeren.[5] Da Navizon baserer seg på innsamlet informasjon via GPS, vil systemet ikke kunne fungere innendørs. Det er også stor usikkerhet ved Navizons nøyaktighet. Nøyaktigheten vil avhenge av hvor mye innsamlet informasjon det finnes for området en befinner seg i, og hvor mange trådløse aksesspunkt samt GSM basestasjoner en vil kunne registrere i området.

1.5.3 Cisco 2700 Series Wireless Location Appliance

Cisco har utviklet en proprietær lokasjonsserver, det vil si den kun vil fungere med en Cisco WLAN infrastruktur. Serveren kan registrere lokasjonen

[3] http://snl.no/GPS, 25. februar 2010
[4] http://ec.europa.eu/transport/galileo/why_en.htm, 25. februar 2010
[5] http://www.navizon.com/, 25. februar 2010

av opptil 1500 simultane brukere av det trådløse nettet, med en innendørs nøyaktighet innenfor ti kvadratmeter. Serveren vil også kunne oppgi hvordan signalstyrkeforholdene vil variere innenfor nettverkets dekingsområde.[6] Kostnaden av en slik server vil avhenge av antallet aksesspunkt som skal inkluderes i lokaliseringsløsningen. I 2006 ble prisen for å inkludere en slik server i det trådløse nettverket ved NTNU Gløshaugen oppgitt til å være 1,2 millioner kroner. Det trådløse nettverket ved NTNU Gløshaugen består av omlag 1500 aksesspunkt.[7]

1.5.4 Radionor

Radionor er et firma som har utviklet et lokaliseringssystem ved hjelp av spesiallaget hardware. De har utviklet tre konkrete enheter for implementasjon av deres system. Avhengig av hvilke komponenter som benyttes kan en oppnå sone- eller koordinatbasert lokalisering.

1. Cordis Access: Inneholder to 802.11 a/b/g aksesspunkt. Cordis Access danner et sonebasert lokasjonssystem ved at en Cordis Localization antenne festes til aksesspunktet. Flere slike aksesspunkt kan sammen danne et lokasjonssystem hvor hvert aksesspunkt dekker en sone.

2. Cordis RadioEye: Er en sensor som vil kunne oppgi lokasjonen til mobile enheter på koordinatform. Et slikt radioøye kan dekke opptil 3000 kvadratmeter, og levere en lokasjonsnøyaktighet på under en meter. Prisen ligger på mellom 70000 og 100000 NOK avhengig av kapasitet og funksjonalitet. Det oppgis ingenting om begrensninger i form av dekningsområde. Med det menes hvordan radioøye vil takle mange små rom, korridorer, flere etasjer og liknende.

3. Cordis WLAN ID Transmitter: Enhet laget for å sende ut meldinger som vil oppfattes av Radionors lokasjonssystem. Enhetene er designet for å kunne plasseres på personer eller gjenstander en ønsker å lokalisere, og skal ha en batteri levetid på mer enn 2 år ved vanlig bruk.

Cordis Access og Cordis RadioEye vil også kunne fungere sammen.[8]

[6] http://www.cisco.com/en/US/products/ps6398/, 25. februar 2010
[7] Thomas Jelle, Trådløse Trondheim
[8] http://radionor.com/, 26. februar 2010

1.5.5 Ekahau

Ekahau har utviklet et lokasjonssystem som baserer seg på eksisterende WLAN infrastruktur. Kjernen av systemet er en patentert programvarebasert algoritme for å beregne lokasjoner. Systemet består av Ekahau Positioning Engine (EPE), Ekahau Site Survey (ESS) og Ekahau trådløse tags. EPE vil beregne lokasjon til trådløse tags ved hjelp av WLAN signaler. EPE må vite dekningsområdet av hvert aksesspunkt inkludert i systemet, samt hvordan signalstyrken varierer innenfor dekningsområdet. Denne informasjonen blir oppnådd ved å benytte ESS, et program for å sette opp lokasjonsmodeller. De trådløse tag-ene må konfigureres med hensyn på blant annet hvor ofte deres lokasjon skal oppdateres, før de kan benyttes. Når systemet benyttes vil tag-ene måle mottatt signalstyrke fra hvert aksesspunkt innen rekkevidde. Denne informasjonen videreformidles til EPE over det trådløse nettet, før EPE kalkulerer tagens lokasjon. For å benytte annet WLAN kompatibelt utstyr med Ekahaus lokasjonsløsning, må utstyret ha installert Ekahaus klient software. Denne softwaren vil ikke kunne fungere på alle plattformer. Ekahaus system baserer seg på en lokasjonsfingeravtrykk liknede løsning.[9]

1.6 Hvorfor lokasjonsfingeravtrykk?

Denne oppgaven er skrevet for Trådløse Trondheim. Et overordnet mål med oppgaven er å finne ut om lokasjonsfingeravtrykk vil være en fornuftig måte å implementere et lokasjonssystem i store bygninger, og da særlig ved NTNU campus. Det trådløse nettverket ved campus eies av NTNU, men Trådløse Trondheim vil være en virtuell leverandør av en slik lokasjonstjeneste. De følgende argumentene vil gå ut i fra denne spesifikke settingen for implementasjon av et lokasjonsbestemmende system. Derfor vil først Trådløse Trondheim bli beskrevet.

1.6.1 Trådløse Trondheim

Trådløse Trondheim ble stiftet i 2006 og har følgende fokusområder:

1. Bygge og levere trådløst bredbåndsnett i Trondheim.
2. Forsking og utvikling av nye trådløse og mobile tjenester.

[9] http://www.ekahau.com/, 26. februar 2010

3. Sammen med NTNU fasilitere og tilrettelegge for forskning og utvikling, gjennom å tilby en innovasjonsarena og testlaboratorium for nye tjenester og produkter - "Wireless Trondheim Living Lab".

I dag dekker Trådløse Trondheim store deler av Midtbyen (Trondheim sentrum) utendørs, og kan levere innendørs dekning i deler av dette området. Samtidig er deres dekningsområde i stadig utvikling. Dette har ledet til at Trondheim ble en av de første trådløse byene i Europa. Trådløse Trondheim har 120 aktive aksesspunkt i Trondheims bykjerne, samtidig opererer de som virtuell leverandør for 350 aksesspunkt tilhørende Trondheim kommune.[10] Campus ved NTNU er dekket av 1500 aksesspunkt, over et brutto areal på mer enn 300000 kvadratmeter[11]. Per i dag har Trådløse Trondheim implementert en lokasjonstjeneste ved hjelp av en lokasjonsserver av typen Cisco 2700 Series Wireless Location Appliance (se avsnitt 1.5.3). Denne lokasjonstjenesten er begrenset til Trondheim sentrum. Trådløse Trondheim ønsker også å kunne tilby en lokasjonsbestemmende tjeneste i andre nettverk hvor de er virtuell leverandør, og da særlig innendørs. Ciscos lokasjonsserver er svært kostbar, og det er dermed et behov for å undersøke om det finnes andre måter en kan implementere en lokasjonstjeneste i store trådløse nettverk.

1.6.2 Fordeler ved lokasjonsfingeravtrykk

Dette avsnittet vil beskrive fordeler ved å implementere en innendørs lokaliseringsløsning ved hjelp av lokasjonsfingeravtrykk.

1. Løsningen kan implementeres på allerede eksisterende infrastruktur, og det vil kun kreves en hyllevare server. Det trådløse nettverket ved NTNU campus består av en svært omfattende infrastruktur, noe som tilrettelegger for et nøyaktig og stabilt system.

2. Løsninger som baserer seg på ekstra utstyr i nettet kan ofte ende opp som svært kostbare (jf. avsnitt 1.5.3 og 1.5.4), da slikt utstyr er relativt dyrt.

3. Signalstyrken fra aksesspunkt i trådløse nettverk varierer som en følge av avstand og signalforurensning (se avsnitt 2.1.2). Trilaterasjon (jf. avsnitt 1.4.1) har vist seg å fungere godt i utendørs lokasjonssystemer. Innendørs vil signalforurensning som en følge av multipath, det vil si at signalet finner flere veier fra sender til mottaker på grunn av refleksjo-

[10] http://tradlosetrondheim.no/, 15. mars 2010

[11] Informasjon fra eiendomssjef Arne Rønning ved NTNU Tekniskavdeling.

ner, spredning og avbøyning, lede til at en slik metode blir unøyaktig [2]. Ved bruk av lokasjonsfingeravtrykk benyttes signalstyrke kun til å identifisere en lokasjon. Så lenge signalforurensningen forholder seg relativt stabil, vil den ikke kunne påvirke nøyaktigheten av systemet.

4. Tidligere forskning på lokasjonssystemer basert på lokasjonsfingeravtrykk har vist lovende resultater. Phongsak Prasithsangaree et al. har implementert en prototype av slikt system, med et dekningsområde over en etasje bestående av forskjellige rom og korridorer. Nøyaktigheten av lokasjonsestimatene i deres system varierte fra to til syv meter, avhengig av nøyaktigheten av lokasjonsinnsamlingen [25]. Kaveh Pahlavan et al. gjennomførte et lite eksperiment med 34 lokasjoner ca to meter fra hverandre i en korridor. Det gjennomsnittlige standardavviket for et lokasjonsestimat ble registret til 2,4 meter [24].

Det er to hovedtilnærminger til hvordan innsamlingen av lokasjonsdata kan foregå. Systemet kan være fellesskapsbasert slik som Navizon. Det blir dermed opp til brukerne av lokasjonstjenesten å definere dekningsområdet. Ved en slik tilnærming legges ansvaret for innsamlingen av lokasjonsdata på systemets brukergruppe. Netteier vil ha minimal kontroll over systemets dekningsområde, og nøyaktigheten av lokasjonsdataene. Et annet alternativ er en nettsentrert løsning, det vil si at netteier sitter på all informasjon om systemets dekningsområde. Det gjennomføres en systematisk innsamling av lokasjonsdata, og netteier har mulighet til å forandre lokasjonsdataene i henhold til forandringer i nettverksinfrastrukturen. Nøyaktigheten av lokasjonsdataene og dekningsområdets omfang vil da være kjent for netteier til en hver tid. Et nettsentrert lokasjonssystem basert på lokasjonsfingeravtrykk krever mye arbeid i form av innsamling av lokasjonsdata. Om denne innsamlingsfasen kan delvis automatiseres vil det være mulig å spare mye tid. Fordelen med en nettsentrert løsning er at systemet vil kunne bli mer nøyaktig grunnet den systematiske datainnsamlingen, og en vil kunne oppnå lokasjoner på koordinatform. Det er urealitisk å forvente at et systems brukergruppe vil kunne registrere lokasjoner på koordinatform, særlig innendørs hvor eventuelle koordinater ikke vil kunne bestemmes ved hjelp av GPS. Det er heller ikke å forvente at en brukergruppe vil samle inn lokasjonsdata for alle lokasjoner. Typisk vil heller lokasjoner hvor brukere ofte befinner seg bli registrert, dette kan for eksempel være lesesaler, datasaler eller kafeer. Systemet utviklet i forbindelse med denne oppgaven er nettsentrert.

1.6.3 Hvordan skiller arbeidet seg fra tidligere forskning?

Dette avsnittet vil beskrive hvordan lokasjonssystem prototypen utviklet i denne oppgaven skiller seg fra andre innendørs lokasjonssystem basert på lokasjonsfingeravtrykk. Generelt skiller prototypen seg fra annen forskning ved at det er fokusert på et helhetlig system. Det har derfor blitt vektlagt flere sentrale aspekter ved systemets funksjonalitet, som vil være essensielle ved kommersiell bruk. Annen forskning har i stor grad fokusert på én utfordring ved lokasjonsfingeravtrykk systemer av gangen. Følgene forskning er lagt til grunn i dette avsnittet når det refereres til annen forskning: [18, 7, 21, 25, 13, 24, 14, 4, 2]. Prototypen utviklet i forbindelse med denne oppgaven skiller seg fra annen forskning på følgene punkter:

Dekningsområde

Dekningsområdet for lokasjonssystemet utviklet i forbindelse med denne oppgaven, skiller seg fra dekningsområder i andre implementasjoner med hensyn på størrelse og variasjon i bygningsmessige omgivelser. Det er fokusert på å identifisere hvordan et fingeravtrykk lokasjonssystem fungerer i områder en typisk vil finne i store bygg, slik som ved NTNU campus. Det er derfor essensielt at dekningsområdet er representativt for slike bygninger. Dekningsområdet inneholder derfor:

- Store åpne arealer, slik som i Glassgården i El-Bygget ved NTNU Gløshaugen.
- Avlukkende rom slik som kontorer, lesesaler og liknende.
- Korridorer.

Dekningsområdet inneholder totalt 856 oppmålte lokasjoner og er lokalisert over to etasjeplan. Det er ikke funnet annen forskning med dekningsområder tilsvarende skalaen tatt i bruk i dette systemet. Ved å ta i bruk et lite dekningsområde oppnås det en minimal forståelse av hvordan systemet vil fungere for et stort dekningsområde. Et lite dekningsområde tilsier få lokasjoner i lokasjonsdatabasen. Det er dermed en mindre sannsynlighet for at fingeravtrykkene tilhørende en unøyaktig lokasjon passer bedre overens med de reelle lokasjonsdataene. Ved å benytte et lite dekningsområde vil en derfor oppnå bedre nøyaktighet, enn ved å benytte et stort dekningsområde for et lokasjonssystem basert på fingeravtrykk.

Ulike mobile enheter

Prototypen utviklet i forbindelse med denne oppgaven tilrettelegger for bruk av ulike typer mobile enheter. Ved å utvikle programvare slik at lokasjonssystemet fungerer på iPhone samt Android-telefoner, i tillegg til at det tidligere er bevist å fungere for bærbare datamaskiner [32], har en utviklet et plattformuavhengig lokasjonssystem. Dette byr på utfordringer en ikke vil identifisere om det kun benyttes en type mobil enhet. Forskjellige mobile enheter inneholder ulik hardware som antenner og nettverkskort. Dette leder til at de vil oppfatte omliggende trådløse nettverk forskjellig. Annen forskning har kun tatt i bruk en type mobil enhet.

Tidsperspektiv

Tiden det tar for lokasjonssystemet å gjennomføre et lokasjonsestimat vil være en essensiell del av brukerens oppfattelse av systemets ytelse. Dette har det blitt tatt hensyn til under utviklingen av prototypen denne oppgaven baserer seg på. Det er ikke funnet annen forskning som oppgir hvor lang tid deres prototype trenger for å gjennomføre et lokasjonsestimat, dette antas derfor ikke å ha blitt vektlagt.

Brukerens nøyaktighetsoppfattning

Annen forskning på lokasjonssystemer basert på lokasjonsfingeravtrykk har kun fokusert på nøyaktigheten av et lokasjonsestimat som avvik i avstand fra reell til estimert lokasjon. For brukeren er det minst like viktig at riktig rom og etasje blir estimert av lokasjonssystemet. Slike utfordringer er adressert i denne oppgaven. Det er ikke funnet annen forskning som studerer hvordan brukeren vil kunne oppfatte resultatet av et lokasjonsestimat.

Kapittel 2

Teknologi

Dette kapittelet vil kartlegge teknologier som har vært essensielle i arbeidet med denne masteroppgaven. Teknologiene er beskrevet overfladisk og det vil bli vektlagt informasjon som er nødvendig for forstå resten av oppgaven. Det er gått ut i fra at lesere av oppgaven allerede har en grunnleggende forståelse for teknologiene som er beskrevet.

2.1 Trådløse nettverk

Med trådløse nettverk menes i denne oppgaven Wireless Local Area Network(WLAN). Dette er en IEEE industristandard, og er den mest utbredte standarden for trådløs IP-trafikk. Det offisielle navnet på standarden er IEEE Std 802.11 [1]. I denne oppgaven vil det fokuseres på aksesspunkter som følger denne standarden, og mobile enheter i kontakt med slike aksesspunkt.

2.1.1 IEEE Std 802.11

IEEE 802.11-standarden har i dag fire protokoller benyttet for kommunikasjon over trådløse nettverk[1]:

- **802.11b**: Den første vidt aksepterte protokollen for trådløs kommunikasjon, og har vært å finne i produkter fra rundt år 2000. Protokollen gir en maksimal overføringsrate av rådata på 11 Mbit/s, og opererer på 2,4 GHz frekvensbåndet. Dette frekvensbåndet benyttes også for blant

annet mikrobølgeovner, bluetooth kommunikasjon og trådløse fasttelefoner, noe som kan lede til interferens. 802.11b er sammen med 802.11g de mest benyttede protokollene innenfor standarden.

- **802.11g**: Benytter samme frekvensbånd som 802.11b protokollen, men en annen modulasjon. Dette gir en maksimal overføringsrate av rådata på 54 Mbit/s. 802.11g hardware er tilbake kompatibelt med 802.11b hardware.
- **802.11a**: Tar i bruk 5GHz frekvensbåndet som sammenliknet med 2,4 GHz frekvensbåndet er relativt ubenyttet. Protokollen gir en maksimal overføringsrate av rådata på 54 Mbit/s. Dekningsområdet for et aksesspunkt som benytter 802.11a vil normalt være mindre enn for et aksesspunkt som benytter 802.11b/g. Årsaken til dette er valget av frekvensbånd. Høyere frekvenser vil gi mindre bølgelender (se ligning 2.2), og signalene blir dermed lettere absorbert i vegger, tak og gjenstander.
- **802.11n**: Protokollen gir støtte for bruk av både 2,4 og 5 GHz frekvensbåndene. I tillegg innførte protokollen blant annet støtte for MIMO(multiple input og multiple output), en teknikk som benytter flere antenner hos både sender og mottaker for å forbedre den trådløse kommunikasjonen.

802.11-standarden deler frekvensbåndene inn i kanaler. Hver kanal har normalt en bredde på 22 MHz, og hver kanal er plassert 5 MHz fra hverandre. Dette gir en stor overlapp mellom kanalene, og kan lede til interferens. 2,4 GHz frekvensområdet som strekker seg fra 2,4000–2,4835 GHz blir delt inn i 13 kanaler. Normalt benyttes kanal 1, 6 og 11 da disse ikke overlapper. I enkelte tilfeller finner en også en 14. kanal plasser 12MHz etter kanal 13, dette gir en fjerde ikke overlappende kanal[1].

5Ghz frekvensbåndet strekker seg fra 4,915-5.825 GHz. Dette frekvensområdet er nasjonalt regulert, og hvert enkelt land har gjerne egne restriksjoner på hvilke kanaler innenfor frekvensområdet en kan benytte til WLAN. I denne oppgaven vil det benyttes fire vesentlige verdier som oppgis av aksesspunkter som følger 802.11-standarden. SSID er nettverkets navn, for eksempel har systemet som har blitt utviklet i forbindelse med denne oppgaven benyttet et nettverk med SSID lik eduroam. BSSID er en unik identifikator for en aksesspunkt innenfor et spesifikt nettverk. Ingen aksesspunkt innenfor eduroam nettverket vil ha samme BSSID. BSSID benytter samme format som MAC-adresser, det vil si at de skrives på heksadesimalt format med seks grupperinger av tall. Hver tallgruppering består som regel av to tall, og holdes adskilt ved hjelp av kolon eller bindestrek. Aksesspunktene kan benytte

forskjellige kanaler, i lokasjonssystemet denne oppgaven baserer seg på blir kanal verdien brukt til å identifisere hvilket frekvensbånd aksesspunktet tar i bruk. Den siste verdien som har blitt benyttet er RSSI, denne er beskrevet i neste avsnitt.

2.1.2 RSSI

RSSI står for Received Signal Strength Indicator og er en indikator på signalstyrke. 802.11-standarden har definert RSSI som et mål på relativt mottatt signalstyrke [1]. RSSI verdien blir kalkulert i brukerutstyret, som en resultat av mottatt signalstyrke fra et aksesspunkt [20]. Systemet som har blitt utviklet i forbindelse med denne oppgaven benytter RSSI som en indikator på hvor mobile enheter befinner seg. Både iPhone OS (se avsnitt 2.7), Android (se avsnitt 2.5) og Macintosh datamaskiner (se avsnitt 2.9) benytter en RSSI skala fra -100 til 0. Benevningen for RSSI er dBm. 0 dBm representerer den teoretisk sterkeste signalstyrken det er mulig å oppnå. De sterkeste verdiene på RSSI skalaen vil man i praksis ikke kunne registrere, normalt vil RSSI verdiene ligge mellom -30 og -95 dBm. Det er verdt å merke seg at skalaen vil kunne variere i forskjellige mobile enheter, avhengig av leverandøren. En skala fra -100 til 0 er det vanligste.

2.1.3 Campus aksesspunkt

Aksesspunktene på campus er av typen Cisco 1131[1] og Cisco 1242[2]. Disse aksesspunktene har både 802.11 b/g og a brikkesett. Hvert aksesspunkt benyttes til å levere flere virtuelle nettverk, et nettverk er da definert ut i fra SSID. På campus gir et aksesspunkt tilgang til nettverkene: Eduroam, NTNU og NTNUguest. Samtidig vil også et aksesspunkt kunne fungere som to imaginære aksesspunkt innenfor et nettverk. Dette realiseres ved at aksesspunktet benytter begge brikkesettene og dermed både 2,4 og 5 GHz frekvensbåndet. Aksesspunktet vil ha en BSSID for hvert brikkesett, og dermed vil dette oppfattes som to separate aksesspunkt for brukeren.

[1] http://www.cisco.com/en/US/products/ps6087/, 3. mars 2010
[2] http://www.cisco.com/en/US/products/ps6521/, 3. mars 2010

2.1.4 Innendørs påvirkning av WLAN radiosignaler

I prototypen utviklet i forbindelse med denne oppgaven vil RSSI fra WLAN radiosignaler benyttes til å identifisere en lokasjon. En utfordring ved dette er at signalforurensning vil påvirke mottatt signalstyrke, slik at den vil variere tidsavhengig. Signalforurensningen blir skapt av radiosignalenes omgivelser mens de overføres fra sender til mottaker. Det er derfor nødvendig å identifisere hvordan radiosignaler vil påvirkes i en innendørs omgivelse, da dette vil være sentralt for prototypen utviklet i forbindelse med denne oppgaven. Innendørs omgivelser består av både statiske elementer som vegger, tak og interiør, samt mobile elementer som mennesker og dører som åpnes og lukkes.

Radiosignaler benyttet i WLAN kan karakteriseres ved at de har en liten bølgelengde sammenliknet med avstanden mellom sender og mottaker. Dette er en vesentlig egenskap som vil være med å definere hvordan slike signaler påvirkes av omgivelsene. I grove trekk vil signalene bli utsatt for tre typer påvirkning [28]:

1. **Refleksjoner:** Refleksjoner oppstår i det radiobølger treffer en overflate, dette kan være en vegg, et tak eller liknende. En refleksjon vil kunne ha tre forskjellige utfall. Signalet kan bli fullstendig reflektert, absorbert, eller en kombinasjon av disse. Resultatet av en refleksjon vil avhenge av fysiske egenskaper ved overflaten signalet treffer som material, form og tykkelse, samt signalets egenskaper som bølgelengde og med hvilken vinkel signalet treffer overflaten.

2. **Avbøyning:** Signaler avbøyes om de treffer skarpe kanter de ikke vil kunne penetrere. Radiosignalene vil bøye seg rundt, og det vil oppstå bølger på baksiden av kanten i henhold til Huygens prinsipp. Dette fenomenet er tilsvarende som når bølger i vann treffer for eksempel en åre. Resultatet av avbøyningen avhenger av signalets egenskaper som amplitude, fase og polarisering, samt kantens egenskaper som form og hvordan den er plassert i forhold til resten av omgivelsene. Avbøyning vil typisk oppstå rundt hjørner i korridorer, eller på kanten av objekter som møbler og liknende.

3. **Spredning:** Små objekter som befinner seg mellom sender og mottaker vil kunne splitte opp radiosignaler, dette kalles spredning. Ved spredning vil de forskjellige delene av radiosignalet bli sendt ut i forskjellige retninger.

Multipath og fading

Refleksjoner, avbøyning og spredning vil lede til at radiosignalet når mottaker via flere veier (paths), dette fenomenet kalles multipath. Multipath vil gi forskjeller i mottatt signalstyrke, avhengig av hvilken vei signalet har tatt. Generelt vil mer påvirkning av signalet lede til svakere mottatt signalstyrke. I en innendørs omgivelse hvor blant annet mennesker beveger seg og dører åpnes og lukkes, vil signalet kunne påvirkes forskjellig avhengig av omgivelsenes bevegelse. Dette leder til at radiosignalets forskjellige veier (multipaths) til mottaker vil forandres tilfeldig, og mottatt signalstyrke vil være tilegnet en tilfeldig variabel. Variasjoner i mottatt signalstyrke som en følge av multipath vil også avhenge av type antenne i mottakerutstyret, samt antennens posisjon[28].

Som en følge av multipath vil signalet kunne mottas flere ganger med små tidsforskjeller hos mottaker, det vil da kunne oppstå interferens. Interferens vil påvirke registret signalstyrke. De interfererende radiobølgene vil kunne skape både konstruktiv interferens som vil øke mottatt signalstyrke, og destruktiv interferens som vil dempe mottatt signalstyrke. Dette fenomenet kalles fading[27]. Fading vil lede til at mottatt signalstyrke vil fluktuere rundt en gjennomsnittsverdi for en bestemt lokasjon[15].

Menneskelig påvirkning av WLAN radiosignaler

Menneskelig påvirkning av radiosignaler vil lede til signalforurensning ved at personer vil være mobile elementer i omgivelsene der radiosignalene overføres. Allikevel vil den kanskje største menneskelige påvirkning av radiosignalene komme fra brukeren av den mottakene mobile enheten. Brukeren vil ofte kunne dekke store deler av "Line-Of-Sight (LOS)" for den mobile enhetens antenne. Det vil si at brukeren kommer i veien for fri luftlinje mellom aksesspunkt og den mobile enheten. Resonans frekvensen i vann er 2,4 GHz. Dette er samme frekvens som i hovedsak benyttes for WLAN radiosignaler (jf. avsnitt 2.1.1). Mennesker består av 70% vann, og radiosignalene har dermed lett for å bli absorbert i menneskekroppen [15]. Hvordan brukeren er posisjonert i forhold til aksesspunkt og mobil enhet er derfor essensielt for hvordan signalstyrken vil oppfattes. Kamol Kaemarungsi og Prashant Krishnamurthy har studert hvordan menneskekroppen påvirker registrert signalstyrke for en mobil enhet. Dette ble gjort ved å la en mobil enhet registrere signalstyrker fra et aksesspunkt i to senarioer. Et senario hvor en person er tilstede ved den mobile enheten som ved normal bruk, og et annet der den mobile

enheten registrerer mottatt signalstyrke uten menneskelig påvirkning. Resultatene viste større spredning og standardavvik i mottatt signalstyrke for senarioet der brukeren var tilstede. Med brukeren tilstede varierte RSSI fra -79 til -67 dBm, mens uten brukeren varierte RSSI fra -72 til -69 dBm. Standardavviket i mottatt signalstyrke med brukeren tilstede ble registrert til 3 dBm, mens når brukeren ikke var tilstede ble det registrert et standardavvik på 0.68 dBm [15].

Signaltap

Signaltap er en degradering av signalstyrke. Alle radio signaler vil oppleve signaltap som en følge av avstand mellom sender og mottaker. Dette signaltapet omtales gjerne som "free space loss (FSL)", og er gitt i likning 2.1.

$$FLS = \left(\frac{4\pi d}{\lambda}\right)^2 \tag{2.1}$$

I denne linkningen er d avstanden mellom sender og mottaker, mens λ er radiosignalets bølgelengden i meter [28]. Ved å relatere frekvensen til bølgelengde (jf. likning 2.2) ser en at det å benytte 5 GHz frekvensbåndet til WLAN kommunikasjon, vil gi et større signaltap som følge av avstand enn ved bruk av 2,4 GHz frekvensbåndet. c i linkning 2.2 er lysets hastighet. Det ble i avsnitt 2.1.1 identifisert at radio signaler i 5 GHz frekvensbåndet også lettere vil bli absorbert i vegger, tak og liknende som følge av mindre bølgelengde.

$$\lambda = \frac{c}{f} \tag{2.2}$$

Et senario der signaltap kun avgjøres av avstanden mellom sender og mottaker er et ideal tilfelle en aldri vil kunne oppnå innendørs. I virkeligheten vil refleksjoner, avbøyning, spredning og multipath sørge for både signaltap og transformering av radiosignalet. Innendørs vil de mange variasjonene og den komplekse sammensetningen av både statiske og mobile elementer gjøre det svært vanskelig å forutse styrken av et mottatt signal. Små variasjoner i mottakers posisjon eller vinkel i forhold til sender vil kunne lede til store forskjeller i mottatt signalstyrke [28]. Denne oppgaven vil ikke forsøke å estimere hvordan WLAN signalene vil forplante seg avhengig av innendørs omgivelser. Allikevel er dette svært sentral kunnskap for å forstå egenskapene til WLAN signaler, og dermed kunne avgjøre hvordan RSSI kan benyttes for å identifisere lokasjoner i et lokasjonsfingeravtrykk system. I denne oppgaven vil det spesielt være den påvirkningen som skjer ved naturlige forandringer

som vil være essensiell. Det kan typisk være møbler og personer som endrer posisjon, eller vinduer og dører som åpnes og lukkes.

2.2 Euklids avstand

Euklids avstand kan defineres som en matematisk måte å utrykke avstanden mellom to punkter som en rett linje. Euklids avstand mellom punktene p og q er lik lengden av linje segmentet $\vec{pq}$. Euklids avstand mellom punktene p og q i et n-dimensjonalt rom defineres som [6]:

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + ... + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \quad (2.3)$$

I implementasjonen av systemet denne oppgaven baserer seg på, benyttes Euklids avstand i et matematisk tredimensjonalt rom. En lokasjon i databasen er definert av tre fingeravtrykk med tilhørende RSSI verdier. Euklids avstand benyttes til å beregne hvilken lokasjon i databasen som er mest nærliggende de reelt målte RSSI verdiene. Det vil være lokasjonen som gir den minste verdien når det beregnes Euklids avstand.

2.3 Socketkommunikasjon

En socket kan ses som et endepunkt av en toveis kommunikasjon over et nettverk. Begrepet socketkommunikasjon brukes ofte om APIer for kommunikasjon over IP. En socket blir definert ut i fra følgene identifikatorer[9]:

- **Protokoll**: Det vil si hvilken protokoll som benyttes under overføringen av data. TCP og UDP er de mest brukte protokollene. Disse protokollene tar i bruk adskilte sockets, slik at en TCP forbindelse på port 100 viser til en annen socket enn en UDP forbindelse på port 100.
- **IP-adresse**: Benyttes for å identifisere dataenheten socketen er lokalisert på.
- **Port nummer**: Benyttes for å identifisere hvilken applikasjon på dataenheten det kommuniseres med.

For systemet denne oppgaven baserer seg på er det to relevante socket typer:

- **Stream socket**: Ved å ta i bruk stream sockets oppnås det en connection-oriented pakkesvitsjet kommunikasjonslink som tar i bruk TCP protokollen. Det vil bli satt opp en forbindelse mellom begge sider av kommunikasjonslinken ved hjelp av et treveis håndtrykk. TCP protokollen sørger for at alle pakker mottas, og at de mottas i riktig rekkefølge. TCP inkluderer også en kontroll av innholdet i hver mottatte pakke, som gjør at korrupte pakker vil bli sendt på nytt.
- **Datagram socket**: Ved å ta i bruk datagram sockets oppnås det en connectionless pakkesvitsjet kommunikasjonslink som tar i bruk UDP protokollen. UDP har ingen garanti for at pakkene ankommer, og leverer dermed en upålitelig forbindelse. Det kreves ingen håndtrykk prosedyre for å sette opp en forbindelse, og protokollen sender ikke tapte pakker på nytt. Dette leder til at dataoverføring ved hjelp av UDP gjerne er raskere enn ved bruk av TCP [9].

2.4 Java

Java plattformen benyttes i dag av mer enn 6.5 millioner softwareutviklere. Plattformen er integrert i fler enn 4.5 milliarder utstyrskomponenter. Dette innebærer:[3]

- Over 800 millioner datamaskiner.
- 2.9 milliarder mobile enheter.
- 3.5 milliarder smart cards.
- Samt printere, webkameraer, medisinsk utstyr, navigeringssystemer for bil, med mer.

Java er uavhengig av underliggende hardware og operativsystem. Dette er fordi koden blir kompilert til en representasjon kalt Java bytecode, i stedet for å kompilere til den plattform spesifikke maskinkoden. Eneste krav er at brukerutstyret har implementert Java Runtime Environment (JRE). Det benyttes en Java Virtual Machine (JVM), spesifikk for underliggende hardware, for å tilpasse Java bytekoden plattformen som benyttes. Dette gjør det enkelt å overføre programvare skrevet i Java til forskjellige enheter uavhengig av underliggende plattform. Grunnet Javas egen bytekode representasjon, har Java-applikasjoner en tendens til å kjøre tregere enn applikasjoner implementert direkte til den plattform spesifikke maskinkoden. Denne forskjellen

[3] http://java.com/en/about/, 26. april 2010

i ytelse har blitt minimert de siste årene, da de plattformspesifikke JVM implementasjonene har blitt optimalisert.[4]

2.5 Android

Android er en programvarepakke for mobiltelefoner som inneholder operativsystem, mellomvare og grunnleggende applikasjoner. Slike applikasjoner er for eksempel SMS tjeneste, kalender, kontaktbok og liknende. Android operativsystemet benytter en Linux-kjerne. Denne kjernen sørger for grunnleggende system funksjonalitet og samkjører hardware med resten av mobiltelefonens programvare.

Android tilbyr en åpen utviklingsplattform som gir utviklere tilgang til de samme utviklingsrammeverkene som kjerneapplikasjonene, og samtlige hardwarekomponenter i telefonen. Android applikasjoner skrives i Java, og kjernefunksjonalitet presenteres til utviklere via Java biblioteker. Android benytter en Dalvik Virtual Machine (VM), og hver Android applikasjon kjører som en egen prosess representert som en instans av en Dalvik VM. Linux-kjernen tilbyr underliggende funksjonalitet som tråder og lav nivå minneallokasjon til Dalvik VM.[5]

Applikasjoner laget for Android-plattformen har ikke en metode eller et punkt i koden som starter opp hele applikasjonen. Dette er i kontrast til vanlige Java applikasjoner hvor det finnes en main-metode. I stedet er det i koden essensielle komponenter som systemet vil kunne instansiere og kjøre når dette er ønskelig. Det finnes fire forskjellige typer komponenter:[6]

- **Activities:** Presenterer et visuelt brukergrensesnitt, dette kan for eksempel være en rullgardinmeny. I det en utfører et valg i rullgardinmenyen vil gjerne en ny activity utløses. Ofte vil flere activities sammen danne det helhetlige brukergrensesnittet for applikasjonen. Hver activity har et vindu til disposisjon for vise brukergrensesnittet for denne activitien.
- **Services:** En service er en programvareoppgave som kjører i bakgrunnen av applikasjonen for en bestemt tidsperiode. Dette kan for eksempel være å lytte etter innkommende data fra nettverket, og levere den mot-

[4]http://java.sun.com/, 26. april 2010
[5]http://developer.android.com/guide/basics/what-is-android.html, 19. mars 2010
[6]http://developer.android.com/guide/topics/fundamentals.html, 25 mars 2010

tatte dataen til en activity. En får tilgang til services via et grensesnitt service klassen tilbyr.

- **Broadcast recievers:** Tar imot broadcast-meldinger og utfører en handling basert på den mottatte meldingen. For eksempel kan en broadcast reciever motta en melding om at mobiltelefonen har lite batteri, eller at den ikke lengre har tilgang til det trådløse nettverket som benyttes.
- **Content providers:** Benyttes for å tilgjengeliggjøre et sett av data fra en applikasjon, slik at andre applikasjoner kan benytte dataene. Dataene kan for eksempel lagres i mobiltelefonens filsystem eller i en database. For å aksessere dataene må en applikasjon ta i bruk en content resolver, som vil kunne kommunisere med content providers.

Ved å arve fra en av disse komponentklassene vil en kunne utvikle en spesifikk komponent. Det vil si at om en ønsker å utvikle en activity må en lage en klasse med den ønskede funksjonaliteten som "extender" Activity-klassen.

For at en applikasjon skal få tilgang til sine omgivelser på mobiltelefonen har Android-plattformen et grensesnitt kalt Context. Context grensesnittet gir tilgang til systemspesifikke ressurser og klasser.[7] For å muliggjøre blant annet meldinger mellom komponenter og kommunikasjon med bakgrunn tjenester på Android-plattformen finnes det Intents. En Intent er en abstrakt beskrivelse av en operasjon som skal utføres.[8] For å kunne ta i bruk systemspesifikke omgivelser vil et objekt bli tildelt en Context systemtjeneste. Dette kan være alt fra å styre telefonens vibrering til å kontrollere det trådløse nettverkskortet. Deretter vil en kunne utløse Intents for den spesifikke Context systemtjenesten.

2.5.1 HTC Hero

Under utviklingen av lokasjonssystemet denne oppgaven baserer seg på, har det blitt tatt i bruk en HTC Hero telefon som benytter Android-plattformen. Telefonen har følgene spesifikasjoner:

- Android plattform versjon 1.5
- Qualcomm 528 MHz prosessor.

[7] http://developer.android.com/reference/android/content/Context.html, 25 mars 2010

[8] http://developer.android.com/reference/android/content/Intent.html, 25 mars 2010

- 288 MB minne.
- 802.11 b/g nettverkskort.

2.6 Objective-C

Objective-C er et programmeringsspråk som i hovedsak benyttes for å utvikle programvare for Apple-produkter. Objective-C kan ses som et utvidelse av programmeringsspråket C med hensikt å muliggjøre objektorientert programmering. Utvidelsen er gjennomført ved å inkludere Smalltalk, et av de første objektorienterte programmeringsspråkene. For Apple-applikasjoner som skrives i Objective-C benyttes det i dag en objektorientert utviklingsomgivelse kalt Cocoa. Cocoa består i hovedsak av to Objective-C rammeverk. Et rammeverk fungerer som et bibliotek, ved at det tilbyr grunnleggende utviklingsfunksjonalitet. De to Cocoa rammeverkene er[9]:

- **Foundation Kit**: Tilbyr grunnleggende programmeringsfunksjonalitet som for eksempel streng og verdi manipulasjon, iteratorer og løkker.
- **Application Kit**: Inneholder kode som muliggjør samhandling med brukergrensesnittet. Application Kit er bygget på toppen av Foundation Kit.

2.7 iPhone OS

iPhone OS er Apples operativsystem for mobiltelefoner, og benyttes for iPhone, iPod Touch og iPad. Operativsystemet er delt inn i fire abstraksjonslag[10]:

- **Cocoa Touch-laget**: Laget med høyest abstraksjonsnivå, som tilbyr grunnleggende rammeverk for utvikling av applikasjoner. Apple anbefaler å benytte Cocoa Touch-laget mest mulig når det utvikles applikasjoner, og å kun benytte de lavere liggende lagene hvis dette er nødvendig. Det er Cocoa Touch-laget som også tilrettelegger for utvikling av brukergrensesnitt og samhandling med touch-skjermen på iPhone.

[9] http://developer.apple.com/mac/library/documentation/Cocoa/Conceptual/ObjectiveC/Introduction/introObjectiveC.html, 10. mars 2010

[10] http://developer.apple.com/iphone/library/documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/Introduction/Introduction.html, 22 mars 2010

- **Media laget**: Skal tilrettelegge for best mulig grafikk, lyd og bilde opplevelse. Om rammeverkene tilbydd i Cocoa Touch-laget ikke er tilstrekkelig for mediahåndteringen i en applikasjon, skal media laget benyttes.
- **Kjerneservice laget**: Sørger for grunnleggende systemfunksjonalitet som benyttes av alle applikasjoner. Dette inkluderer blant annet rammeverk som gir enklere datahåndtering.
- **Kjerne OS laget**: Inneholder blant annet et rammeverk kalt CF-Network som gir støtte for objektorientert håndtering av nettverkskommunikasjon. Rammeverket gjør det mulig å kontrollere nettverksprotokoller uten å måtte programmere på et lavere abstraksjonsnivå. Kjerne OS laget sørger også for tilgang til hardware, mulighet for utvikling av sikkerhetsløsninger, og sentrale systemoperasjoner som minne og I/O håndtering.

Utvikling av programvare til iPhone OS foregår i en utviklingsomgivelse kalt Xcode og det skrives Objective-C kode. XCode består av flere programmer blant annet XCode IDE hvor en skriver kode, Interface Builder for å lage brukergrensesnitt og en iPhone Simulator hvor en kan test utviklede applikasjoner. Det er foreløpig ingen støtte for utvikling i Java for iPhone OS.

2.7.1 iPhone

Under utviklingen av lokasjonssystemet har det blitt benyttet en iPhone 3G med følgene spesifikasjoner[11]:

- iPhone OS versjon 3.1.2.
- Samsung ARM 11, 412 MHz prosessor.
- 128 MB minne.
- 802.11 b/g nettverkskort.

2.8 MySQL

MySQL står for My Structured Query Language, og er verdens mest brukte open source databaseløsning. Databasen som er en del av implementasjonen

[11]http://www.apple.com/iphone/specs-3g.html, 19. mars 2010

i denne masteroppgaven ble realisert i MySQL. Årsaken til dette er at en MySQL database vil fungere på de fleste plattformer. Dette gjør det mulig å flytte databasesystemet mellom servere. Om systemet skal benyttes i ettertid av denne utviklingsperioden settes det dermed ingen krav til hva slags type server dette må være. De fleste programmeringsspråk har APIer og biblioteker for å aksessere MySQL databaser. Dette forenkler prosessen med å utvikle software som skal samhandle med databasen. MySQL databaseløsningen er også kjent for å være rask, dette vil være hensiktsmessig da det er ønskelig at brukere skal få estimert sin lokasjon så fort som mulig[12].

Det ble benyttet SequelPro[13], et gratis program for Mac OS X, til å forenkle arbeidet med databasen. SequelPro gir en blant annet mulighet til å se grafiske representasjoner av innholdet i databasen.

2.9 Datamaskin

Systemet utviklet i forbindelse med denne masteroppgaven har blitt utviklet på en Macbook Pro med følgene spesifikasjoner:

- Mac OS X versjon 10.5.8.
- 2,4 Ghz Intel Core 2 Duo prosessor.
- 4 GB 1067 Mhz DDR3 minne.
- Airport extreme 802.11n nettverkskort.

Datamaskinen har blitt benyttet til utvikling av all programvare knyttet til denne oppgaven, samt under innsamlingen av lokasjonsdata. Datamaskinen har også fungert som server under hele utviklingen og testingen av systemet.

[12]http://www.mysql.com/why-mysql/, 20. mars 2010
[13]http://www.sequelpro.com/, 20. mars 2010

Kapittel 3

Implementasjon

Dette kapittelet vil beskrive hvordan lokasjonssystemet denne oppgaven baserer seg på er implementert. Kapittelet er bygget opp slik at senere avsnitt vil bygge på informasjon beskrevet i tidligere avsnitt.

3.1 Kravspesifikasjon

Kravspesifikasjonen er utarbeidet i forkant av implementasjonen for å definere hva som ønskes av systemet. Systemet er en prototype med hensikt å utforske teknologien rundt lokasjonsfingeravtrykk, det er derfor ikke tatt hensyn til hva fremtidige brukere eventuelt ville ønske av funksjonalitet i systemet. Kravspesifikasjonen er delvis definert ut i fra masteroppgavens oppgavetekst. De resterende kravene er satt opp ut i fra hva det er ønskelig at systemet skal kunne oppnå, for å kunne definere implementasjonen som velfungerende. Enkelte av de ikke-funksjonelle kravene er spesifisert ut i fra resultater identifisert under et prosjektarbeid gjennomført høsten 2009 [32]. Følgende krav er satt for systemet:

3.1.1 Funksjonelle krav

De funksjonelle kravene har som hensikt å beskrive tjenesten systemet skal levere.

- **FK1**: Det skal utvikles et lokasjonssystem for WLAN basert på lokasjonsfingeravtrykk, beregnet for innendørs bruk.

- **FK2**: Systemets design skal være distribuert, og lokasjonsestimeringen skal foregå på serversiden av systemet.
- **FK3**: Systemet skal levere lokasjonen til brukeren innenfor systemets dekningsområde. Lokasjonen skal oppgis i form av koordinater og grafisk representert på et kart.
- **FK4**: Systemet skal være nettsentrert.
- **FK5**: Systemets design skal være plattformuavhengig. Slik at det tilrettelegges for bruk av alle mobile terminaler/bærbare datamaskiner.
- **FK6**: Systemet skal være dynamisk og skalerbart med hensyn på utvidelse av dekningsområde og funksjonalitet.
- **FK7**: Lokasjoner estimert av systemet skal kunne oppgis som GPS standard koordinater.
- **FK8**: Systemet skal være modulært oppbygget for å muliggjøre gjenbruk av funksjonelle komponenter.
- **FK9**: Systemet skal ha handlingsbasert lokasjonsestimering. Med dette menes at en bruker må gjennomføre en handling for å motta sin lokasjon. Dette vil gjøre det enklere å teste systemets virkemåte etter ferdig implementasjon. Samtidig vil også en handlingsbasert lokasjonsuthenting kreve minst mulig ressurser av brukerutstyret. Dette skal ikke begrense muligheten for senere å implementere automatisk lokasjonsoppdatering.
- **FK10**: Systemet skal utvikles med hensyn på å kreve minst mulig batteri kapasitet av brukerutstyret.
- **FK11**: Systemets funksjonalitet skal tilrettelegges slik at den enkelt skal kunne benyttes av tredjeparts utviklere.

3.1.2 Ikke-funksjonelle krav

De ikke-funksjonelle kravene har som hensikt å beskrive kvaliteten som ønskes av tjenesten systemet leverer.

- **IFK1**: Dekningsområde: Systemet utviklet i forbindelse med denne oppgaven skal være en test på om lokasjonsfingeravtrykk er en egnet måte å skape et lokasjonssystem i store bygninger slik som ved NTNU. I den anledning er det essensielt at dekningsområdet for denne prototypen er representativt for slike bygninger. Dekningsområdet skal derfor

dekke typiske innendørsområder en vil finne på campus, da disse antas å være representative for store bygninger. Dette innbærer:

  - Åpne arealer, et eksempel på dette er Glassgården i El-bygget.
  - Avlukkende rom, dette innbærer blant annet lese og datasaler.
  - Korridorer.

  Samtidig skal dekningsområdet være over to etasjer. Ved å ta i bruk et slikt dekningsområde vil utfordringer relatert til bygningsmessige omgivelser bli identifisert.

- **IFK2**: Responstid: Systemet skal gjennomføre et lokasjonsestimat i løpet at ett sekund.
- **IFK3**: Nøyaktighet: 85 % av lokasjonsestimatene skal returnere en lokasjon med maksimal feilmargin på 5 meter. 75 % av lokasjonsestimatene skal returnere en lokasjon med maksimal feilmargin på 3 meter.
- **IFK4**:Nøyaktighet 2: Systemets nøyaktighet skal ikke påvirkes av hva slags mobil enhet som benyttes.
- **IFK5**: Pålitelighet: Systemet skal alltid oppgi en lokasjon så lenge brukeren befinner seg innenfor systemets dekningsområde, og den mobile enheten er koblet til det trådløse nettverket.
- **IFK6**: Brukergruppe: Systemet skal takle 1000 brukere per time, uten at dette leder til en forverring i systemets responstid.

3.2 Signalstyrketester

Det er et overordnet mål at systemet skal fungere for alle typer mobile enheter. Forskjellige mobile enheter vil være utstyrt med ulike nettverkskort og antenner. Dette vil lede til at RSSI verdiene registret fra omliggende aksesspunkt vil avhenge av hardwaren i de mobile enhetene. Lokasjonsdataene systemet baserer seg på er innsamlet ved hjelp av en datamaskin som beskrevet i avsnitt 2.9, denne datamaskinen vil heretter bli omtalt som referansemaskin. Det blir dermed essensielt å identifisere hvordan mobile enheter en ønsker skal benytte lokasjonssystemet oppfatter signalstyrker fra omliggende nettverk, sammenliknet med referansemaskinen brukt under innsamlingen av lokasjonsdata. I systemet utviklet i forbindelse med denne oppgaven har det

blitt utviklet programvare slik at Android og iPhone telefoner skal kunne benyttes i lokasjonssystemet. Det ble derfor gjennomført tester for å kontrollere hvordan slike smarttelefoner oppfattet signalstyrker fra omliggende trådløst nett sammenliknet med referansemaskinen.

3.2.1 Test 1

For å få mest mulig nøyaktige lokasjonsestimat er det ønskelig at en, ved hjelp av RSSI verdien registrert for et aksesspunkt med en smarttelefon, vil kunne estimere hvordan referansemaskinen hadde oppfattet RSSI verdien fra det samme aksesspunktet for den samme lokasjonen. Hensikten med denne signalstyrketesten var å identifisere om dette er mulig. Samtidig er det essensielt at en slik sammenheng er enkel å identifisere og lett å ta i bruk. Om det kreves store mengder signalstyrkemålinger for å identifisere en sammenheng, vil lokasjonssystemet miste mye av sin hensikt. Det vil være kostbart å gjennomføre en slik kalibrering for hver enkelt type mobile enhet en ønsker å benytte i lokasjonssystemet. Forskjellige modeller av mobile enheter vil oppfatte signalstyrker ulikt, selv om de er laget av samme produsent og har samme WLAN antenne og hardware. Årsaken til dette er at hvordan antennen er plassert i forhold til resten av hardwaren i den mobile enheten spiller en vesentlig rolle for hvordan den mobile enheten oppfatter signalstyrke, dette gjelder særlig for smarttelefoner. For smarttelefoner vil også antennens plassering, i forhold til hvordan brukeren normalt vil holde telefonen, påvirke hvordan signalstyrkene oppfattes. Ut i fra dette kan vi konkludere med at en vil kunne ende opp med et stort antall mobile enheter som må kalibreres, og en kalibrering bør derfor være enkel å gjennomføre.

For enklest mulig å identifisere forskjeller i signalstryker ble målingene gjennomført mot et aksesspunkt av gangen. Målingene for HTC-telefonen og iPhonen ble også separert, slik at RSSI målingene fra en og en smarttelefon ble sammenliknet med RSSI målinger fra referansemaskinen. For hvert aksesspunkt ble det gjennomført 27 simultane målinger for HTC-telefon og referansemaskinen. Deretter ble det gjennomført 27 simultane målinger for iPhonen og referansemaskinen. Under alle målingene ble telefonene holdt som ved naturlig bruk, og på nøyaktig samme lokasjon. Denne lokasjonen var så nær som mulig referansemaskinens WLAN antenne. Alle målingene ble gjennomført manuelt. Fluktuering i signalstyrke vil være tidsavhengig, ved å gjennomføre testene manuelt ble det forsikret om at målingene ble fordelt over en vis tidsperiode. Alle målingene ble gjort mot aksesspunkt som finnes på campus ved NTNU Gløshaugen (jf avsnitt 2.1.3). Disse er tilsva-

rende aksesspunktene som benyttes av Trådløse Trondheim.

Det ble gjennomført signalstryke tester for fire aksesspunkt. Den første aksesspunktet det ble testet for er et aksesspunkt Trådløse Trondheim har satt opp i Trondheim sentrum. Aksesspunktet er plassert utendørs mens målingene ble gjennomføre innendørs. De tre neste aksesspunktene det ble testet for er plassert på campus ved NTNU Gløshaugen. Alle målingene på campus ble gjennomført fra samme lokasjon, og det ble valgt tre aksesspunkt som tilnærmet alltid er hørbare fra denne lokasjonen. Både lokasjonen og aksesspunktene er lokalisert innendørs. Alle målingene ble gjort mot aksesspunktenes b/g brikkesett.

- **Aksesspunkt1**: Trådløse Trondheim AP med BSSID = 00:23:5d:0d:57:f4
- **Aksesspunkt2**: Campus AP med BSSID = 00:0b:85:89:b8:cd
- **Aksesspunkt3**: Campus AP med BSSID = 00:0b:85:89:b2:3d
- **Aksesspunkt4**: Campus AP med BSSID = 00:0b:85:89:b3:7d

Resultatene fra signalstyrketestene er representert i tabell 3.1. Tallene som er oppgitt er gjennomsnittet av de 27 målingene gjennomført for hvert aksesspunkt, med hver mobile enhet. Resultatene i tabellen er sortert ut i fra mottatt signalstyrke med de sterkeste RSSI verdiene først i tabellen.

	iPhone	*referansemaskin*	*differanse*	*HTC*	*referansemaskin*	*differanse*
Aksesspunkt4	-57.963	-49.593	8.37	-58.519	-49.483	9.036
Aksesspunkt2	-71.444	-67.667	3.777	-75.000	-70.333	4.667
Aksesspunkt1	-72.889	-71.889	1	-76.556	-71.519	5.037
Aksesspunkt3	-80.556	-75.926	4.63	-85.148	-76.741	8.407

Tabell 3.1: Resultater signalstyrketest 1, alle verdier oppgitt som RSSI.

3.2.2 Evaluering test 1

Hensikten med Test 1 var å identifisere om det er mulig å finne en sammenheng slik at en vil kunne estimere hvordan referansemaskinen ville ha oppfattet signalstyrken fra et aksesspunkt, basert på hvordan iPhone eller Android-telefonen oppfattet signalstryken fra dette aksesspunktet. Ut i fra resultatene av Test 1, presentert i tabell 3.1, ble ikke funnet noen slik sammenheng. Det er derfor behov for å identifisere hvordan denne utfordringen kan løses.

Refleksjon, avbøyning, spredning, multipath og fading er alle faktorer som vil være med å skape fluktuering i mottatt signalstyrke for en bestemt lokasjon. Slike påvirkninger oppstår for alt WLAN utstyr (jf. avsnitt 2.1.4). Lokasjonssystemet utviklet i forbindelse med denne oppgaven skal gi støtte for bruk av ulike mobile enheter. Forskjellig design og bruk av ulike WLAN antenner leder til at mobile enheter vil oppfatte signalstyrken fra et aksesspunkt forskjellig. Samtidig vil også posisjonen til brukeren av den mobile enheten kunne påvirke mottatt signalstyrke. Det er naturlig at brukerens påvirkning av den mottatte signalstyrken vil være større for smarttelefoner som holdes i hånden, enn for bærbare datamaskiner som normalt vil være plassert på et objekt. Alle disse faktorene har spilt en rolle for resultatet av denne signalstyrketesten. Dette leder til at disse resultatene i hovedsak kan ses som et tidsavhengig sample av hvordan signalstyrken oppfattes for en lokasjon. Det er en reell mulighet for at resultatene ville ha vært annerledes om det hadde blitt gjennomført en tidsvarende måling ved et annet tidspunkt. Ut i fra dette kan det virke som at å benytte konkrete RSSI verdier til å definere lokasjoner, i et lokasjonssystem basert på fingeravtrykk, vil kunne bli unøyaktig når systemet skal benyttes av ulike mobile enheter. Med konkrete RSSI verdier menes det at en lokasjon er definert ved for eksempel:

- *Fingeravtrykk 1*: Aksesspunkt x med RSSI verdi -48.
- *Fingeravtrykk 2*: Aksesspunkt y med RSSI verdi -72.
- *Fingeravtrykk 3*: Aksesspunkt z med RSSI verdi -79.

Resultatene i tabell 3.1 viser at differansen i mottatt signalstyrke for Android-telefonen sammenliknet med mottatt signalstyrke for referansemaskinen, er større enn differansen for iPhonen sammenliknet med referansemaskinen. Dette kan være en indikator på at å benytte konkrete RSSI verdier vil gi mer nøyaktige lokasjonsestimat iPhonen enn for Android-telefonen. Videre ser vi ut i fra tabell 3.1 at forholdet mellom de ulike aksesspunktene er det samme for alle de tre mobile enhetene. Med dette menes at aksesspunktet som ble registrert med den gjennomsnittlig sterkeste signalstyrken er det samme for alle de tre mobile enhetene, dette gjelder også aksesspunktet med den nest sterkeste signalstyrken også videre. Som et alternativ til å benytte konkrete RSSI verdier for å estimere en lokasjon, kan en derfor heller se på forholdet mellom signalstyrkene for de ulike aksesspunktene. Denne masteroppgaven er en fortsettelse av et prosjektarbeid gjennomført høsten 2009. I prosjektarbeidet ble Euklids avstand med konkrete RSSI verdier benyttet for å estimere brukerens lokasjon. Dette ga nøyaktige lokasjonsestimat når samme mobile enhet ble benyttet til å samle inn lokasjonsdata og teste systemets nøyaktighet. For å identifisere om forholdet mellom aksesspunktenes signal-

styrke vil være en mer fornuftig måte å estimere lokasjoner når ulike mobile enheter skal benytte lokasjonssystemet, har det blitt utviklet en alternativ lokasjonsbestemmende algoritme. Implementasjonen av denne algoritmen er beskrevet i avsnitt 3.5.2 (Vektingsalgoritme), og en sammenlinkning av nøyaktigheten av de to lokasjonsbestemmende algoritmene vil blir gjort i avsnitt 4.3.1.

3.2.3 Test 2

Lokasjonssystemet utviklet i forbindelse med denne oppgaven benytter fingeravtrykk fra tre aksesspunkt for å identifisere en lokasjon. Det ble derfor gjennomført en test for å registrere hvorvidt de ulike mobile enhetene oppfattet omliggende trådløs infrastruktur likt. Testen hadde som hensikt å identifisere om de samme aksesspunktene ble registrert med de sterkeste signalstyrkene, for alle de mobile enhetene. Målingene ble gjennomført manuelt på samme måte som beskrevet for test 1. Testen gikk ut på at alle de mobile enhetene leste inn BSSID og RSSI for alle hørbare aksesspunkt. Deretter ble aksesspunktene sortert ut i fra RSSI verdi. Aksesspunkt som ble registrert med en av de tre sterkeste RSSI verdiene for hver mobile enhet ble sammenliknet. Ut i fra denne testen ble det identifisert at referansemaskinen registrerer flere aksesspunkt en de to smarttelefonene. Årsaken til dette er at noen aksesspunkt benytter 5 GHz frekvensbåndet. Da smarttelefonene har 802.11 b/g nettverkskort, har de ikke mulighet til å lytte på dette frekvensbåndet. Referansemaskinen er utstyrt med 802.11n nettverkskort og vil derfor høre disse aksesspunktene.

3.2.4 Evaluering test 2

Ut i fra dette funnet ble det konkludert med at lokasjonsdatabasen opparbeidet under prosjektarbeidet høsten 2009 ikke vil gi støtte for mobiltelefoner, da mobiltelefoner utelukkende har 802.11 b/g nettverkskort. Ved å kontrollere alle aksesspunktene i lokasjonsdatabasen ble det identifisert at 27 av 81 aksesspunkt i databasen benyttet seg av 5 GHz frekvensområdet. Dette resulterer i at tilnærmet alle lokasjoner i databasen er knyttet til et fingeravtrykk levert av et aksesspunkt som benytter 5 GHz frekvensbåndet. Da det under innsamlingen av lokasjonsdata høsten 2009 ikke ble tatt hensyn til at et fysisk aksesspunkt vil kunne fungere som to imaginære aksesspunkt innenfor et nettverk (jf. avsnitt 2.1.3), vil det være en reell mulighet for at noen lokasjoner i databasen er knyttet til to fingeravtrykk fra det samme fysiske

aksesspunktet. Om dette er tilfelle vil systemet miste noe av trianguleringseffekten ved utvalg av mulige lokasjoner. Dette skjer i den sonebaserte delen av lokasjonsestimeringen (beskrevet i avsnitt 3.5.2). En ny lokasjonsdatabase vil dermed, i tillegg til å gi støtte for bruk av mobile enheter med 802.11b/g nettverkskort, også kunne gi et mer nøyaktig lokasjonssystem.

3.3 Innsamling av lokasjonsdata

I avsnitt 3.2.4 ble det identifisert at lokasjonsdatabasen fra prosjektarbeidet høsten 2009 ikke vil fungere for mobile enheter med 802.11 b/g nettverkskort, det ble derfor opparbeidet en ny lokasjonsdatabase. For den nye databasen ble det samlet inn lokasjonsdata for to etasjer av Gamle elektro samt A-blokka i El-bygget ved NTNU Gløshaugen. Under innsamlingen av lokasjonsdata ble et programvare verktøy benyttet, implementasjonen av dette verktøyet er beskrevet i avsnitt 3.7.

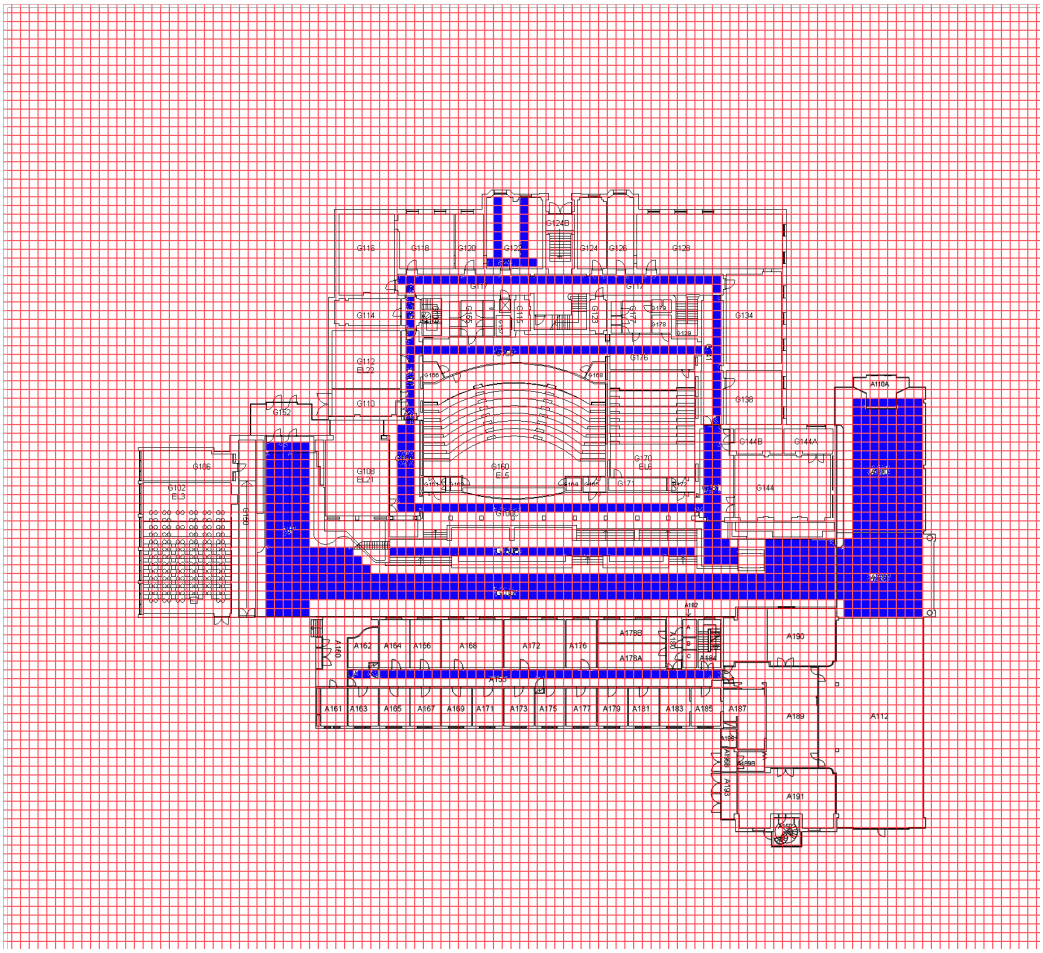
En lokasjon registreres i lokasjonsdatabasen ved at en taster inn koordinatene for lokasjonen en befinner seg ved, i brukergrensesnittet til programvareverktøyet. Verktøyet vil automatisk gjøre et skann etter informasjon fra omliggende trådløse nettverk. De tre kraftigste signalstyrkene identifiseres, samt hvilke aksesspunkt som leverte disse signalstyrkene. Disse lokasjonsdataene vil bli linket til koordinatene som ble tastet inn, for deretter å bli lagret i databasen. Aksesspunktene ved NTNU Gløshaugen vil kunne tilhøre flere virtuelle nettverk (jf. avsnitt 2.1.3). For å unngå å definere en lokasjon ved hjelp av flere fingeravtrykk fra samme fysiske aksesspunkt, vil kun aksesspunkt tilhørende nettverket med SSID lik eduroam tas i betraktning. Normalt ble det observert signalstyrker fra 8 til 12 aksesspunkt for hver lokasjon. En lokasjon ble identifisert ved å benytte et kart med rutenett hvor en kan lese av nøyaktige koordinater for en lokasjon. Rutenettet på kartet er delt opp slik at en rute på kartet tilsvarer en kvadratmeter i virkeligheten. Før innsamlingen av lokasjonsdata ble programvare verktøyet satt til å kun registrere aksesspunkt som benyttet 2,4 GHz frekvensområdet, da dette er aksesspunkter som vil kunne registreres med 802.11b/g nettverkskort. Dette ble realisert ved at det kontrolleres hvilken kanal aksesspunktet tar i bruk. For aksesspunktene på campus benyttes kanalene 1, 6 og 11 for 2,4 GHz frekvensbåndet. Alle aksesspunkt som tar i bruk kanaler over 11 ble derfor ikke tatt i betraktning under innsamlingen av lokasjonsdata.

Det er valgt et rutenett for systemet hvor hver rute tilsvarer en kvadratmeter. Det er ikke sannsynlig at systemet vil kunne oppnå denne nøyaktigheten,

men det er ønskelig å se hvor nøyaktig det er mulig å estimere lokasjoner. Samtidig vil en så omfattende innsamling av lokasjonsdata kunne være med å øke nøyaktigheten av systemet, sett fra en brukers perspektiv. For nærliggende kvadrater, i henhold til rutenettet, vil det observeres relativt like signalstyrker så lenge de er innenfor samme rom. Dette leder kanskje til at brukerens lokasjon blir estimert til et nærliggende kvadrat ved et lokasjonsestimat, men for brukeren oppfattes dette som en relativt nøyaktig gjengivelse av den reelle lokasjonen. Samtidig vil en nøyaktig innsamling av lokasjonsdata være med å eliminere bort muligheten for grove avvik mellom reell og estimert lokasjon. Under innsamlingen av lokasjonsdata kan det ha forekommet uregelmessigheter slik at en lokasjon har blitt linket til aksesspunkt, eller det har blitt registrert RSSI verdier, som normalt ikke er representative for denne lokasjonen. Ved å ha et rutenett med relativt små kvadrater minker sannsynligheten for at slike misvisende lokasjonsdata leder til et stort avvik mellom estimert og reell lokasjon. For en lokasjon hvor det har blitt registrert misvisende lokasjonsdata vil systemet sannsynligvis oppgi et en nærliggende kvadrat som brukerens lokasjon, da det er lite sannsynlig at slike unøyaktige lokasjonsdata har blitt registrert for alle nærliggende kvadrat i rutenettet. Også andre implementasjoner har benyttet et rutenett med ruter tilsvarende en kvadratmeter [14, 7].

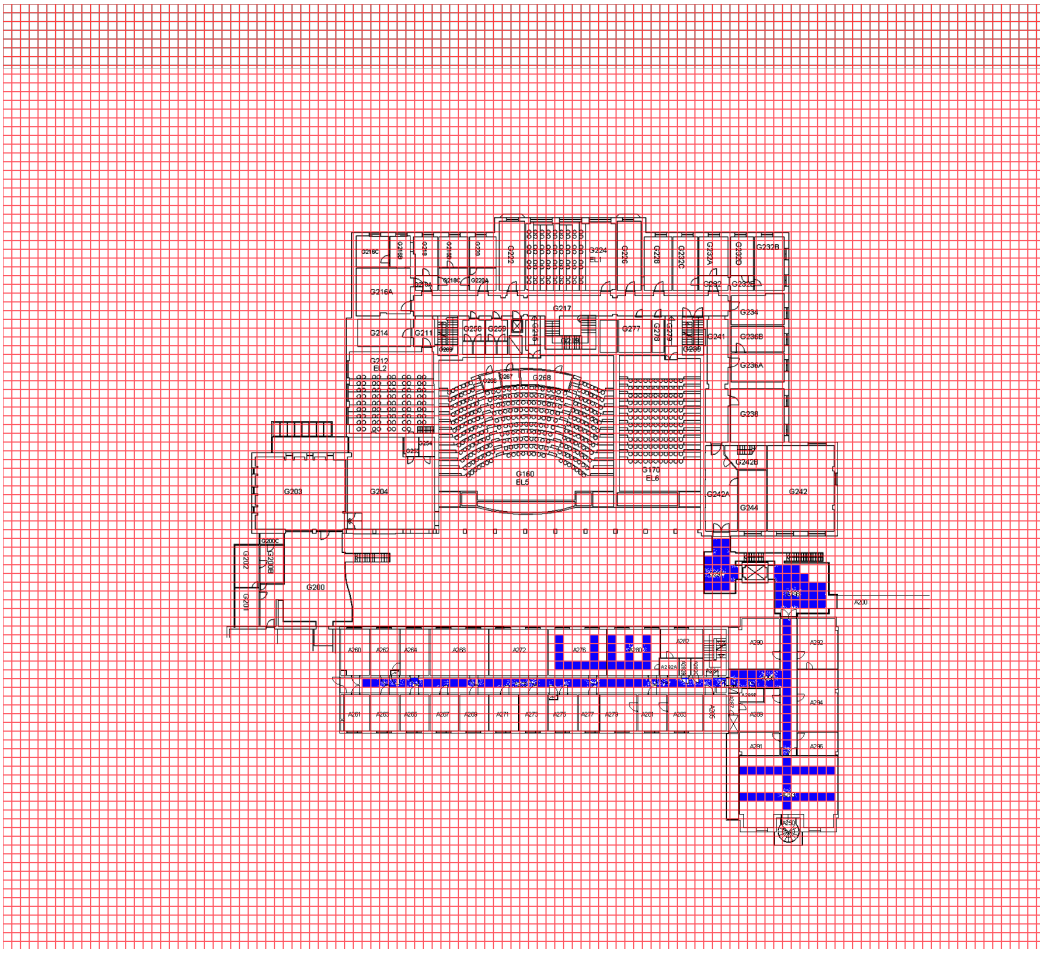
Totalt er det registrert 856 lokasjoner i databasen, det tilsvarer at det er samlet inn lokasjonsdata for 856 kvadratmeter. Figur 3.1 og 3.2 viser det konkrete dekningsområdet av systemet, for henholdsvis første og andre etasje. Med konkret dekningsområde menes hvilke kvadratmeter i koordinatsystemet det er samlet inn lokasjonsdata for. I praksis vil systemet kunne fungere for et vesentlig større område enn dette. Det vil fungere ved at lokasjoner det ikke har blitt samlet inn lokasjonsdata for, blir tildelt koordinatene til en nærliggende lokasjon når systemet brukes. Det vil, for eksempel på en lesesal, være møbler som gjør innsamling av lokasjonsdata for hver kvadratmeter kronglete. Allikevel vil hele lesesalen kunne oppgis som dekket av systemet, da det aldri vil være langt til nærliggende lokasjoner registrert i databasen. For en gang som er to meter bred er det kun registrert lokasjoner for én rekke kvadratmeter ned gangen. De kvadratmeterne av gangen det ikke er samlet inn lokasjonsdata for vil allikevel kunne regnes som dekket av systemet, da de høyst sannsynlig vil få oppgitt lokasjonen til en nærliggende kvadratmeter i den samme gangen. Ut i fra dette kan vi estimere systemets funksjonelle dekningsområde til omlag 25% større enn det konkrete dekningsområdet. Dette overslaget er gjort ut i fra Figur 3.1 og 3.2.

Det konkrete dekningsområdet benyttet under prosjektarbeidet gjennomført høsten 2009 var på 1019 kvadratmeter [32], det nye dekningsområdet er noe



Figur 3.1: Dekningsområde for første etasje.

mindre. Årsaken til dette er at det ikke ble sett som nødvendig å inkludere mange områder som under testfasen for [32] ga relativt like og stabile resultater med hensyn på nøyaktigheten av et lokasjonsestimat. Dette betyr i hovedsak at det ikke er inkludert like mange korridorer. I prosjektarbeidet ble det bevist at korridorer er områdene med de mest nøyaktige og stabile lokasjonsestimatene, og det ble heller ikke registrert noen rom- eller etasjefeil i korridorene [32]. Det ble derfor under den nye innsamlingen av lokasjonsdata fokusert på å inkludere områder hvor det ble registrert utfordringer ved lokasjonsestimeringene. Samtidig er det inkludert områder som ga nøyaktige lokasjonsestimat, for å kontrollere nøyaktigheten av lokasjonssystemet ved bruk av ulike mobile enheter i disse områdene. I prosjektarbeidet ble tiden brukt for å samle inn lokasjonsdata for dekningsområdet estimert til to dagsverk, og det ble antatt at denne tidsperioden ville kunne kortes ned



Figur 3.2: Dekningsområde for andre etasje.

ved trening. Til sammenlikning tok den nye innsamlingen av lokasjonsdata i underkant av 1,5 dagsverk.

På kartet med rutenettet som ble benyttet under innsamlingen av lokasjonsdata er det definert et GPS referansepunkt. Dette punktet er i koordinatsystemets origo. Det vil for GPS referansepunktet være mulig å gjennomføre et lokasjonsestimat ved hjelp av GPS. Det er forsikret om dette ved at lokasjonen til GPS referansepunktet er adskilt fra friluft med kun store vinduer, det vil dermed være mulig å gå utenfor vinduet for å gjennomføre et GPS lokasjonsestimat. Hensikten med dette er at en da kan omgjøre lokasjonsløsningens koordinatsystemet til standardiserte GPS-koordinater. Om systemets dekningsområdet skal omfatte flere bygninger vil det være naturlig å danne separerte koordinatsystem for hver bygning. Dette vil forenkle innsamlingen

av lokasjonsdata. For å oppgi brukeren lokasjon om dekningsområdet er over flere bygninger, kan koordinatene gjøres om til et universelt format slik som for GPS, eller en kan introdusere en fjerde koordinat for hver lokasjon. Denne koordinaten vil beskrive bygningen brukeren befinner seg i.

3.4 Overordnet design

Dette avsnittet skal gi en overordnet forståelse av virkemåten til lokasjonssystemet. En mer detaljert beskrivelse av lokasjonssystemets implementasjonen er å finne i senere avsnitt.

Kommunikasjon mellom systemets server og klientene som benytter seg av lokasjonstjenesten er en essensiell funksjonalitet for lokasjonssystemet virkemåte. Lokasjonssystemet skal støtte bruk at forskjellige mobile enheter, det er derfor vesentlig at det velges en universell kommunikasjonsløsning som vil kunne fungere for alle mobile enheter. Samtidig er det viktig at kommunikasjonen mellom klientene og serveren blir gjennomført raskt. Lokasjonssystemet tar derfor i bruk socketkommunikasjon. Socket APIer er nettverk standarden for kommunikasjon over TCP/IP, og de fleste operativsystemer støtter socket APIer[1]. Ved å ta i bruk sockets oppnår en dermed en tilnærmet plattform uavhengig kommunikasjonsløsning. Socketkommunikasjon vil også være raskere enn for eksempel "Remote Method Invocation" for Java, eller tilsvarende "Distributed Objects" for Objective-C[2].

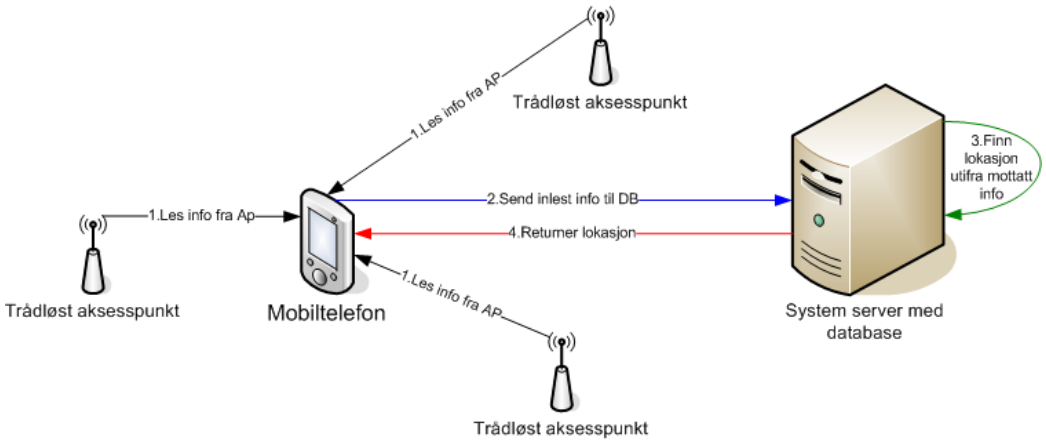
Lokasjonssystemet skal kunne støtte bruk av ulike mobile enheter. Det er derfor essensielt at alle delene av lokasjonssystemet benytter samme format for felles verdier slik som lokasjonsdata. Hvilket format som skal benyttes for lokasjonsdata bestemmes av databaseløsningen på serversiden av systemet, da databasen utgjør kjernen av lokasjonstjenesten. Databasen tar derfor i bruk de mest universelt benyttede formatene. Det blir dermed nødvendig at mobile enheter som skal benytte lokasjonssystemet tilpasser seg denne måten og uttrykke lokasjonsdata.

- **SSID:** SSID er beskrevet ved hjelp av en normal tekststreng. Dette vil støttes av alle mobile enheter og byr dermed ikke på noen utfordringer.

[1] http://publib.boulder.ibm.com/infocenter/iseries/v5r3/index.jsp?topic=/rzab6/rzab6soxoverview.htm, 7. mai 2010

[2] http://dev.fyicenter.com/Interview-Questions/Socket-2/RMI_vs_Sockets_and_Object_Serialization.html, 7. mai 2010

- **BSSID:** Det finnes flere mulige måter å oppgi BSSID. Databasen benytter et heksadesimalt format for å beskrive BSSIDene (jf. avsnitt 2.1.1), hvor hver tallgruppering alltid består av to tall og hver tall gruppering er adskilt ved hjelp av kolon. Dette er den mest vanlige format for BSSID.
- **RSSI:** RSSI vil kunne oppgis forskjellig avhengig av leverandør (jf. avsnitt 2.1.2). Databasen benytter en RSSI skala fra fra 0 til -100, der 0 representere den teoretisk sterkeste signalstyrken det er mulig og oppnå. Benevningen for RSSI er dBm.



Figur 3.3: Overordnet design.

Den overordnede virkemåten av lokasjonssystemet er vist i figur 3.3, og vil her bli beskrevet nærmere:

1. En mobil enhet leser inn lokasjonsdata fra omliggende trådløse nettverk. Fingeravtrykkene i lokasjonsdatabasen er kun definert ut i fra aksesspunkt tilhørende det virtuelle nettverket eduroam (jf. avsnitt 3.3). Lokasjonsdata tilhørende andre nettverk blir derfor forkastet. Aksesspunktene tilhørende eduroam nettverket blir deretter sortert ut i fra RSSI verdi.
2. Den mobile enheten setter opp en TCP socketforbindelse med serveren. Deretter sendes BSSID samt registrert RSSI for de tre aksesspunktene med sterkest RSSI verdi over socketforbindelsen. Her er det essensielt at informasjonen sendes på det formatet serveren ønsker, slik som beskrevet tidligere i avsnittet. Informasjonen sendes over socketforbindelsen som en tekststreng, det er også essensielt at denne tekststrengen er på det formatet serveren forventer. BSSID og RSSI for et aksess-

punkt ligger etter hverandre i strengen. Hver verdi er adskilt med et mellomrom. Formatet for strengen som sendes til serveren blir dermed: *BSSID_ aksesspunkt1 RSSI_ aksesspunkt1 BSSID_ aksesspunkt2 RSSI_ aksesspunkt2 BSSID_ aksesspunkt3 RSSI_ aksesspunkt3.*

3. Serveren estimerer lokasjonen til den mobile enheten ut i fra de mottatte dataene. Dette er realisert ved hjelp av Java programvare med tilgang til lokasjonsdatabasen. Detaljene av en lokasjonsestimering er beskrevet i avsnitt 3.5.

4. Serveren sender den estimerte lokasjonen tilbake til klienten. Lokasjonen sendes som en tekststreng bestående av tre koordinater separert ved mellomrom. Formatet på lokasjonsestimatet som sendes fra serveren blir dermed:*X_ koordinat Y_ koordinat Z_ koordinat.* Tilslutt vil klienten vise den estimerte lokasjonen i sitt brukergrensesnitt.

3.5 Serverløsning

Lokasjonssystemets server har som oppgave å estimere en lokasjon ut i fra lokasjonsdata mottatt fra klienter. Serverløsningen består av to hovedkomponenter, en database med lokasjonsdata og programvare for å estimere lokasjoner. Serveren er implementert på en datamaskin som beskrevet i avsnitt 2.9.

3.5.1 Lokasjonsdatabase

Databasen har som oppgave å skape en forbindelse mellom lokasjoner, i form av koordinater, og lokasjonsdata. Med lokasjonsdata menes BSSID og RSSI for aksesspunkter registrert for en lokasjon. Et fingeravtrykk i systemet består av: En unik ID for dette spesifikke fingeravtrykket, en RSSI verdi, en link til aksesspunktet som leverte RSSI verdien, og en link til lokasjonen fingeravtrykket refererer til. En lokasjon kan i prinsippet være definert av så mange fingeravtrykk en ønsker, men antallet fingeravtrykk vil kunne påvirke nøyaktigheten av systemet. Ved å ta i bruk få fingeravtrykk kan en risikere vanskeligheter ved å identifisere riktig lokasjon. Dette skyldes at det da er få verdier benyttet for å unikt identifisere denne lokasjonen. Dermed øker sannsynligheten for at en misvisende lokasjon passer bedre overens med lokasjonsdataene klienten registrert fra omliggende trådløse nettverk.

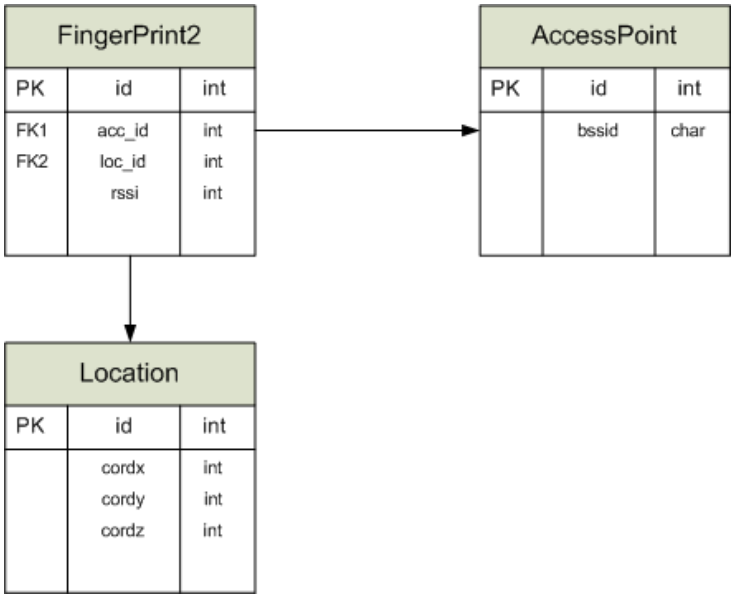
For en klient vil RSSI verdiene som registreres variere tidsavhengig for samme lokasjon. Disse variasjonene vil i stor grad være vanskelige å forutse (jf. avsnitt 2.1.4). Ved å gjennomføre noen nettverksskann for en bestemt lokasjon, ser en at det er stor utskiftning av aksesspunkt som gir svært svake RSSI verdier. På et tidspunkt vil en kunne høre et aksesspunkt, mens ved neste skann er ikke aksesspunktet lengre hørbart. Dette fenomenet oppstår selv ved nettverksskann med kun sekunders mellomrom. RSSI verdiene for aksesspunktene som gir sterke signalstyrker vil ikke være mer stabile, men aksesspunktene vil tilnærmet alltid være hørbare. En kan dermed konkludere med at å knytte for mange fingeravtrykk opp mot en lokasjon, også vil kunne lede til vanskeligheter med å identifisere riktig lokasjon. Om et for stort antall fingeravtrykk tas i bruk, vil lokasjoner bli definert ved hjelp av aksesspunkter som for denne lokasjonen ikke alltid er hørbare. En risikerer dermed at aksesspunkt som skal identifisere en lokasjonen ikke er hørbare i det lokasjonsestimatet gjennomføres. Det å knytte mange fingeravtrykk til en lokasjon vil også lede til at estimeringen av riktig lokasjon kan bli en tyngre oppgave, og dermed kreve mer prosessorkraft. I systemet laget i forbindelse med denne oppgaven blir det benyttet tre fingeravtrykk for å identifisere en lokasjon.

Et ER-diagram for databaseløsningen er vist i figur 3.4. Tabellen AccessPoint lagrer BSSIDen for aksesspunkt som er knyttet til lokasjoner i systemet, og gir hver AccessPoint entitet en ID som benyttes som primærnøkkel. BSSIDen vil også være unik for hver AccessPoint entitet, men en slik ID verdi er enklere å håndtere ved databaseinteraksjon. Hver AccessPoint entitet kan være med å definere flere lokasjoner. Lokasjoner er representert i databasen som entiteter i tabellen Location. Hver Location entitet inneholder en ID som primærnøkkel samt lokasjonens koordinater. Tabellen Fingerprint2 inneholder alle fingeravtrykkene i databasen, og skaper en forbindelse til AccessPoint og Location entiteter ved at deres primærnøkler er lagret som fremmednøkler i FingerPrint2 entiteter. En entitet i FingerPrint2 inneholder også en ID primærnøkkel, samt RSSI verdien for dette fingeravtrykket.

3.5.2 Lokasjonsbestemmende algoritme

Med lokasjonsbestemmende algoritme menes fremgangsmåten for hvordan serveren skal finne lokasjonen i databasen, som mest sannsynlig er klientens lokasjon, ut i fra ut i fra de mottatte lokasjonsdataene.

Mottatt signalstyrke vil fra et aksesspunkt vil variere tidsavhengig for den samme lokasjonen, dette skyldes omgivelsenes påvirkning av signalene (jf.



Figur 3.4: ER-diagram for databaseløsning.

avsnitt 2.1.4). Samtidig vil også forskjellige mobile enheter oppfatte signalstyrker forskjellig, avhengig av hvordan de er designet og hvordan hardware de inneholder (jf. avsnitt 3.2). Dette betyr i praksis at det er lite sannsynlig at RSSI verdiene lagret i databasen for en lokasjon, vil være identiske med RSSI verdiene lest inn fra en mobil enhet ved et vilkårlig tidspunkt for den samme lokasjonen. I en stor database med mange lokasjoner er det høyst sannsynlig at RSSI verdiene lest inn på et vilkårlig tidspunkt vil stemme bedre overens med RSSI verdiene lagret for en annen lokasjon. Det blir derfor eliminert bort flest mulig av lokasjonene i databasen før det blir gjennomført en sammenlikning av RSSI verdier. Denne elimineringen blir kalt sonebasert lokasjonsestimering.

Sonebasert lokasjonsestimering

Serveren vil via en socketforbindelse motta BSSID og RSSI for de tre aksesspunktene registrert med sterkest RSSI verdi hos en klient. Vi kaller for enkelhet skyld disse klient-aksesspunkt. Lokasjoner i databasen er knyttet til tre aksesspunkt identifisert ved deres BSSID. Vi kaller disse aksesspunktene database-aksesspunkt. Først vil det undersøkes om det finnes lokasjoner hvor alle tre database-aksesspunktene, knyttet til denne lokasjonen, har tilsvarende BSSID som for de tre klient-aksesspunktene. Om det finnes en eller flere

slike lokasjoner vil kun disse bli behandlet videre. Hvis dette ikke er tilfelle, vil lokasjoner knyttet til to database-aksesspunkt med tilsvarende BSSID som for to av de tre klient-aksesspunktene velges ut. Hvordan dette er implementert i detalj er beskrevet i avsnitt 3.5.3. Ved hjelp av denne elimineringen har det blitt skapt et sonebasert lokasjons system, hvor det defineres hvilken sone en befinner seg i ut i fra hvilke aksesspunkt som kan høres fra lokasjonen til klientens mobile enhet.

I avsnitt 3.2.1 ble det identifisert at å benytte konkrete RSSI verdier for å identifisere en lokasjon, vil kunne bli unøyaktig i et lokasjonssystemet som skal støtte bruk av ulike mobile enheter. Lokasjonssystemet inneholder derfor to ulike lokasjonsbestemmende algoritmer, Euklids avstand og en vektingsalgoritme. Vektingsalgoritme vil ta hensyn til forholdet mellom aksesspunktenes signalstyrker, mens Euklids avstand benytter konkrete RSSI verdier for å estimere en lokasjon. For begge algoritmene vil det først bli gjennomført en sonebasert lokasjonsestimering som beskrevet i dette avsnittet.

Euklids avstand

Euklids avstand er beskrevet i avsnitt 2.2. Algoritmen benyttes til å estimere brukerens lokasjon etter at den sonebaserte lokasjonsestimeringen har blitt gjennomført. Euklids avstand tar i bruk konkrete RSSI verdier, og for dette lokasjonssystemet vil beregningen dermed se slik ut:

$$Euklidsavstand = \sqrt{(KA_1RSSI - FA_1RSSI)^2 + (KA_2RSSI - FA_2RSSI)^2 + (KA_3RSSI - FA_3RSSI)^2}$$

I likningen står KA for klient-aksesspunkt og FA står for fingeravtrykk. BSSIDen og RSSI verdien til klient-aksesspunktene er mottatt fra klienten over socketforbindelsen. BSSIDen for KA_1 tilsvarer BSSIDen for aksesspunktet linket til FA_1, BSSIDen for KA_2 tilsvarer BSSIDen til aksesspunktet linket til FA_2 også videre.

Det er tre hovedårsaker til at Euklids avstand antas å være egnet som lokasjonsbestemmende algoritme:

1. Euklids avstand er en enkel matematisk algoritme som vil kreve lite ressurser i form av prosessorkraft. Dette vil være med å skape et tidseffektivt lokasjonssystem.
2. Annen forskning på lokasjonssystemer basert på lokasjonsfingeravtrykk har vist at Euklids avstand er en velegnet algoritme [7, 2, 24, 14, 32, 21]. Dette innebærer blant annet et system der implementasjonen har foregått på en type smarttelefoner [21].

3. Andre alternative algoritmer har ikke vist noen ytelses forbedring sammenliknet med Euklids avstand. Kamol Kaemarungsi et al. beskriver at som et alternativ til Euklids avstand kan en benytte algoritmer som tar i bruk Bayesian modellering eller "neural networks", men nøyaktigheten av lokasjonssystemet vil være tilnærmet identisk [14]. Phongsak Prasithsangaree et al. har sammenliknet Euklids avstand med Manhattan avstand, og identifiserte heller ingen forskjell i nøyaktigheten av lokasjonssystemet [25].

Det er ikke funnet forskning på lokasjonssystemer, basert på lokasjonsfingeravtrykk, der det undersøkes hvordan Euklids avstand vil fungere ved bruk av ulike mobile enheter. I signalstyrketest 1 (jf. avsnitt 3.2.1) ble det identifisert at ytelsen av Euklids avstand vil kunne bli svekket i et slik senario, det er derfor et behov for å undersøke dette nærmere.

Vektingsalgoritme

Hensikten med denne vektingsalgoritmen er å undersøke om en ved å ta hensyn til forholdet mellom aksesspunktenes signalstyrker oppnår en mer nøyaktig lokasjonsbestemmende algoritme enn Euklids avstand, for et lokasjonssystem som støtter bruk av ulike mobile enheter.

Etter den sonebaserte delen av lokasjonsestimatet har blitt gjennomført vil en ha estimert en sone brukeren kan befinne seg i bestemt av en delmengde lokasjoner fra lokasjonsdatabasen. Vektingsalgoritmen vil gjennomføre en vekting av disse lokasjonene for ytterligere å begrense sonen brukeren kan befinne seg i. Vektingen vil baseres på rekkefølgen av aksesspunktene når de er sortert ut i fra mottatt signalstyrke. Dette vil foregå på følgende måte:

1. Det blir undersøkt om det finnes lokasjoner hvor det tilhørende fingeravtrykket med den sterkeste RSSI verdien er knyttet til et databaseaksesspunkt med samme BSSID som klient-aksesspunktet med sterkest RSSI verdi. Hvis dette er tilfelle blir lokasjonens vekting økt med fire.
2. Det blir undersøkt om det finnes lokasjoner hvor det tilhørende fingeravtrykket med den nest sterkeste RSSI verdien er knyttet til et databaseaksesspunkt med samme BSSID som klient-aksesspunktet med nest sterkest RSSI verdi. Hvis dette er tilfelle blir lokasjonens vekting økt med to.
3. Det blir undersøkt om det finnes lokasjoner hvor det tilhørende fingeravtrykket med den svakeste RSSI verdien er knyttet til et database-

aksesspunkt med samme BSSID som klient-aksesspunktet med svakeste RSSI verdi. Hvis dette er tilfelle blir lokasjonens vekting økt med en.

En lokasjon vil dermed kunne oppnå en vekting fra 7 til 0, der en vekting på 7 oppnås når rekkefølgen av aksesspunktene sortert ut i fra RSSI verdi er den samme for database-aksesspunktene som for klient-aksesspunktene. Vekting løsningen vil fungere på nøyaktig samme måte om den sonebaserte delen av lokasjonsestimatet kun finner lokasjoner hvor to av database-aksesspunktene, knyttet til lokasjon, tilsvarer to av de tre klient-aksesspunktene. Hvis dette er tilfelle vil en lokasjon kunne oppnå en vekting fra 6 til 0. Algoritmen vil registrere den høyeste vektingen som har blitt oppnådd, kun lokasjoner som har oppnådd denne vektingen vil bli vurdert videre.

For at nøyaktigheten av vektingsalgoritmen skal kunne sammenliknes med Euklids avstand må brukerens lokasjon oppgis på koordinatform. Hvis det er flere lokasjoner som har oppnådd den høyeste vektingen vil nøyaktigheten av systemet fortsatt være sonebasert, men sonen har blitt ytterligere avgrenset. Det vil i så fall bli beregnet Euklids avstand for disse lokasjonene. Vektingsalgoritmen sørger altså for at det beregnes Euklids avstand for et mindre antall lokasjoner, enn ved normal bruk av Euklids avstand. Konkrete RSSI verdier vil dermed være mindre avgjørende under estimeringen av brukerens lokasjon.

3.5.3 Programvare

Dette avsnittet vil forklare hvordan programvaren på serveren er implementert. Serverprogramvaren består av følgene klasser og er realisert i Java:

MainSystem

MainSystem har som oppgave å starte opp serverprogramvaren. Dette gjøres ved at klassen inneholder systemets main-metode. Videre binder MainSystem serverens funksjonelle komponenter sammen, slik at det kan gjennomføres lokasjonsestimat. MainSystem inneholder en DatabaseReader, en Calculator og en NetworkIO instans. I MainSystems konstruktør opprettes det en forbindelse til lokasjonsdatabasen, og det settes opp en serversocket. Serverprogramvaren benytter port nummer 8888, og den vil lytte på denne porten til programvaren avsluttes.

AccessPoint

Denne klassen er laget for å kunne representere aksesspunkt i programvaren. Et AccessPoint objekt inneholder en streng for å lagre BSSID, en streng for å lagre SSID, samt et heltall for å lagre RSSI verdien registrert for aksesspunktet. AccessPoint objekter benyttes i all programvare utviklet i forbindelse med lokasjonssystemet, det vil si for serverløsning, verktøyet for innsamling av lokasjonsdata, og klientløsningene for Android og iPhone.

NetworkIO

NetworkIO muliggjør kommunikasjon med klienter som benytter seg av lokasjonssystemet. Klassen inneholder metoder for å opprette, lytte på, og avslutte en socket. Java.net er et API for nettverkskommunikasjon som inneholder to typer sockets for kommunikasjon over TCP, disse er kalt *ServerSocket* og *Socket*. En *ServerSocket* åpner en socket på en bestemt port og lytter på denne porten. Det er denne typen socket som benyttes i serverprogramvaren. Klientprogramvaren inneholder *Sockets*. En *Socket* kan opprette en kommunikasjonsforbindelse med serverprogramvaren ved og sende en forespørsel til porten *ServerSocketen* lytter på. Figur 3.5 viser hvordan samhandling mellom disse socket typene er implementert, dette er en connection-oriented løsning som tar i bruk TCP protokollen for dataoverføring. Dette avsnittet vil beskrive serversocket funksjonaliteten ut i fra figur 3.5.

- **Socket()** Det opprettes et socket endepunkt og defineres hvilken protokoll som skal benyttes for overføring av data. Dette er funksjonalitet som blir håndtert av Java. Ved å benytte en instans av klassen ServerSocket oppnås en stream socket som tar i bruk TCP for dataoverføring, mens en instans av klassen DatagramSocket vil ta i bruk UDP.
- **Bind()** Det spesifiseres hvilken port socket endepunktet vil benytte seg av, i denne implementasjonen tas port nummer 8888 i bruk. Dette er en port satt av til webtjenester for Macintoshmaskiner[3]. Lave portnummer benyttes gjerne til grunnleggende tjenester på maskinen som HTTP, SNMP og liknende. Socket endepunktet er nå instansiert og klart til bruk.
- **Listen()** Etter socket endepunktet er satt opp vil programvaren lytte etter klienter som etterspør et lokasjonsestimat. Før selve dataoverføringen mellom klient og server kan gjennomføres må det settes opp en

[3]http://support.apple.com/kb/ts1629, 20 april 2010

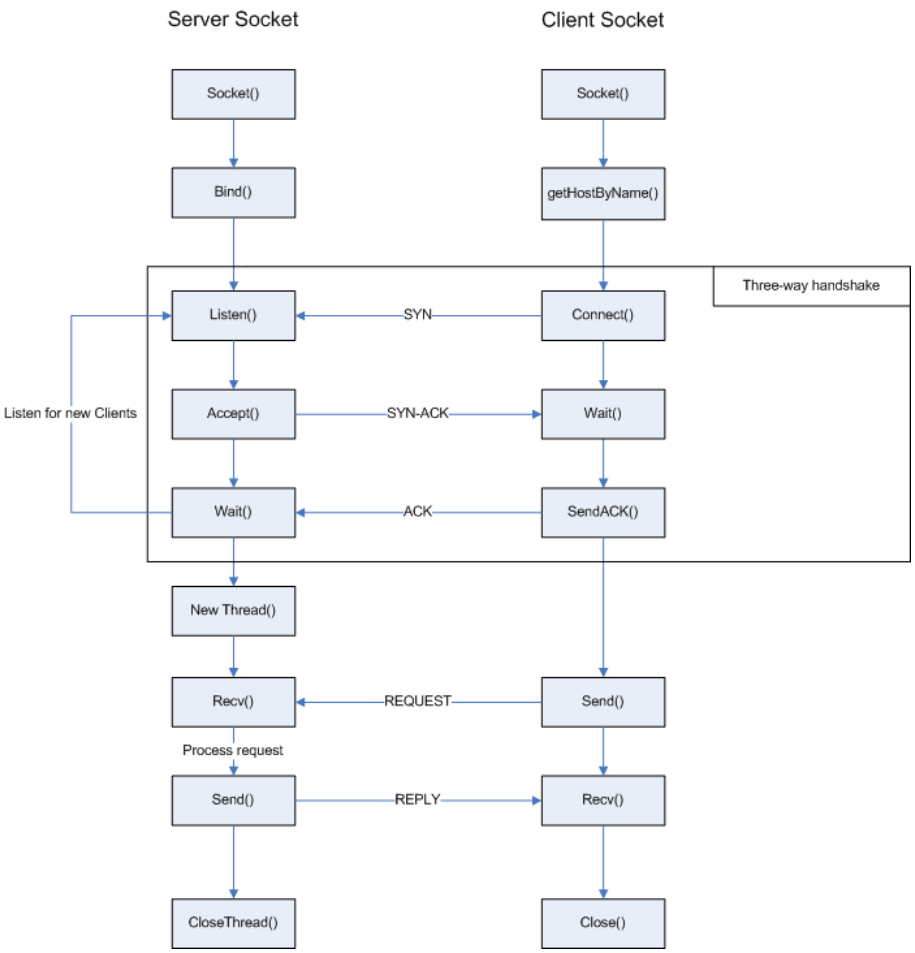
forbindelse mellom de to enhetene, dette gjøres som et treveis håndtrykk. Selve implementasjonen av håndtrykket er skjult for en Java-utvikler, men er vist i figur 3.5 for å gi en mer utdypende beskrivelse av hvordan socketforbindelsen fungerer. Tilstanden *Listen()* vil lytte etter SYN meldinger fra klienter. En SYN melding inneholder et sekvensnummer som settes til et tilfeldig tall A.

- **Accept()** I det serveren har mottatt en SYN melding vil den generere en SYN-ACK melding som sendes tilbake til klienten. SYN-ACK meldingen inneholder et bekreftelsesnummer som settes til en mer enn det mottatte sekvensnummeret, det vil si (A+1). Meldingen inneholder også et nytt sekvensnummer som settes til et tilfeldig tall B.
- **Wait()** Serveren vil vente på et ACK melding fra klienten som vil fullføre det treveis håndtrykket. ACK meldingen inneholder et bekreftelsesnummer som settes til en mer en sekvensnummeret fra SYN-ACK meldingen, det vil si (B+1). I det ACK meldingen mottas hos serveren er kommunikasjonslinken mellom klient og server etablert og klar til bruk [29].
- **New Thread()** For at serveren skal kunne betjene flere klienter samtidig vil det, etter socket forbindelsen er etablert, opprettes en ny tråd for hver klient. En tråd er realisert ved hjelp av en instans av klassen ClientHandler, og har ansvaret for resten av kommunikasjonen med klienten den betjener. De resterende delene av figur 3.5 tilhører dermed klassen ClientHandler og vil bli beskrevet i neste avsnitt.

ClientHandler

En ClientHandler tråd har som oppgave gjennomføre et lokasjonsestimat for en spesifikk klient. Serverprogramvaren vil kunne ha flere ClientHandler instanser kjørende i parallell, slik at flere klienter vil kunne få gjennomført lokasjonsoppslag samtidig.

- **Recv()** Tråden vil motta en tekststreng med lokasjonsdata fra klienten den behandler, sendt over socketforbindelsen. Formatet for denne tekststrengen er beskrevet i avsnitt 3.4. Dataene i tekststrengen vil omformes til AccessPoint objekter, og AccessPoint objektene legges i en ArrayList. ClientHandler instansen vil deretter be MainSystems DatabaseReader om å gjennomføre et lokasjonsestimat basert på lokasjonsdataene lagret i ArrayListen.



Figur 3.5: Socket flytdiagram for lokasjonsløsningen.

- **Send()** Resultatet av lokasjonsestimatet sendes tilbake klienten over socketforbindelsen som en tekststreng. Formatet for denne tekststrengen er også beskrevet i avsnitt 3.4.
- **CloseThread()** Etter en ClientHandler instans har behandlet en lokasjonsestimatforespørsel vil den termineres. En klient vil derfor bli tildelt en ny ClientHandler instans for hvert lokasjonsestimat. Dette er for å unngå at det ligger ubrukte tråder kjørende i bakgrunnen på serveren. Ved store brukergrupper ville slike ubrukte bakgrunnstråder kunne lede til unødvendig bruk av ressurser på serveren.

Hvordan kommunikasjon over en socketforbindelse er implementert er nærmere beskrevet i avsnitt 3.6.2, under løsningen for Android. Da både serveren og Android klientløsningen er utviklet i Java vil kommunikasjonsløsningene

være tilnærmet identisk implementert på begge sider av socketforbindelsen.

DatabaseReader

DatabaseReader tar seg av all kommunikasjon med MySQL-databasen. I tillegg til å ha metoder for å koble til og fra databasen inneholder klassen metoden *findLocation*. *findLocation* utgjør strukturen av og sentral funksjonalitet for å gjennomføre et lokasjonsestimat, denne metoden vil derfor bli beskrevet nærmere. Det refereres til figur 3.4 for en beskrivelse av databasestrukturen.

findLocation: *findLocation* Tar inn en ArrayList med AccessPoint objekter. Disse AccessPoint objektene inneholder dataene ClientHandler har mottatt fra klienten den behandler. *findLocation* vil deretter generere en subspørring som finner alle $AccessPoint.id$ i lokasjonsdatabasen, hvor BSSID verdien for disse entitetene tilsvarer BSSID verdien for en av de tre AccessPoint objektene i ArrayListen. Resultatet av denne subspørringen blir sammenliknet med $FingerPrint2.acc\_id$ i en hovedspørring, som henter ut $FingerPrint2.loc\_id$. Resultatene blir gruppert ut i fra $FingerPrint2.loc\_id$, og det telles antallet entiteter for hver gruppering. Om det finnes grupperinger med tre entiteter, vil det si at det er funnet en eller flere lokasjoner, definert av samtlige av de tre aksesspunktene klienten registrerte med sterkest RSSI verdi*. Hvis dette er tilfellet hentes det fra lokasjonsdatabasen, for hver lokasjon, ut RSSI verdiene for de tre fingeravtrykkene som definerer lokasjonen. Det dannes Location objekter for hver lokasjon. Et Location objekt inneholder tre AccessPoint objekter, disse representerer lokasjonsdataene som definerer lokasjonen i databasen. Et Location objekt inneholder også en ID som er primærnøkkel for denne lokasjonen i databasen. Deretter legges alle Location objektene i en ArrayList kalt locations. *findLocation* vil deretter be Calculator instansen i MainSystem utføre en av de to lokasjonsbestemmende algoritmene for lokasjonene i locations ArrayListen, sammenliknet med dataene ClientHandler mottok fra klienten. De lokasjonsbestemmende algoritmene i Calculator vil returnere IDen til Location objektet estimert til mest sannsynlig å være brukerens lokasjon, ut i fra lokasjonsdataene mottatt fra klienten. Tilslutt vil *findLocation* gjøre et database oppslag for å finne koordinatene tilhørende denne lokasjonen. Metoden *findLocation* returnerer disse koordinatene.

*Om det ikke finnes grupperinger i hovedspørringen som inneholder tre entiteter, velges det grupperinger med to entiteter. For slike grupperinger er det funnet en match mellom to av database-aksesspunktene linket til en lokasjon, og to av de tre AccessPoint objektene dannet ut i fra informasjonen mottatt fra klienten. Resten av fremgangsmåten vil være tilsvarende som be-

skrevet over, med ett unntak. Det vil kun bli beregnet Euklids avstand for aksesspunktene med identiske BSSID verdier.

Calculator

Klassen Calculator inneholder implementasjonene av de lokasjonsbestemmende algoritmene (jf. avsnitt 3.5.2). Ved hjelp av metoden *euclidianDistance* beregner klassen Euklids avstand, mens vektingsalgoritmen er realisert via metoden *locationRating*. De lokasjonsbestemmende algoritmene er separert ut i en egen klasse for å få en mer oversiktlig kode. Begge metodene tar inn en Arraylist med Location objekter og en ArrayList med AccessPoint objekter, slik det er beskrevet under avsnittet omhandlende metoden *findLocation*. Ut i fra verdiene som ligger i AccessPoint objektene i hvert Location objekt, blir lokasjonen mest nærliggende de reelle verdiene mottatt fra klienten estimert. Hvordan algoritmene estimerer lokasjoner ble beskrevet i avsnittet lokasjonsbestemmende algoritme (jf. 3.5.2). De lokasjonsbestemmende algoritmene vil returnere IDen til Location objektet som har blitt estimert til klientens lokasjon.

Når det beregnes Euklids avstand kan en risikere at ikke alle aksesspunktenes BSSIDer fra Location objektet samsvarer med de reelle aksesspunktenes BSSIDer. Hvis dette er tilfelle blir RSSI verdiene for aksesspunktene hvor det ikke blir funnet noen match satt til null. Det blir dermed beregnet Euklids avstand for et matematisk todimensjonalt rom. På denne måten er det forsikret om at kun aksesspunktene som definerer lokasjonen påvirker resultatet av avstandsberegningen.

3.6 Klientløsning

Klientløsningene for lokasjonssystemet består utelukkende av programvare. For at en mobil enhet skal kunne benytte lokasjonssystemet må den har tilgang til trådløse nett (WLAN hardware), og den må kunne kjøre tredjepart programvare. For å bevise at lokasjonssystemet fungerer for ulike typer mobile enheter, er det implementert klientløsninger for iPhone OS og Androidplattformen. Systemet vil også fungere for bærbare datamaskiner, dette ble bevist under prosjektarbeidet utført høsten 2009 [32]. Om det er ønskelig å benytte mobile enheter som kjører på en annen plattform i lokasjonssystemet, må det implementeres kode for disse. Fremgangsmåten for å utvikle klient programvare vil være lik for alle plattformer. Programvaren må kunne

identifisere omliggende aksesspunkt identifisert ved SSID og BSSID, samt registrere RSSI verdier for hvert aksesspunkt. Den mobile enheten må deretter kunne sette opp en socketforbindelse med lokasjonssystemets server. De to klientløsningene utviklet i forbindelse med denne masteroppgaven er objektorienterte. Disse to kjernefunksjonalitetene har derfor blitt separert ut i to klasser NetworkReader og NetworkIO.

3.6.1 NetworkReader

NetworkReader er en imaginær klasse ment for å beskrive hvordan de mobile enhetene skal registrere lokasjonsdata. De konkrete implementasjonene vil variere noe avhengig av den underliggende plattformen. NetworkReader har ansvaret for å lese inn informasjon fra omliggende trådløse nettverk. De fleste mobile enheter vil ha APIer for å gjennomføre slike nettverksskann. Dette innebærer å hente inn BSSID for aksesspunktene, samt deres RSSI verdier, innenfor et nettverk bestemt av en SSID. I denne implementasjonen er det tatt i bruk et nettverk med SSID lik eduroam. Det verdt å merke seg at aksesspunktene som danner fingeravtrykkene ikke trenger å tilhøre samme nettverk. Et lokasjonssystem vil kunne fungere på akkurat samme måte ved bruk av aksesspunkter fra forskjellige nettverk. Her er det allikevel lagt opp til at en skal bestemme hvilket nettverk som skal benyttes. Årsaken til dette er at det er de samme fysiske aksesspunktene som leverer flere virtuelle nettverk ved campus på NTNU. Aksesspunktene registrert ved et nettverksskann vil lagres i AccessPoint objekter som legges i en ArrayList. NetworkReader inneholder også en metode for å sortere denne ArrayListen, slik at aksesspunktet med den sterkeste RSSI verdien legges øverst.

Den resterende teksten i dette avsnittet vil gi en utdypende forklaring av de konkrete NetworkReader klient-implementasjonene. Det er kun beskrevet detaljer som er spesifikke for den enkelte implementasjonen.

iPhone OS

For iPhone OS er NetworkReaders funksjonalitet realisert via klassen WiFiReader. Det offisielle APIet for å aksessere det trådløse nettverkskortet på enheter som kjører iPhone OS, er ikke gjort tilgjengelig for utviklere. Det ble derfor benyttet et alternativt privat API. Dette leder til at det ikke vil være mulig å distribuere klientprogramvaren via AppleStore. Om det lanseres et offisielt API for å aksessere det trådløse nettverkskortet er programvaren

bygget opp slik at en da enkelt vil kunne ta i bruk dette APIet i stedet, uten at det krever store forandringer i koden. APIet som er brukt i denne implementasjonen kalles Apple80211Open. Dette er APIet er ikke ordentlig dokumentert, men har blitt gitt tilgang til via diverse utviklerforum på Internett[4]. Dokumentasjonen av APIet er gjerne kode eksempler som viser hvordan det kan benyttes. En utfordring ved å benytte et slikt API er at en ikke vil kunne oppgradere operativsystemet for den mobile enheten som skal kjøre lokasjonssystemet. Apple har ved oppgraderinger av operativsystemet gjort små forandringer slik at private rammeverk ikke lengre fungerer, og det tar gjerne litt tid før noen publiserer hvordan en kan få APIet til å fungere igjen.

Apple80211Open muliggjør registrering av omliggende aksesspunkt, hvilke nettverk disse tilhører, hvilken kanal aksesspunktet benytter, samt den registrerte RSSI verdien for hvert aksesspunkt. Metoden *scanNetworks* i WiFiReader benytter en metode fra Apple80211Open kalt *apple80211Scan* for å utløse et søk etter omliggende trådløse nettverk og aksesspunkt. Resultatet av dette nettverksskannet legges i en NSMutableDictionary kalt Networks. En NSDictionary er en type hashmap, at den er mutable betyr at en dynamisk ville kunne forandre størrelsen av hashmapet. Networks benytter BSSID som nøkkelverdi i hashmapet, en nøkkel gir en forbindelse til en ny NSDictionary instans som inneholder den resterende informasjonen om aksesspunktet. På grunn av denne relativt kompliserte datastrukturen rammeverket Apple80211Open legger opp til, inneholder WiFiReader også en metode kalt *getNetworksInfo*. *getNetworksInfo* sorterer ut lokasjonsdataene lokasjonssystemet behøver. Det lages AccessPoint objekter tilsvarende som beskrevet for serverprogramvaren for hvert aksesspunkt, og disse AccessPoint objektene legges deretter i et NSMutableArray som returneres i metoden.

BSSIDene blir for iPhone OS registrert på et annet format enn for resten av lokasjonssystemet (jf. avsnitt 3.4). Om det er nullverdier som første del av tallgrupperinger blir disse ikke inkludert BSSIDen. En BSSID som for resten av systemet blir registrert som 00:0b:85:88:44:0d, blir dermed lest inn som 0:b:85:88:44:d for iPhone OS. Det var derfor nødvendig å inkludere en metode i NetworkReader som sørget for at BSSIDene var på samme format som for resten av lokasjonssystemet. Denne funksjonaliteten er realisert via metoden *changeBssidFormat*.

[4]http://www.iphonedevsdk.com/forum/95141-post32.html og http://hkserv.ugent.be/boudewijn/blogs/?p=135, 20 februar 2010

Android

Android-plattformen gir utviklere tilgang til et API for å kunne benytte nettverkskortet i telefonene. Klientløsningen for Android-plattformen benytter to egen utviklede klasser for å realisere funksjonaliteten til NetworkReader. *FindLocation* kan ses på som hovedklassen i Android-løsningen. Den binder de forskjellige funksjonelle komponentene sammen og sørger for den totale funksjonaliteten av programvaren. *FindLocation* arver fra Activity klassen, og er den første aktiviteten som benyttes i det systemet starter opp. *FindLocation* inneholder en instans av en klasse kalt *WiFiManager*, dette er nødvendig for å muliggjøre nettverksskann. *WiFiManager* blir tildelt en Context systemtjeneste kalt *WIFI_SERVICE* (jf. avsnitt 2.5). Ved å kalle en metode på *WiFiManager* instansen vil en kunne gjennomføre et nettverksskann. Et utført nettverksskann vil utløse en Intent, denne Intenten må tas i mot ved hjelp av en BroadcastReceiver (jf. avsnitt 2.5). *FindLocation* inneholder derfor også en instans av en BroadcastReciever. BroadcastReceieveren blir instansiert med et IntentFilter kalt *SCAN_RESULTS_AVAILABLE_ACTION*. Dette gjør at BroadcastReceiveren kun vil kunne ta i mot meldinger fra nettverksskann. Klassen *WiFiScan* arver fra BroadcastReceiver, og ble laget for å gi BroadcastRecieveren den ønskede funksjonaliteten. I det *WiFiScan* mottar en Intent som beskriver at nettverksskannet er gjennomført vil klassen utløse metoden getAndDisplayLocationEstimate i *FindLocation*. I getAndDisplayLocationEstimate vil det for hvert registrerte aksesspunkt innen eduroam-nettverket lages et AccessPoint objekt, som beskrevet i den generelle forklaringen av NetworkReader.

I prosjektarbeidet gjennomført høsten 2009 ble tiden det tar å gjennomføre et skann for å identifisere omliggende trådløs infrastruktur, identifisert som den tidsmessige flaskehalsen av et lokasjonsestimat [32]. Under utviklingen av NetworkReader klientløsningene, beskrevet i dette avsnittet, ble det identifisert at et nettverksskann blir gjennomført vesentlig raskere for smarttelefonene, sammenliknet med resultatene for lokasjonsløsningen fra prosjektarbeidet. Dette tyder på at det vil være mulig å øke lokasjonssystemets ytelse med hensyn på tiden det tar å gjennomføre et lokasjonsestimat.

3.6.2 NetworkIO

NetworkIO er en imaginær klasse som beskriver hvordan klient programvaren skal kommunisere med lokasjonssystemets server. Det benyttes stream sockets for kommunikasjon mellom klientene og serveren. Årsaken til dette

er at en da er forsikret om at alle datapakker vil komme frem, og at innholdet av pakkene ikke er korrupt (Jf. avsnitt 2.3). NetworkIO inneholder metoder for å sette opp en socket forbindelse til serveren, sende og motta data, og avslutte socketen etter bruk. Figur 3.5 viser kommunikasjonsflyten mellom serversocketen og klientens socket. Figuren er beskrevet fra serverens perspektiv i avsnitt 3.5.3. Derfor vil kun delene av figuren som ikke tidligere har blitt beskrevet forklares her:

- **Socket()** Det opprettes et nytt socket endepunkt. Socketen blir satt til å være en klientsocket og å benytte seg av TCP protokollen.
- **getHostByName()** Her vil klientsocketen bli tildelt informasjon nødvendig for å identifisere serveren. I denne implementasjonen blir lokasjonsserveren identifisert ved en IP-adresse og et portnummer. Serveren vil være tilkoblet Internett via Ethernet, dette leder til at serveren alltid vil ha den samme IP-adressen. Serverens IP-adresse og portnummer lagres derfor statisk i klientprogramvarens NetworkIO klasse.
- **Close()** Etter et lokasjonsestimat vil klientsocketen lukkes. Det er normalt at en klientsocket kun benyttes til en datautveksling med serveren. Det opprettes dermed en ny socket forbindelse for hvert lokasjonsestimat.

Den resterende teksten i dette avsnittet vil gi en utdypende forklaring av de konkrete NetworkIO klient-implementasjonene. Det er kun beskrevet detaljer som er spesifikke for den enkelte implementasjonen.

iPhone OS

NetworkIO er i implementasjonen for iPhone OS realisert ved hjelp av klassen LXSocket. Når det opprettes en instans av LXSocket vil det instansieres en socket av typen *AF_INET address family*, og socketen blir satt til å være streambasert. Før socketen kan tas i bruk må det opprettes en forbindelse til server socketen. Dette er realisert gjennom LXSockets *connect* metode. *Connect* tar inn IP-adressen til serveren samt portnummeret serverapplikasjonens socket benytter, deretter settes parametere nødvendige for å kunne opprette en kommunikasjonslink. Selve oppkoblingen av kommunikasjonslinken blir tatt hånd om av underliggende funksjonalitet gjort tilgjengelig via C-biblioteker.

Data som skal sendes over socketforbindelsen må være på format som en bytestream. I lokasjonssystemet blir dataen som skal sendes over kommuni-

kasjonslinken behandlet som tekststrenger før og etter transmisjon. Metoden *sendString* tar inn tekststrengen som skal overføres til serveren, og gjør denne om til en bytestream. Deretter vil metoden *sendBytes* bli kalt. *SendBytes* tar inn bytestreamen som skal sendes samt lengden av denne, sender dataene over socketforbindelsen. Etter det har blitt sendt en forespørsel om lokasjonsestimering til serveren, vil klientapplikasjonen lytte etter svar. Svaret på lokasjonsestimatet blir tatt i mot ved hjelp av metoden *readString* i LXSocket. *readString* vil instansiere et buffer, og bytestreamen vil legges i dette bufferet. Deretter vil dataen bli lest ut fra bufferet tegn for tegn og enkodet til ASCII format. Underveis blir tegnene satt sammen til en leselig tekststreng, og tilslutt returnerer *readString* denne tekststrengen.

Android

Android løsningen for NetworkIO inneholder metoder for å sette opp en streamsocket forbindelse, samt sende og motta data over denne forbindelsen. Klassen benytter en instans av *Socket* fra Java.net APIet (jf. avsnitt 3.5.3, NetworkIO). Den kommende forklaringen av hvordan socketkommunikasjon er implementert er også gjeldene for programvaren på serversiden av systemet. Da begge implementasjonene er gjennomført ved hjelp av Java, vil også løsningene være tilnærmet identiske. Konstruktøren til *Socket* tar inn en IP-adresse og et portnummer. I det en *Socket* instansieres skapes det en kommunikasjonslink til *ServerSocketen* IP-adressen og portnummeret tilhører. Dette er realisert i NetworkIO metoden *sendToServer*. *sendToServer* vil også sende lokasjonsdataene nødvendig for at serveren skal kunne utføre et lokasjonsestimat. Da socketkommunikasjon benytter bytestreams, tas det i bruk flere klasser fra APIet Java.IO[5] for å muliggjøre overføring av objekter over socketforbindelsen:

- *PrintWriter* omgjør java objekter til en tekstbasert outputstream.
- *BufferedWriter* omgjør den tekstbaserte outputstreamen til en tegnbasert outputstream, og implementerer et buffer for streamen.
- *OutputStreamWriter* utgjør en overgang fra tegnbasert stream til bytebasert outputstream.

Tilslutt sendes den bytebaserte outputstreamen over socketen ved å kalle metoden *getOutputStream* på socket objektet.

[5]http://java.sun.com/j2se/1.4.2/docs/api/java/io/package-summary.html, 20 april 2010

Metoden *listenToConnection* benyttes for å ta i mot svaret av et lokasjonsestimat fra serveren. Ved å kalle metoden *getInputStream* på socket objektet vil en motta svaret på lokasjonsestimatet som en bytebasert inputstream. Da det er ønskelig at dataene mottatt fra serversiden av systemet er på et tekstbasert format, benyttes det igjen to klasser fra Java.IO APIet:

- *InputStreamReader* omgjør den bytebaserte inputstreamen til en tegnbasert stream.
- *BufferedReader* omgjør den tegnbaserte inputstreamen til en tekstbasert stream. En BufferedReader vil også implementere et buffer for streamen slik at en kan lese inn lengre tekststrenger av gangen.

Resultatet av lokasjonsestimatet mottatt fra lokasjonsserveren lagres i en tekststreng som returneres i metoden *listenToConnection*. For å avslutte en socketforbindelse kan en kalle metoden *closeConnection*.

3.6.3 Klient brukergrensesnitt

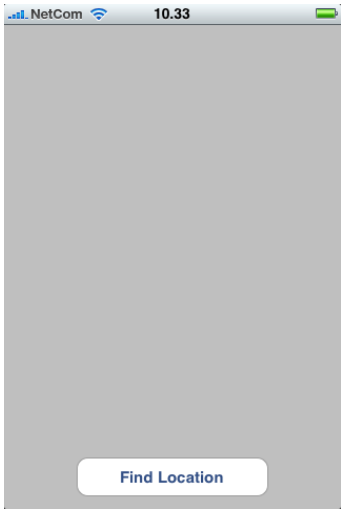
Da dette lokasjonssystemet er beregnet for innendørs bruk, vil det ikke være mulig å benytte seg av eksisterende karttjenester som for eksempel Google Maps for å vise brukerens lokasjon. Det er derfor laget enkle plattform spesifikke brukergrensesnitt for hver av de to klientløsningene. Brukergrensesnittene har som oppgave å grafisk vise brukerens lokasjon slik at en enkelt kan kontrollere hvorvidt riktig lokasjon har blitt estimert, samt demonstrere lokasjonssystemets virkemåte. Brukergrensesnittet har derfor ikke blitt utstyrt med mer enn helt grunnleggende funksjonalitet. Lokasjonssystemets dekningsområde er fordelt over to etasjer. Brukergrensesnittet benytter et kart for hver etasje, og disse kartene er lagret i programvaren som png filer. Etter et lokasjonsestimat har blitt gjennomført vil brukergrensesnittet beregne hvilken piksel i kartet som tilsvarer den estimerte lokasjonen. Smarttelefoner som iPhone og Android-telefoner har begrenset skjermstørrelse. Dette setter en begrensning på hvor store bilder en vil kunne vise på skjermen uten at det går utover kartets oppløsning og dermed brukervennlighet. Derfor blir kartet klippet til slik at ikke hele etasjen vises for hvert lokasjonsestimat. Brukeren lokasjons vises som en rød prikk på kartet, og kartet klippes til slik at brukerens lokasjon blir vist midt i skjermen.

iPhone OS

Utviklingsomgivelsen XCode, som brukes for å lage iPhone-applikasjoner, har et integrert program for å designe brukergrensesnitt kalt Interface Builder. Denne programvaren har blitt benyttet for å utvikle brukergrensesnittet for iPhone OS. Interface Builder tilbyr en grafisk editor for utforming av brukergrensesnitt, og en rekke ferdig utviklede elementer som knapper, rullgardinmenyer og tekstfelt. De forskjellige elementene settes til å tilhøre et UIWindow. Et UIWindow kan ses på som et lerret en kan tildele brukergrensesnitt elementer. En iPhone-applikasjon har vanligvis kun et UIWindow, mens elementene som vises i vinduet vil kunne skiftes ut dynamisk. Elementene i brukergrensesnittet lagres som ressurser i XCode IDE slik at de kan benyttes fra Objective-C koden.

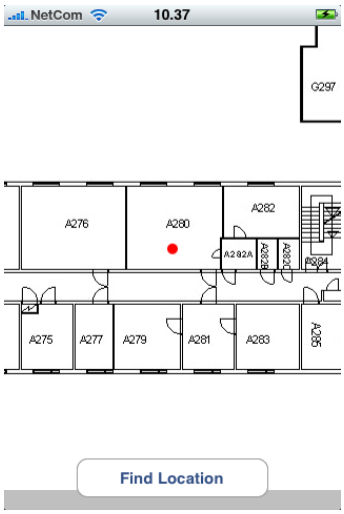
Brukergrensesnittet for iPhone OS består av to elementer: Et UIImageView brukt for å vise lokasjonen til brukeren, og en knapp som vil utløse et lokasjonsestimat. Disse elementene har blitt satt til å tilhøre et UIWindow i Interface Builder. I Objective-C koden er det laget instanser som tilsvarer brukergrensesnitt elementene, og det har blitt skapt en forbindelse mellom brukergrensesnitt elementene og kodeinstansene ved hjelp av Interface Builder. Brukergrensesnitt er lagret i XCode IDE som en .xib fil, for lokasjonssystemet er denne kalt LocationViewController.xib. XCode IDE prosjektet inneholder en klasse kalt *LocationViewController* som samhandler med LocationViewController.xib filen. *LocationViewController* klassen benytter seg av funksjonalitetene til andre klasser som *WiFiReader* og *LXSocket*, og sørger for den totale funksjonaliteten av klient programvaren. Ved oppstart av programvaren vil *LocationViewController* opprette en instans av *WiFiReader*, og brukergrensesnittet vil vises som i figur 3.6.

Et trykk på knappen "Find Location" i brukergrensesnittet vil opprette en instans av *LXSocket* samt starte opp metoden locationLookUp i *LocationViewController*. LocationLookUp tar i bruk *WifiReader* (jf. avsnitt 3.6.1) instansen for å lese inn informasjon fra omliggende trådløst nettverk. Deretter sendes lokasjonsdata nødvendig for et lokasjonsestimat til serveren ved hjelp av *LXSocket* instansen. *LXSocket* instansen tar også i mot resultatet av lokasjonsestimatet. For å grafisk vise den estimerte lokasjonen i brukergrensesnittet benyttes metoden displayImage i *LocationViewController*. displayImage tar inn de tre koordinatene funnet ved lokasjonsestimatet. Z-koordinaten vil avgjøre hvilket kart som skal vises i brukergrensesnittet. Metoden inneholder informasjon om hvilke piksler som tilsvarer koordinatsystemets origo for hvert av de to kartene, samt hvor mange piksler som tilsvarer en meter.



Figur 3.6: iPhoneGUI ved oppstart.

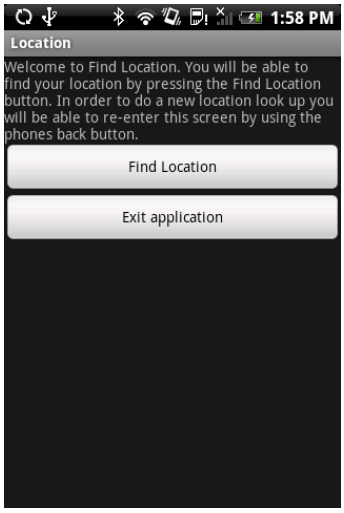
Ut i fra denne informasjonen beregnes pikslene i kartet som tilsvarer brukerens lokasjons, og kartet blir klippet til slik at denne lokasjonen vises midt i kartutsnittet. Brukerens lokasjon blir deretter markert som en rød prikk, og kartutsnittet blir satt til å vises i brukergrensesnittets UIImageView. Brukergrensesnittet etter et gjennomført lokasjonsestimat er vist i figur 3.7. Her er brukerens lokasjon estimert til rom A280. Dette er i andre etasje i A-blokken i Elektro-bygget ved Gløshaugen NTNU. Et nytt lokasjonsestimat vil gjennomføres om knappen "Find Location" trykkes igjen.



Figur 3.7: iPhoneGUI etter lokasjonsestimat.

Android

For Android brukes en komponenttype kalt Activity (jf. avsnitt 2.5) for å presentere et brukergrensesnitt. Selve rammeverket for brukergrensesnittet defineres ved XML. Android stiller til rådighet et enkelt XML vokabular som tilsvarer standard UI komponenter som knapper, tekstfelt og menyer. Det skrives gjerne et XML dokument for hver aktivitet, og hver aktivitet som skal benyttes i programvaren inkluderes i en XML manifestfil som definerer strukturen av hele programvaren. Det også mulig via Javakode å dynamisk legge til, fjerne eller forandre brukergrensesnittet mens programmet kjører[6].



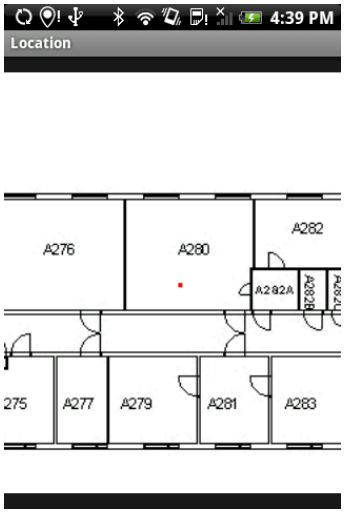
Figur 3.8: AndroidGUI før lokasjonsestimat.

Klientapplikasjonen for Android benytter to aktiviteter for å skape brukergrensesnittet for lokasjonssystemet. FindLocation.xml definerer brukergrensesnittet slik det ser ut i det applikasjonen starter opp, dette er vist i figur 3.8. Dette er et statisk brukergrensesnitt bestående av et tekstfelt og to knapper. Tekstfeltet beskriver hvordan applikasjonen fungerer. Knappen "Find Location" benyttes for å utløse et lokasjonsestimat, mens "Exit application" vil avslutte applikasjonen. Tilhørende hver aktivitet er det en tilsvarende Java klasse. Klassen *FindLocation* responderer på knappetrykk fra brukergrensesnittet definert i FindLocation.xml. *FindLocation*, som beskrevet i avsnitt 3.6.1, utgjør en hovedklasse i Android-løsningen og binder funksjonalitet fra resten av systemet sammen. Når "Find Location" knappen blir trykket vil *FindLocation* sørge for at nettverksinformasjon blir lest inn

[6]http://developer.android.com/guide/topics/ui/declaring-layout.html, 15 april 2010

fra omliggende trådløse nettverk, at informasjonen blir videreformidlet til serveren, samt at lokasjonsestimatet fra serveren blir tatt i mot. Deretter vil *FindLocation* starte opp en ny aktivitet kalt *MapDisplay*. Dette gjøre ved at det sendes en Intent (jf. avsnitt 2.5), som utløser aktiviteten. I en Intent kan det legges ved verdier som for eksempel tall eller tekststrenger [19]. I Intenten som starter opp *MapDisplay* legges koordinatene som ble funnet ved lokasjonsestimatet med.

MapDisplay består av et ImageView som benyttes til å grafisk vise brukerens lokasjon. Det er ønskelig å kunne gjøre justeringer på kartet som vises i brukergrensesnittet til systemet. For å gjøre dette mulig må klassen som justerer kartet arve fra klassen View. Java lar en klasse kun arve fra én annen klasse. Derfor vil det ikke være mulig å gjøre slike justeringer direkte i *MapDisplay*, da en aktivitet allerede arver fra klassen Activity (jf. avsnitt 2.5). For å løse dette problemet ble klassen *Zoom* laget, *Zoom* arver fra View. *MapDisplay* oppretter en instans av *Zoom* og tildeler denne instansen samme Context som seg selv. Deretter blir *Zoom*-instansen satt til å vises i *MapDisplays* ImageView. *Zoom* inneholder metoder for å finne pikslene i kartet tilsvarende brukerens lokasjon og å klippe til kartet så det passer en mobiltelefonskjerm. Denne funksjonaliteten er tilsvarende som beskrevet under avsnittet omhandlene brukergrensesnittet for iPhone OS. Brukergrensesnittet for Android-løsningen etter et lokasjonsestimat er vist i figur 3.9. For å gjennomføre et nytt lokasjonsestimat kan en benytte telefonens tilbakeknapp. Dette vil lede brukergrensesnittet tilbake til oppstartsvinduet, som vist i figur 3.8, og systemet vil være klart for et nytt lokasjonsestimat.



Figur 3.9: AndroidGUI etter lokasjonsestimat.

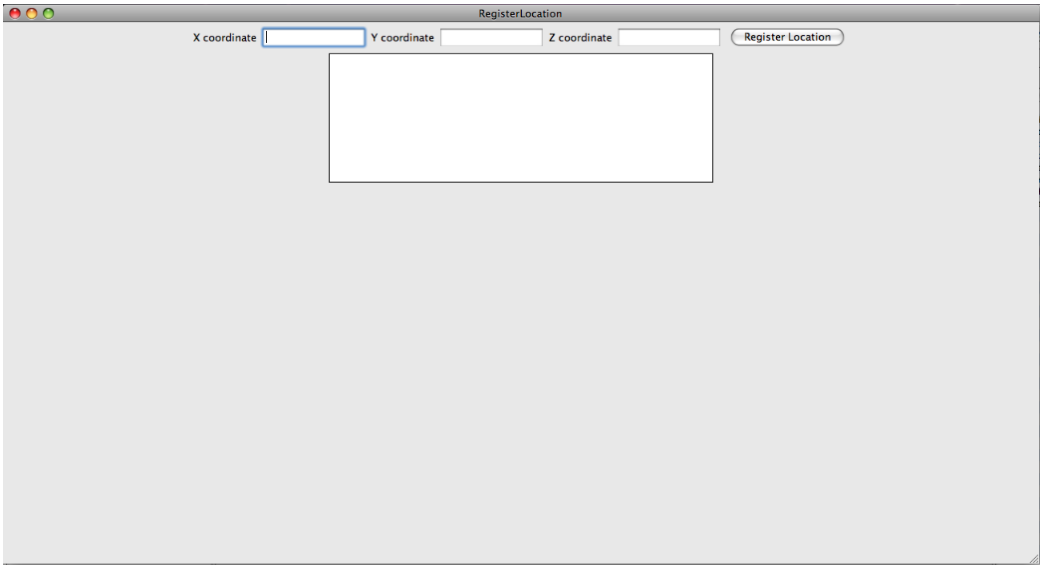
Zoom muliggjør også at en kan zoome inn og ut rundt lokasjonen som er estimert for brukeren. Dette gjøres ved hjelp av telefonens museball. Opprinnelig var det tenkt at hele kartet over dekningsområdet skulle vises i etterkant av et lokasjonsestimat, for så å la brukeren zoome inn om det var ønskelig. Dette lot seg ikke gjøre på grunn av minnet tilgjengelig på Android-telefonen. Kartfilene er svært detaljerte og det kreves god oppløsning om kartene skal fremstå som leselige. Samtidig vil en zoom-funksjon kreve at kartfilen blir klippet til for å vise den utvalgte delen av kartet. Tidligere utklipp av kartfilen må mellomlagres for å tillate at kartet zoomes ut igjen. Telefonen hadde ikke tilstrekkelig minne for en slik funksjonalitet. Brukergrensesnittet for Android-plattformen ble utviklet før brukergrensesnittet for iPhone OS, og da denne utfordringen ble identifisert ble det ikke forsøkt å lage noen zoom funksjonalitet for iPhone.

3.7 Verktøy for innsamling av lokasjonsdata

For å effektivisere og forenkle innsamlingen av lokasjonsdata har det blitt utviklet en programvare. Denne programvaren er implementert ved hjelp av Java. Innsamlingen av lokasjonsdata har blitt gjennomført ved å benytte programvaren på en datamaskinen som beskrevet i avsnitt 2.9. Programvaren har som oppgave å registrere informasjon fra omliggende trådløse nettverk, og ut i fra denne informasjonen lagre en lokasjon med tilhørende lokasjonsfingeravtrykk i lokasjonsdatabasen. Programvaren inneholder to klasser sentrale for funksjonaliteten til verktøyet. *NetworkReader* inneholder funksjonalitet for å samle inn lokasjonsdata fra omliggende trådløse nettverk, mens *DatabaseReader* håndterer registrering av lokasjoner med tilhørende lokasjonsfingeravtrykk i databasen. De resterende klassene i programvaren inneholder minimalt med funksjonalitet og vil derfor ikke bli beskrevet her. Brukergrensesnittet for verktøyet er vist i figur 3.10. Tekstfeltet vil gi beskjed om lokasjonsregistreringen var vellykket. Hvis dette ikke er tilfelle vil det bli gitt en årsak til at lokasjonen ikke kunne registreres. Dette kan for eksempel være at en koordinat ikke har blitt tastet inn, eller at det ikke ble registrert et tilstrekkelig antall omliggende aksesspunkt.

3.7.1 NetworkReader

NetworkReader inneholder den samme funksjonaliteten som NetworkReader-klassen beskrevet for klientløsningene i avsnitt 3.6.1. Selve implementasjonen



Figur 3.10: Verktøy for innsamling av lokasjonsdata.

er allikevel ganske annerledes da Java SE, for utvikling på datamaskiner, ikke inneholder noe API for å aksessere nettverkskortet. Årsaken til dette er at denne Java plattformen skal kunne benyttes på alle datamaskiner uavhengig av underliggende hardware, og det er derfor ikke APIer for å aksessere hardware som vil kunne variere fra maskin til maskin.

For å samle inn lokasjonsdata benyttes det derfor et systemkall. Ved å kalle ”airport -s” i en Macintosh-maskins terminalvindu, vil maskinen gjennomføre en lytting etter trådløse aksesspunkt. Airport refererer til maskinens trådløse nettverkskort mens s står for skann. Tilsvarende kommandoer finnes også for maskiner som kjører Windows og Linux operativsystemer. Resultatet av nettverksskannet blir levert som en bytestream. Bytestreamen tas derfor imot i en InputStreamReader som vil kode bytestreamen om til en tegnbasert datastrøm. For å kunne lese resultatet av nettverkskannet som tekstlinjer, puttes den tegnbaserte datastrømmen i en BufferedReader. En tekstlinje tilsvarer all informasjon tilgjengelig for et aksesspunkt. For å separere ut informasjonen fra nettverksskannet benyttes Regexp (Regular Expressions). Først blir alle aksesspunkt som ikke tilhører nettverket med SSID lik eduroam forkastet, deretter kontrolleres det hvilken kanal aksesspunktet tar i bruk. Da det er ønskelig at lokasjonssystemet skal kunne benyttes av mobile enheter med 802.11b/g nettverkskort, må lokasjonsdatabasen kun inneholde aksesspunkt som anvender 2,4 GHz frekvensbåndet. I den trådløse infrastrukturen ved NTNU Gløshaugen brukes det tre kanaler i 2,4 GHz frekvensområdet, ka-

nal 1, 6 og 11. Alle aksesspunkt som benytter kanaler over 11 vil derfor bli forkastet. Det lages deretter AccessPoint objekter (beskrevet i avsnitt 3.5.3) for de resterende aksesspunktene, disse lagres i en ArrayList. AccessPoint objektene i ArrayListen vil bli sortert ut i fra RSSI verdi.

3.7.2 DatabaseReader

DatabaseReader har ansvaret for å registrere nye lokasjoner i lokasjonsdatabasen. I tillegg til å inneholde metoder for å koble til og fra databasen, inneholder klassen metoden *regLoc* som håndterer den konkrete registreringen av lokasjonsdata og lokasjoner.

regLoc

Metoden tar inn tre heltallsverdier. Dette er koordinatene som har blitt tastet inn i brukergrensesnittet for lokasjonen som skal registreres. Det er brukeren av verktøyet sitt ansvar at riktige koordinater blir registret for lokasjonen. *RegLoc* vil først opprette en ny entitet i databasens Location tabell (jf. figur 3.4), og ta vare på IDen for lokasjonsentiteten. Hver lokasjon skal identifiseres ved hjelp av tre fingeravtrykk, og hvert fingeravtrykk linkes til et spesifikt aksesspunkt. Metoden kontrollerer derfor om aksesspunktene som skal være en del av fingeravtrykkene allerede er registret i databasen. Om dette er tilfelle lagres IDene for disse aksesspunktene. Det opprettes nye entiteter i databasens AccessPoint tabell for aksesspunkt som ikke er registrert, og IDen for disse lagres. Tilslutt opprettes tre entiteter i FingerPrint2 tabellen. Hver fingeravtrykk entiteten linkes til lokasjonsIDen, samt aksesspunktIDen til aksesspunktet som leverte RSSI verdien i fingeravtrykket.

Kapittel 4

Testing

Dette kapittelet vil beskrive hvordan systemets ytelse har blitt testet, samt presentere resultatene av disse testene. To typer testing har blitt benyttet for å kontrollere systemets funksjonalitet, modultesting og svartboks-testing. Avsnitt 4.1 vil beskrive hvordan lokasjonssystemet har blitt modultestet, mens testene i avsnitt 4.2 og 4.3 har blitt gjennomført som svartboks-testing.

4.1 Modultesting

Modultesting har blitt benyttet for å kontrollere adskillbare komponenters funksjonalitet under implementasjonen. På denne måten har en forsikret seg om riktig adferd fra disse komponentene, før de har blitt integrert med resten av systemet. Modultestingen har foregått underveis i implementasjonen. Om mulig har en komponent blitt adskilt fra resten av systemet mens den har blitt utviklet. Når komponenten er ferdig vil funksjonaliteten kontrolleres ved at komponenten blir betraktet som en svart boks. Komponenten tar inn en input og det kontrolleres at riktig output returneres avhengig av input verdien. Komponenter er her et vidt begrep og kan omfatte både klasser og metoder. Modultesting har blitt gjennomført på komplekse metoder i systemet. Dette innbærer metoder grunnleggende for systemets funksjonalitet. Eksempler er metodene beskrevet i avsnitt 3.6.1 og i avsnitt 3.5.3 under DatabaseReader og Calculator. Modultesting har også blitt benyttet for komplekse MySQL databasespørringer. Dette har blitt gjort ved å benytte en egen metode for å kontrollere at spørringen returnerer de ønskede verdiene. På denne måten har logikken rundt spørringen blitt fjernet, og det har vært enklere å kontrollere resultatet samt hastighet av spørringene. Et eksempel på modultesting av

hele klasser er NetworkIO klassene beskrevet i avsnitt 3.5.3 og 3.6.2. For at programvaren skal kunne sende noe over nettverket må all funksjonalitet i disse klassene være på plass. NetworkIO klassene ble testet ved at klientene sendte en enkel melding til serveren, og serveren tok imot og svarte på denne meldingen. Tilslutt tok klientene i mot svarmeldingen sendt fra serveren.

Etter at en komponent har blitt modultestet har den blitt integrert med resten av systemet, som allerede er blitt modultestet. Deretter har det blitt kontrollert at komponenten fungerer som ønsket knyttet sammen med resten av systemet.

4.2 Tiden av et lokasjonsestimat

Dette avsnittet vil identifisere ytelsen av lokasjonssystemet med hensyn på tiden det tar å gjennomføre et lokasjonsestimat. For å kunne identifisere den tidsmessige flaskehalsen av systemets ytelse, har testen blitt delt opp ut i fra sentrale deler av et lokasjonsestimat.

4.2.1 Serveren

Testene beskrevet i dette avsnittet har blitt gjennomført med én mobil enhet av gangen. Resultatene av denne testen må dermed ses som hvordan serveren yter uten en kø av lokasjonsforespørsler. Tiden det tar å gjennomføre et lokasjonsestimat for serveren blir beregnet fra serveren mottar en forespørsel om et lokasjonsestimat, og til svaret på lokasjonsestimatet er sendt tilbake til den forespørrende klienten. Tiden serveren trenger for å gjennomføre et lokasjonsestimat vil teoretisk sett være uavhengig av hva slags mobil enhet som benyttes. Forespørslene og svarene på lokasjonsestimatene vil være på samme format uavhengig av den mobile enheten (jf. avsnitt 3.4), og selve lokasjonsestimatet vil gjennomføres nøyaktig likt.

En faktor som vil påvirke tiden det tar for serveren å gjennomføre et lokasjonsestimat, er lokasjonsdataene som blir mottatt over socketforbindelsen fra den mobile enheten. Om disse lokasjonsdataene leder til at det må beregnes Euklids avstand/vektingsalgoritme for mange lokasjoner i lokasjonsdatabasen, vil tiden serveren trenger for å estimere en lokasjon øke. For å identifisere om det vil være ulikheter i tiden serveren gjennomsnittlig trenger for å estimere en lokasjon for de ulike mobile enhetene, har serverens ytelse

med hensyn på tid blitt testet ved bruk av både iPhone og Android telefonen. For å identifisere om det finnes noen tidsmessige forskjeller for de to lokasjonsbestemmende algoritmene, har serverens ytelse også blitt testet ved bruk av begge disse algoritmene. Det ble gjennomført 50 lokasjonsestimat for hver algoritme, med hver mobile enhet. Tabell 4.1 viser resultatet av testen når iPhonen ble benyttet, mens tabell 4.2 viser resultatet av testen når Android-telefonen ble benyttet.

	Euklids avstand	*Vektingsalgoritme*
Tid i millisekunder	61.62	95.26

Tabell 4.1: Gjennomsnittlig tidsbruk for server ved bruk av iPhone.

	Euklids avstand	*Vektingsalgoritme*
Tid i millisekunder	106.48	119.62

Tabell 4.2: Gjennomsnittlig tidsbruk for server ved bruk av Android.

Ut i fra testen ble det identifisert at tiden serveren trenger for å gjennomføre et lokasjonsestimat kan deles inn i to grupperinger. Disse grupperingene er gjeldene for begge de lokasjonsbestemmende algoritmene og oppstår uavhengig av hvilken mobile enhet som benyttes:

- **Gruppering 1:** En gruppering hvor tiden for å gjennomføre et lokasjonsestimat er relativt kort. Ut i fra testen identifiseres det at et lokasjonsestimat som blir gjennomført i løpet av 40 til 120 millisekunder vil tilhøre denne grupperingen. De fleste lokasjonsestimat tilhørende gruppering 1 vil bli gjennomført i løpet av 40 til 90 millisekunder.
- **Gruppering 2:** En gruppering hvor tiden det tar å gjennomføre et lokasjonsestimat er lang sammenliknet med gruppering 1. Ut i fra testen identifiseres det at et lokasjonsestimat som tar lengre enn 150 millisekunder å gjennomføre, vil tilhøre denne grupperingen. De fleste lokasjonsestimat tilhørende gruppering 2 vil bli gjennomført i løpet av 180 til 250 millisekunder.

Denne oppdelingen av resultatene oppstår som en følge av den sonebaserte lokasjonsestimeringen systemet tar i bruk. Som beskrevet i avsnitt 3.5.2 vil systemet først kontrollere om det finnes lokasjoner i databasen som er knyttet til alle de tre aksesspunktene den mobile enheten registrerte med sterkest signalstyrker. Hvis ikke hentes det ut lokasjoner som er knyttet til to av aksesspunktene den mobile enheten registrert med sterkest signalstyrker. Dette gjennomføres før Euklids avstand eller vektingsalgoritmen benyttes.

I de tilfellene hvor systemet kun finner lokasjoner som er knyttet til to av aksesspunktene den mobile enheten registrerte med sterkest signalstyrker, vil antallet lokasjoner som må behandles av en lokasjonsbestemmende algoritme bli betraktelig større enn for tilfeller der det er funnet lokasjoner linket til alle tre aksesspunktene. Ved mange lokasjoner vil de lokasjonsbestemmende algoritmene bli tregere da det må gjøres beregninger for, og itereres gjennom, flere Location objekter (jf. avsnitt 3.5.3).

	Gruppering 1	*Gruppering 2*
iPhone, Euklids avstand	47	3
iPhone, Vektingsalgoritme	38	12
Android, Euklids avstand	36	14
Android, Vektingsalgoritme	39	11

Tabell 4.3: Gruppering av serverens tidsbruk.

Tabell 4.3 viser fordelingen av lokasjonsestimat tilhørende de to grupperingene for de ulike algoritmene og mobile enhetene. Ut i fra disse resultatene ser vi at for Android-telefonen blir lokasjonen oftere estimert ut i fra kun to aksesspunkt enn for iPhonen. Dette tilsier at iPhonen oppfatter omliggende trådløs infrastruktur mer likt slik referansemaskinen brukt for å samle inn lokasjonsdata gjør, sammenliknet med Android-telefonen. Dette stemmer overens med resultatene identifisert i signalstyrketesten beskrevet avsnitt 3.2. Årsaken til at de ulike mobile enhetene oppfatter omliggende trådløs infrastruktur forskjellig, er høyst sannsynlig et resultat av at de er utstyrt med ulike antenner og nettverkskort. Det faktum at en lokasjon oftere blir estimert ved hjelp av kun to aksesspunkt for Android-telefonen er en indikator på at lokasjonssystemets ytelse, med hensyn på nøyaktighet av et lokasjonsestimat, vil være dårligere for Android-telefonen enn for iPhonen.

Sannsynligheten for å ende opp med et lokasjonsestimat tilhørende gruppering 1 er uavhengig av hvilken lokasjonsbestemmende algoritme som benyttes. Dette vil avhenge av lokasjonsdataene registrert for den mobile enheten. Den gjennomsnittlige tiden serveren trenger for å gjennomføre et lokasjonsestimat vil allikevel være noe lengre for vektingsalgoritmen. Årsaken til dette er at denne algoritmen inneholder en ekstra iterering gjennom Location objektene.

4.2.2 Klientene

Testene beskrevet i dette avsnittet har blitt gjennomført med en klient av gangen slik at det ikke oppstår noen kø på serversiden av systemet. For

klientene har det blitt identifisert tre sentrale faser av en lokasjonsestimering:

1. Tiden den mobile enheten benytter på å lese inn nødvendig informasjon fra omliggende trådløse nettverk. Denne delen av lokasjonsestimatet inneholder også sortering av aksesspunktene basert på signalstyrkeverdi. I prosjektarbeidet ble det identifiserte at sorteringen av aksesspunktene utgjør en ubetydelig del av den totale tiden av et lokasjonsestimat [32]. Dette er derfor inkludert i denne fasen.

2. Tiden det tar fra lokasjonsdata blir sendt til serveren, og til klienten mottar resultatet av et lokasjonsestimat. Denne fasen inneholder tiden serveren behøver for å gjennomføre et lokasjonsestimat, identifisert i avsnitt 4.2.1.

3. Tiden klientløsningen trenger for å grafisk vise den estimerte lokasjonen i brukergrensesnittet. I avsnitt 3.6.3 ble det identifisert at det krevde mye ressurser av Android-telefonen for å håndtere kartfilene bruk i systemet. Dette kan være en indikator på at denne fasen av et lokasjonsestimat vil kunne være tidkrevende. Det er derfor essensielt å identifisere hvor mye tid som går med til å generere systemets brukergrensesnitt.

For å identifisere hvor mye tid som går med til de forskjellige fasene av en lokasjonsestimering ble det derfor gjennomført 50 lokasjonsestimeringer for hver av de to smarttelefonene. I prosjektarbeidet gjennomført høsten 2009 ble det identifiserte at bygningsmessige omgivelser ikke vil påvirke hvor lang tid et lokasjonssystemet trenger for å gjennomføre et lokasjonsestimat [32]. De mobile enhetene har derfor befunnet seg på samme sted under hele denne testen. Serveren har benyttet Euklids avstand som lokasjonsbestemmende algoritme.

Android

Resultatene av tidsbruk testen for Android-telefonen er vist i tabell 4.4. Tallene oppgitt er gjennomsnittsverdien for hver fase ut i fra de 50 lokasjonsestimatene gjort for Android-telefonen.

	Fase 1	*Fase 2*	*Fase 3*	*Total*
Tid i millisekunder	730.36	373.86	864.54	1968.76

Tabell 4.4: Gjennomsnittlig tidsbruk for Android.

- For fase 1 ble det registrert svært like målinger for alle de 50 lokasjonsestimatene. Standardavviket for denne fasen var 34.6 millisekunder.

Gjennomsnittsverdien er dermed representativ for hvor lang tid denne fasen av et lokasjonsestimat krever.

- Tiden som kreves for å gjennomføre fase 2 vil kunne uttrykkes som: *tiden serveren trenger for å estimere en lokasjon + tiden det tar å sende data til og fra serveren.* Tiden det tar å sende data til og fra serveren vil i liten grad påvirkes av lokasjonssystemet, da det er samme antall bytes som sendes over nettverket for hvert lokasjonsestimat (jf. avsnitt 3.4). Som identifisert i avsnitt 4.2.1 vil tiden serveren trenger for å gjennomføre et lokasjonsestimat kunne fordeles i to grupperinger. Hvilken gruppering lokasjonsestimatet tilhører vil dermed være avgjørende for hvor lang tid som behøves for å gjennomføre fase 2.

  Spredningen av målingene for fase 2 var fra 221 til 994 millisekunder. Lokasjonsestimatet som trengte 994 millisekunder for fase 2 kan ses som et spesialtilfelle. Det ble registrert at for dette lokasjonsestimatet ble mobiltelefonens forbindelse til det trådløse nettverket fornyet. Dette er funksjonalitet som er ferdig innebygget i telefonenes operativsystem og vil bli prioritert fremfor annen aktivitet på telefonen. Dette ledet dermed til at fase 2 for dette lokasjonsestimatet tok lengre tid en normalt. Den nest lengste tiden som ble registrert for fase 2 var på 616 millisekunder. Ved å studere gjennomsnittstiden for fase 2 sammenliknet med tiden serveren trenger for å gjennomføre et lokasjonsestimat, identifisert i avsnitt 4.2.1, kan vi estimere den gjennomsnittlige tiden som kreves for å sende data over nettverket til å være $373.86 - 106.48 = 267.38$ millisekunder. Dette estimatet går ut ifra den gjennomsnittlige tiden serveren trengte ved bruk av Euklids avstand som lokasjonsbestemmende algoritme.

- For fase 3 ble det registrert svært like målinger. Dette står i sammenheng med oppgavene som utføres i fasen, som vil være identiske for hvert lokasjonsestimat (se avsnitt 3.6.3). For samtlige målinger av fase 3 ble det identifisert en tidsbruk mellom 800-900 millisekunder, med to unntak. For disse to unntakene ble tiden bruk i fase 3 registrert til henholdsvis 1212 og 1422 millisekunder. Årsaken til disse unntakstilfellene er at mobiltelefonen fornyet forbindelsen til det trådløse nettverket tilsvarende som beskrevet for fase 2.

iPhone

Resultatene av tidsbruk testen for iPhone er vist i tabell 4.5. Tallene oppgitt er gjennomsnittsverdien for hver fase ut i fra de 50 lokasjonsestimatene gjort for iPhonen.

	Fase 1	*Fase 2*	*Fase 3*	*Total*
Tid i millisekunder	1120.96	221.62	791.22	2133.80

Tabell 4.5: Gjennomsnittlig tidsbruk for iPhone.

- Målingene viste at tiden det tar å gjennomføre fase 1 av et lokasjonsoppslag for iPhone er svært stabil. Standardavviket for denne fasen var 48.22 millisekunder. Ut i fra gjennomsnittsverdien for fase 1 ser en at iPhonen trenger lengre tid for å identifisere omliggende trådløse nettverk enn Android-telefonen. Da Apple ikke har lansert noe offisielt rammeverk for å aksessere det trådløse nettverkskortet på iPhone (jf. avsnitt 3.6.1), benyttes det et private rammeverk for å muliggjøre denne funksjonaliteten. Dette kan være årsaken til at iPhonen krever lengre tid for å gjennomføre fase 1. Da tilgangen til det trådløse nettverkskortet opprinnelig er avstengt, må det private rammeverket inneholde en måte å jobbe seg rundt denne sperringen. Det vil være naturlig at en slik prosess vil kunne øke tidsbruken for fase 1.
- For iPhone var spredningen av målingene for fase 2 fra 86 til 778 millisekunder. I avsnittet omhandlende Android-telefonen ble det identifisert at tiden som kreves for å gjennomføre fase 2 vil kunne uttrykkes som: *Tiden serveren trenger for å estimere en lokasjon + tiden det tar å sende data til og fra serveren*, der tiden serveren behøver for å estimere en lokasjon vil være lokasjonssystemets tidsmessige påvirkning av fase 2. For iPhone estimeres dermed den gjennomsnittlige tiden som kreves for å sende data over nettverket til å være $221.62 - 61.62 = 160$ millisekunder. Ved å sammenlikne denne gjennomsnittsverdien med spredningen av målingene, ser en at det vil variere hvor lang tid det tar å overføre data mellom klient og server. Den korteste registrerte tiden for fase 2 var 86 millisekunder.
- For fase 3 ble det tilsvarende som for Android telefonen registrert svært like målinger, med en spredning fra 729 til 986 millisekunder. Målingene var jevnt fordelt innenfor dette intervallet. Gjennomsnittsverdien for tidsbruken i fase 3 er også relativt lik for de to smart telefonene.

4.2.3 Evaluering av systemets tidsbruk

Hensikten med en omfattende test av systemets ytelse, med hensyn på tiden det tar å gjennomføre et lokasjonsestimat, er å identifisere hvilke deler av systemets funksjonalitet en vil kunne forbedre om det ønskes et raskere system. For prosjektet gjennomført høsten 2009 ble den tidsmessige flaskehalsen av lokasjonssystemet identifisert til å være innlesning av informasjon fra omliggende trådløse nettverk. Gjennomsnittsverdien for denne fasen ble da funnet til å være 2863 millisekunder [32]. Tidsbruken for denne fasen av et lokasjonsestimat er nå er blitt redusert med over 60 % for begge de mobile enhetene. Det antas dermed at det er lite rom for tidsmessig forbedring av denne fasen av et lokasjonsestimat. For fase 2 er den gjennomsnittlige tidsbruken henholdsvis 373.86 og 221.62 millisekunder for de to mobile enhetene. Denne tiden inneholder selve beregningen av lokasjonsestimatet samt dataoverføring mellom de mobile enhetene og serveren. Samtidig utgjør fasen under 20% av den gjennomsnittlige totale tiden for et lokasjonsestimat, for både iPhone og Android-telefonen. Dermed vil et forsøk på å gjøre denne fasen av et lokasjonsestimat mer tidseffektiv bare kunne gi relativt små forbedringer. I avsnitt 4.2.1 ble det registrert at vektingsalgoritmen gjennomsnittlig behøvde en litt lengre tid for å gjennomføre et lokasjonsestimat, sammenliknet med Euklids avstand. Denne forskjellen var 33 millisekunder for iPhonen og 13 millisekunder for Android-telefonen. Sett i sammenheng med den totale tiden som trengs for å utføre et lokasjonsestimat blir denne tidsforskjellen uvesentlig. Tiden som trengs for å gjennomføre et helt lokasjonsestimat, identifisert i dette avsnittet, vil derfor være gjeldene uavhengig av hvilken av de to lokasjonsbestemmende algoritmene som benyttes.

Fase 3 utgjør i gjennomsnitt 44% av tiden brukt på et lokasjonsestimat for Android-telefonen, og 37% av tiden brukt på et lokasjonsestimat for iPhonen. Å generere brukergrensesnitt skal normalt være en lite tidkrevende oppgave, spesielt for en applikasjon med enkel grafikk som for systemet utviklet i forbindelse med denne oppgaven. Ut ifra disse resultatene kan vi konkludere med at den tidsmessige flaskehalsen i systemet er fase 3. Om det ønskes et raskere lokasjonssystem må det dermed utviklets et mer tidseffektivt brukergrensesnitt. En årsak til at fase 3 er tidkrevende er at det kreves mye ressurser av mobiltelefonene for å håndtere bildefilene som benyttes i systemet, dette ble identifisert i avsnitt 3.6.3. Andre kartapplikasjoner som den som følger med telefonene, eller Google Maps, streamer kartinnholdet fra en server. Slike løsninger er også trege sammenliknet med lokasjonssystemets gjennomsnittlige tid nødvendig for å utføre et lokasjonsestimat.

Brukergrensesnittet er ikke en del av selve lokasjonssystemet, men heller en tilleggsfunksjon for å enkelt vise nytten av et slikt system. Det er derfor ikke forsøkt å forbedre tiden som trengs for å gjennomføre fase 3, tatt i betraktning tiden tilgjengelig under arbeidet med denne masteroppgaven. Mobile enheter vil ha mottatt koordinatene fra et lokasjonsestimat etter fase 1 og 2 er gjennomført. Ved å kun studere tidene for de to første fasene oppnår vi dermed et kanskje mer nøyaktig estimat på hvor lang tid lokasjonssystemet trenger for å estimere en lokasjon. Ved å summere de gjennomsnittlige tidene for fase 1 og fase 2 ser vi at:

- Et lokasjonsestimat for Android telefonen i gjennomsnitt vil bli gjennomført på 1104.22 millisekunder.
- Et lokasjonsestimat for iPhone i gjennomsnitt vil bli gjennomført på 1342.58 millisekunder.

Ut i fra tabell 4.4 og 4.5 ser en at årsaken til at et lokasjonsestimat blir utført raskere for Android-telefonen, er at denne mobile enheten trenger kortere tid for å gjennomføre et nettverksskann (fase 1).

4.3 Systemets nøyaktighet

Testene for å identifisere nøyaktigheten av lokasjonssystemet ble delt opp i to faser. Første fase hadde som hensikt å identifisere hvilken av de to lokasjonsbestemmende algoritmene som gir de mest nøyaktige lokasjonsestimatene. Samtidig ga fase 1 også en indikator på systemets nøyaktighet. Fase 2 ble gjennomført i etterkant av fase 1 for å ytterligere identifisere systemets nøyaktighet, ved å tilføre resultatene en større mengde kvantitative data. Begge fasene ble delt opp ut i fra innendørsområder en typisk vil finne ved campus på NTNU. Dette ble i avsnitt 3.1.2 identifisert til å være:

- Korridorer
- Åpne arealer
- Lukkede rom

For hvert område ble nøyaktigheten for iPhone og Android-telefonen testet. Nøyaktigheten av et lokasjonsestimat ble identifisert ved å beregne avstanden fra lokasjonen systemet estimerte og til den reelle lokasjonen. Nøyaktigheten oppgis dermed som avvik i meter mellom reell og estimert lokasjon. Det ble også registrert når det oppstod rom- og etasjefeil. For etasjefeil ble avstanden

mellom reell og estimert lokasjon beregnet som om de to lokasjonene befant seg på samme etasjeplan, det ble dermed ikke tatt hensyn til avstandsforskjell i z-planet.

4.3.1 Fase 1

Hensikten med fase 1 var å identifisere eventuelle forskjeller i ytelse for de to lokasjonsbestemmende algoritmene. For å muliggjøre sammenlinkning av de to algoritmene, ble det gjennomført 25 lokasjonsestimat for hver algoritme med hver mobile enhet for hvert innendørsområde. Dette er best beskrevet ved tabell 4.6.

	iPhone og Android	
	Euklids avstand	*vektingsalgoritme*
Korridorer	25	25
Åpne arealer	25	25
Lukkede rom	25	25

Tabell 4.6: Antall gjennomførte lokasjonsestimat, fase 1.

Ut i fra disse lokasjonsestimatene har den gjennomsnittlige nøyaktigheten blitt beregnet for de to lokasjonsbestemmende algoritmene. Dette er vist i tabell 4.7. Ut i fra resultatene ser en at vektingsalgoritmen gjennomsnittlig gir mer nøyaktige lokasjonsestimat sammenliknet med Euklids avstand, for alle innendørsområdene og for begge de mobile enhetene. Vektingsalgoritmen har spesielt gitt en forbedring i den gjennomsnittlige nøyaktigheten for Android-telefonen. I avsnitt 3.2 ble det identifisert at avviket mellom oppfattet signalstyrke for Android-telefonen sammenliknet med referansemaskinen, er større enn for iPhonen sammenliknet med referansemaskinen. Ut i fra dette ble det antatt at når lokasjonssystemet benytter konkrete RSSI verdier til å estimere en lokasjon, slik som ved bruk av Euklids avstand som lokasjonsbestemmende algoritme, vil lokasjonsestimatene bli mer nøyaktige for iPhonen enn for Android-telefonen. Resultatene i tabell 4.7 bekrefter dette, og en kan konkludere med at å ta i bruk konkrete RSSI verdier for å estimere en lokasjon vil være vanskelig i et lokasjonssystemet som skal gi støtte for bruk av ulike mobile enheter.

Å studere den gjennomsnittlige nøyaktigheten av lokasjonsestimatene gir ikke et helt representativt bilde av systemets ytelse med hensyn på nøyaktighet, dette fordi det var stor spredning i målingene. Med spredning menes forskjellen i nøyaktighet for det mest nøyaktige lokasjonsestimatet, sammenliknet

	iPhone		**Android**	
	Euklids avstand	*Vektingsalgoritme*	*Euklids avstand*	*Vektingsalgoritme*
Korridorer	4	3.52	4.68	4.32
Åpne arealer	6.96	6.8	9.2	5.04
Lukkede rom	5.48	4.68	6.56	3.72

Tabell 4.7: Systemets gjennomsnittlig nøyaktighet i meter, fase 1.

med det mest unøyaktige lokasjonsestimatet. Tabell 4.8 viser spredningen og standardavviket for resultatene i fase 1. For spredningen er både det mest nøyaktige og det mest unøyaktige lokasjonsestimatet presentert.

	iPhone				**Android**			
	Euklids avstand		*Vektingsalgoritme*		*Euklids avstand*		*Vektingsalgoritme*	
	SPR	STD	SPR	STD	SPR	STD	SPR	STD
Korridorer	0-20	5,34	0-20	4,37	0-16	5,73	0-12	3,59
Åpne arealer	1-33	8,23	1-30	7,98	1-40	10,49	1-12	3,46
Lukkede rom	0-25	6,27	0-25	5,58	1-35	7,06	0-22	4,39

Tabell 4.8: Spredning (SPR) og standardavvik (STD) for resultatene i meter, fase 1.

For de fleste lokasjonsestimatene ble det registrert mer nøyaktige gjengivelser av reell lokasjon, enn den identifiserte gjennomsnittlige nøyaktigheten av lokasjonssystemet. Samtidig var det noen lokasjonsestimat som skilte seg ut med et stort avvik mellom reell og estimert lokasjon. Resultatene fra fase 1 vil dermed kunne fordeles i to grupperinger:

- **Gruppering 1:** Lokasjonsestimat der avviket mellom reell og estimert lokasjon tilsvarer den gjennomsnittlige nøyaktigheten eller mindre. Denne grupperingen inneholder de fleste av lokasjonsestimatene.
- **Gruppering 2:** Lokasjonsestimat der avviket mellom reell og estimert lokasjon er stort. Normalt fra ti meter over den gjennomsnittlige nøyaktigheten av et lokasjonsestimat og oppover. Denne grupperingen inneholder få lokasjonsestimat.

At et lokasjonsestimat vil kunne havne i begge disse grupperingene gjør at nøyaktigheten av lokasjonssystemet er svært varierende. Lokasjonssystemets ytelse med hensyn på nøyaktigheten av et lokasjonsestimat er derfor upålitelig. For et pålitelig lokasjonssystem vil avviket mellom reell og estimert lokasjon forholde seg relativt stabilt, slik at en bruker vil ha en konkret feilmargin å forholde seg til.

Ut i fra tabell 4.8 ser en at det er registrert en mindre spredning i resultatene for vektingsalgoritmen enn for Euklids avstand når de benyttes på Android-telefonen. For iPhonen ble det ikke registrert noen slik forbedring. Allikevel ble det registrert en forbedring i standardavvik ved bruk av vektingsalgoritmen for begge smarttelefonene. Det antas derfor vektingsalgoritmen vil gjøre lokasjonssystemet mer pålitelig i tillegg til å øke den gjennomsnittlige nøyaktigheten av lokasjonsestimatene.

For en bruker av lokasjonssystemet vil det være viktig at riktig rom og etasje oppgis ved et lokasjonsestimat. En sentral faktor for systemets ytelse med hensyn på nøyaktighet er derfor antallet rom- og etasjefeil. Antall rom- og etasjefeil identifisert i fase 1 er vist i tabell 4.9. Tabellen viser prosentandelen av lokasjonsestimatene som resulterte i rom- eller etasjefeil. Ut i fra resultatene presentert i tabellen ser en at vektingsalgoritmen også generelt yter bedre med hensyn på rom- og etasjefeil.

	iPhone				**Android**			
	Euklids avstand		*Vektingsalgoritme*		*Euklids avstand*		*Vektingsalgoritme*	
	Romfeil	Etasjefeil	Romfeil	Etasjefeil	Romfeil	Etasjefeil	Romfeil	Etasjefeil
Korridorer	4%	4%	4%	0%	8%	0%	8%	4%
Åpne arealer	0%	12%	0%	4%	0%	12%	0%	8%
Lukkede rom	16%	12%	16%	4%	28%	12%	8%	8%

Tabell 4.9: Prosent av lokasjonsestimatene som ga rom- og etasjefeil, fase 1.

Evaluering fase 1

Vektingsalgoritmen har vist en forbedret nøyaktighet for lokasjonssystemet sammenlinket med Euklids avstand. Både den gjennomsnittlige nøyaktigheten av et lokasjonsestimat og antallet rom- og etasjefeil har blitt forbedret. For fase 2 vil derfor nøyaktigheten av lokasjonssystemet kun bli testet med vektingsalgoritmen.

Det ble for iPhonen under fase 1 identifisert en svært spesiell type feil. Ved to lokasjonsestimat i et lukket rom ble lokasjonen estimert til å være i det forrige rommet lokasjonssystemet hadde blitt testet for. Disse rommene befinner seg på hver sin side av lokasjonssystemets dekningsområde, og mellom dem er det en rekke tykke vegger. Det skal derfor ikke være mulig at det oppstår slike feil. Det er fysisk umulig at en vil kunne høre de samme aksesspunktene i de to rommene. Disse feilene kan derfor ikke skyldes hvordan lokasjonsserveren estimerer lokasjoner, men må være et resultat av lokasjonsdataene serveren mottar fra iPhonen. For at iPhonen skal ha mulighet til å

sende lokasjonsdata tilhørende tidligere lokasjoner må disse ha vært lagret et sted i applikasjonen, og ikke blitt slettet før det gjennomføres nye lokasjonsestimat. Dette stemmer overens med at problemet så ut til å forsvinne i det applikasjonen ble startet på nytt. All objective-c koden som har blitt skrevet for iPhone applikasjonen har derfor blitt gått igjennom. Det ble identifisert at en datastruktur, tilhørende det private rammeverket brukt for å få tilgang til det trådløse nettverkskortet på telefonen, ikke ble tømt før det ble gjennomført et nytt nettverksskann. Før fase 2 ble gjennomført ble det derfor rettet opp i dette, for å identifisere om det ville gi noen forbedringer i systemets nøyaktighet for iPhonen.

Nøyaktigheten av de to lokasjonsestimatene hvor denne typen feil oppstod ble ikke tatt med i beregningen av systemets nøyaktighet, da det var tydelig at dette ikke skyldes designet av lokasjonsfingeravtrykk systemet. I stedet ble det gjennomført to nye lokasjonsestimat, og nøyaktigheten av disse lokasjonsestimatene ble registrert. Det er en mulighet for at dette problemet også kan ha oppstått i mindre skala under andre lokasjonsestimat. I så fall vil feil av denne typen kunne ha påvirket den identifiserte nøyaktigheten av lokasjonssystemet for iPhonen.

4.3.2 Fase 2

I fase 1 ble det identifisert at lokasjonssystemets ytelse med hensyn på nøyaktighet ble forbedret ved å benyttet vektingsalgoritmen sammenliknet Euklids avstand. For fase 2 har derfor vektingsalgoritmen blitt tatt i bruk for samtlige lokasjonsestimat. Hensikten med fase 2 var å oppnå et mer ekstakt mål på systemets nøyaktighet. Dette har blitt gjort ved å gjennomføre flere lokasjonsestimat, for deretter å studere nøyaktigheten av disse. For både iPhone og Android telefonen har det blitt gjennomført:

- 50 lokasjonsestimat i korridorer.
- 50 lokasjonsestimat i åpne arealer.
- 40 lokasjonsestimat i lukkede rom.

Årsaken til at det ble gjennomført færre lokasjonsestimat for lukkede rom, er at dette antallet lokasjonsestimat var tilstrekkelig for å identifisere nøyaktigheten for de lukkede rommene inkludert i systemets dekningsområde. Lukkende rom utgjør en betraktelig mindre del av det totale dekningsområdet sammenliknet med korridorer og åpne arealer (jf. figur 3.1 og 3.2). Den gjennomsnittlige nøyaktigheten identifisert i fase 2 er vist i tabell 4.10.

	iPhone	**Android**
Korridorer	2.68	3.64
Åpne arealer	3.82	5.62
Lukkede rom	3.875	5.35

Tabell 4.10: Systemets gjennomsnittlig nøyaktighet i meter, fase 2.

Sammenliknet med resultatene fra fase 1 er det, for samtlige innendørs områder, registrert en vesentlig ytelses forbedring med hensyn på gjennomsnittlig nøyaktighet for iPhonen i fase 2. Ut i fra dette konkluderes det med at feilen identifisert for iPhonen i fase 1, en datastruktur ikke ble tømt før det ble gjennomført et nytt lokasjonsestimat, svekket systemets nøyaktighet. Nøyaktigheten for iPhonen identifisert i fase 1, vil dermed ikke være representativ. I fase 2 oppstod det heller ingen feil der lokasjonen ble estimert til andre siden av systemets dekningsområde, slik som i fase 1. Systemets nøyaktighet for iPhonen er derfor kun identifisert ut i fra resultatene av testene utført i fase 2. Spredningen og standardavviket for resultatene i fase 2 er vist i tabell 4.11. For iPhonen ble det registrert vesentlig mindre spredning og standardavvik for resultatene i fase 2 sammenliknet med resultatene av fase 1. Dette er sannsynligvis også et resultat av at implementasjonsfeilen oppdaget i fase 1 har blitt rettet opp, og enda en indikator på at lokasjonssystemet nå yter bedre med hensyn på nøyaktighet for iPhonen.

	iPhone		**Android**	
	Spredning	*Standardavvik*	*Spredning*	*Standardavvik*
Korridorer	0-14	3,06	0-23	4,15
Åpne arealer	0-20	4,47	0-26	6,21
Lukkkede rom	0-12	2,79	0-25	5,29

Tabell 4.11: Spredning og standardavvik for resultatene i meter, fase 2.

Selv om nøyaktigheten identifisert for iPhonen i fase 1 ikke kan regnes som gjeldene, vil de resterende funnene i fase 1 fortsatt anses som sentrale. De to lokasjonsbestemmende algoritmene ble begge testet med implementasjonsfeilen for iPhonen. Det antas dermed at forholdet mellom nøyaktighetene for de to algoritmene forholder seg relativt stabilt. Samtidig er det ønskelig at lokasjonssystemet skal kunne benyttes ved hjelp av ulike mobile enheter, vektingsalgoritmen viste en tydelig ytelses forbedring for Android-telefonen. En slik ytelses forbedring vil også oppnås ved bruk av vektingsalgoritmen for andre mobile enheter, med store avvik i oppfattet signalstyrke sammenliknet med referansemaskinen brukt under innsamlingen av lokasjonsdata.

Ved å sammenlikne de gjennomsnittlige nøyaktighetene identifisert i fase 1 og 2 ved bruk av vektingsalgoritmen for Android-telefonen, ser en at fase 2 viser en forbedret ytelse for korridorer, mens den gjennomsnittlige nøyaktigheten for åpne arealer og lukkende rom har gått ned. Det er ikke gjort noen forandringer i implementasjonen for Android-telefonen før fase 2 ble gjennomført. Nøyaktigheten av lokasjonssystemet vil bli påvirket av at mottatt signalstyrke fra trådløse aksesspunkt vil variere. Slike variasjoner skyldes naturlig påvirkning av signaler i en innendørsomgivelse (jf. avsnitt 2.1.4). Forskjellene i de gjennomsnittlige nøyaktighetene identifisert i fase 1 og fase 2 for Android-telefonen, uttrykker dermed naturlige forandringer en vil oppleve for lokasjonssystemets nøyaktighet. Det beste estimatet for systemets nøyaktighet ved bruk av Android-telefonen vil være gitt av de samlede resultatene av fase 1 og fase 2.

Ved å sammenlikne spredningen av resultatene for de to fasene, ser en at spredingen i fase 2 er større for samtlige innendørsområder ved bruk av Android-telefonen. Dette skyldes sannsynligvis at det ble gjennomført færre lokasjonsestimat i fase 1. Det største avviket mellom reell og estimert lokasjon vil være avgjørende for hvor stor spredingen det blir registrert for resultatene, og kan anses som et "worst-case senario". Ved å kun gjennomføre et lite antall lokasjonsestimat risikerer en å ikke registrere dette "worst-case senarioet", resultatet blir dermed en mindre spredning i målingene. Standardavviket er nært knyttet opp til spredningen av målingene, og det ble også registrert et større standardavvik for samtlige innendørsområder for Android-telefonen i fase 2. Disse resultatene kan tyde på at nøyaktigheten identifisert for Android-telefonen i fase 1 var noe misvisende. Dette kan skyldes at det ble gjennomført relativt få lokasjonsestimat for hvert innendørsområde. Ved å kombinere resultatene av fase 1 og fase 2 oppnår en dermed det beste estimatet for lokasjonssystemets reelle ytelse med hensyn på nøyaktighet for Android-telefonen.

	iPhone		**Android**	
	Romfeil	*Etasjefeil*	*Romfeil*	*Etasjefeil*
Korridorer	4%	0%	10%	0%
Åpne arealer	2%	6%	2%	6%
Lukkede rom	20%	7.5%	22.5%	7.5%

Tabell 4.12: Prosent av lokasjonsestimatene som ga rom- og etasjefeil, fase 2.

Antall rom- og etasjefeil identifisert i fase 2 er vist i tabell 4.12. Tabellen viser prosentandelen av lokasjonsestimatene som ga rom- eller etasjefeil for hvert innendørsområde. Ut i fra resultatene i tabell 4.9 og 4.12 ser vi at

spesielt lukkede rom er utsatt for rom- og etasjefeil. Dette kan forklares ved å studere de bygningsmessige omgivelsene for de lukkede rommene inkludert i systemets dekningsområde. Som beskrevet i avsnitt 2.1.4 vil signalene fra de trådløse aksesspunktene bli utsatt for signalforurensning innen de når frem til en mottaker. En vesentlig del av denne signalforurensningen er at signalene vil treffe vegger. I lokasjonssystemet vil vegger bidra til å adskille ulike grupperinger av lokasjoner. Et lukket rom er avgrenset av vegger på alle sider, lokasjonene på innsiden av disse veggene kan ses på som en gruppering av lokasjoner. For at signaler fra aksesspunkt på utsiden av et rom skal kunne nå en mobil enhet inne i rommet må de passere en vegg. Veggene vil redusere signalstyrken eller hindre signalene i å nå en mobil enhet inne i rommet. Hvor stor en veggs påvirkning av signalene blir, vil avgjøres av blant annet materialet og tykkelsen av veggen.

Lokasjonssystemets dekningsområde inneholder fire lukkede rom, og tre av disse rommene har vegger bestående av store vinduer som vender ut mot andre områder inkludert i systemets dekningsområde. Korridorer vil på lik linje som lukkede rom være avgrenset av vegger. I kontrast til de lukkede rommene er korridorerene inkludert i systemets dekningsområde i hovedsak avgrenset av betongvegger. [11] beskriver at vinduer vil skape en mindre reduksjon i signalstyrke sammenlinket med andre materialer slik som betong, samtidig vil også vinduer være tynnere enn betongvegger. At det har blitt registrert et vesentlig antall rom- og etasje feil i de lukkede rommene skyldes derfor hvordan disse områdene er avgrenset i forhold til resten av systemets dekningsområde. Da vinduene gir en begrenset reduksjon i signalstyrke, vil aksesspunkt som definerer lokasjoner på andre siden av denne veggen tydelig kunne høres i et slikt lukket rom. En risikerer dermed at lokasjonen til den mobile enheten blir estimert til å være på andre siden av veggen, altså tilhørende en misvisende gruppering av lokasjoner. Årsaken til at det oppstår etasjefeil, i tillegg til romfeil for de lukkede rommene, er at to av rommene har store vinduer ut mot innendørsarealer som befinner seg på et annet etasjeplan.

Det ble tidligere i dette avsnittet avdekket en implementasjonsfeil for iPhone programvaren. Ved å rette opp i denne feilen ble det identifisert en bedre nøyaktighet, en mindre spredning og mindre standardavvik for lokasjonsestimatene gjennomført for iPhonen. Ut i fra resultatene i tabellene 4.9 og 4.12 ser en at denne feilen ikke har hatt noen innvirkning på antallet rom- og etasjefeil.

4.3.3 Evaluering av systemets nøyaktighet

Dette avsnittet vil evaluere systemets nøyaktighet ut i fra resultatene identifisert i avsnitt 4.3.1 og 4.3.2. I henhold til krav IFK 3 beskrevet i systemets kravspesifikasjon (jf. avsnitt 3.1.2) er det ønskelig å oppgi prosentvis nøyaktighet for lokasjonssystemet, dette har blitt gjort i listen under. Samtidig er også den samlede gjennomsnittlige nøyaktigheten for alle innendørsområdene oppgitt. Under beregningen av disse resultatene har det blitt gått ut i fra resultatene av fase 1 og fase 2 for Android-telefonen. Resultatene for iPhonen er kun hentet fra fase 2.

- **iPhone**
  - 68.57% av lokasjonsestimatene returnerte en lokasjon med maksimal feilmargin på 3 meter.
  - 80.0% av lokasjonsestimatene returnerte en lokasjon med maksimal feilmargin på 5 meter.
  - 90% av lokasjonsestimatene returnerte en lokasjon med maksimal feilmargin på 8 meter.
  - Gjennomsnittlig nøyaktighet: 3,46 meter.
- **Android**
  - 53.35% av lokasjonsestimatene returnerte en lokasjon med maksimal feilmargin på 3 meter.
  - 71.63% av lokasjonsestimatene returnerte en lokasjon med maksimal feilmargin på 5 meter.
  - 90% av lokasjonsestimatene returnerte en lokasjon med maksimal feilmargin på 10 meter.
  - Gjennomsnittlig nøyaktighet: 4,62 meter.

Ut i fra disse resultatene er det tydelig at lokasjonssystemet er mer nøyaktig for iPhonen enn for Android-telefonen. I signalstyrketesten beskrevet i avsnitt 3.2.1, ble det identifisert hvordan iPhonen og Android-telefonen oppfattet signalstyrker fra omliggende trådløse aksesspunkt sammenliknet med referansemaskinen. Resultatene av testen viste tydelig at avviket mellom oppfattet signalstyrke for iPhonen sammenliknet med referansemaskinen, var mindre enn for Android-telefonen sammenliknet med referansemaskinen. Dette ga en tydelig indikator på at lokasjonssystemet vil være mer nøyaktig for iPhonen enn for Android-telefonen når Euklids avstand benyttes. Resultatene

presentert i dette avsnittet viser at lokasjonssystemets nøyaktighet også er bedre for iPhonen ved bruk av vektingsalgoritmen. Dette kan ha en sammenheng med at lokasjonssystemet fortsatt benytter Euklids avstand, som en del av vektingsalgoritmen, for noen lokasjonsestimat der flere lokasjoner har oppnådd en maksimal vekting (jf. avsnitt 3.5.2).

Ut i fra resultatene av testene utført i fase 1 og 2, konkluderes det med at systemets ytelse med hensyn på nøyaktighet ikke tilfredsstiller kravspesifikasjonen. IFK 3 i avsnitt 3.1.2 definerer hva som ønskes av systemets nøyaktighet:

- 85 % av lokasjonsestimatene skal returnere en lokasjon med maksimal feilmargin på 5 meter.
- 75 % av lokasjonsestimatene skal returnere en lokasjon med maksimal feilmargin på 3 meter.

Hverken nøyaktigheten identifisert for iPhonen eller Android-telefonen tilfredsstiller disse kravene. Samtidig har det også blitt oppdaget to faktorer relatert til nøyaktigheten av lokasjonssystemet som vil påvirke lokasjonssystemets pålitelighet:

- I både fase 1 og fase 2 ble det registrert et relativt stort antall rom- og etasjefeil. For en brukers opplevelse av systemets gjengivelse av den reelle lokasjonen er det viktig at estimert lokasjon blir oppgitt i riktig rom og etasje. Da det har blitt identifisert et høyt antall slike feil, vil systemets pålitelighet med hensyn på nøyaktighet bli svekket.
- Det er en stor spredning i resultatene identifisert i både fase 1 og fase 2. Spredningen forteller hvor stort maksimalt avvik mellom reell og estimert lokasjon en vil kunne oppleve for et lokasjonsestimat. Det ble også registrert relativt store standardavvik for resultatene. Standardavvikene forteller hvor mye nøyaktigheten av lokasjonsestimatene normalt vil avvike fra den gjennomsnittlige nøyaktigheten av et lokasjonsestimat. Dette tilsier dermed at det vil variere hvor nøyaktig lokasjonssystemet estimerer en lokasjon.

Resultatene av nøyaktighetstestene viser at før en slik lokaliseringsløsning eventuelt benyttes i en kommersiell sammenheng, er det essensielt å identifisere hvordan en kan øke systemets ytelse med hensyn på nøyaktighet. Utfordringen ligger i å utvikle en nøyaktig metode for å estimere en lokasjon, basert RSSI som har vist seg å fluktuere tidsavhengig for samme lokasjon, og som oppfattes forskjellig avhengig av hva slags utstyr som måler RSSI verdiene. I fase 1 ble det identifisert at vektingsalgoritmen ga en vesentlig

ytelsesforbedring for Android-telefonen sammenliknet med Euklids avstand. Resultatene av signalstyrketesten beskrevet i avsnitt 3.2.1 viste at Android-telefonen oppfattet signalstyrker svært ulikt referansemaskinen. Ut i fra dette antas det derfor at en fornuftig tilnærming for å løse denne utfordringen, vil være å jobbe videre med en algoritme som ser på forholdet mellom signalstyrkene fra ulike aksesspunkt slik som vektingsalgoritmen.

I lokasjonssystemet beskrevet i denne oppgaven benyttes det tre fingeravtrykk for å definere en lokasjon. Vektingsalgoritmen tar i bruk et vektingssystem for å estimere en lokasjon. Hvis flere lokasjoner ender opp med den maksimale vektingen, vil det benyttes Euklids avstand for å ytterligere estimere lokasjonen. For at en skal kunne oppnå en bedre nøyaktighet for lokasjonssystemet kan det derfor være fornuftig å inkludere flere fingeravtrykk for å definere en lokasjon. Dette vil gjøre så det er flere signalstyrkeforhold å benytte under vektingen av lokasjoner for vektingsalgoritmen, og en vil på denne måten minimere behovet for å ta i bruk Euklids avstand.

Kamol Kaemarungsi et al. har gjennomført et forsøk der de ønsker å finne ut hvordan brukeren påvirker mottatt signalstyrke for en mobil enhet[15]. I dette forsøket identifiseres det at standardavviket for mottatt signalstyrke er mindre for svake signaler. Det vil si at en mottaker vil oppleve mindre fluktuering i mottatt signalstyrke for svake signaler. Ut i fra dette trekkes slutningen at signaler som er velegnet for trådløs kommunikasjon ikke nødvendigvis er egnet for lokalisering. I forsøk på å identifisere hvordan lokasjonssystemets ytelsen med hensyn på nøyaktighet kan økes, vil det derfor bli undersøkt om dette funnet kan være med å skape et mer nøyaktig lokasjonssystem. I avsnitt 4.4 vil det bli forsøkt å identifisere et RSSI-intervall for signalstyrker som vil være egnet å bruke i et lokasjonssystem. Om det finnes et slikt intervall vil en kunne øke nøyaktigheten av lokasjonssystemet ved å danne fingeravtrykk fra aksesspunkt som leverer signalstyrker innenfor dette intervallet. Det er ikke funnet annen forskning hvor det har blitt forsøkt å identifisere et slikt intervall.

4.3.4 Feilkilder

Dette avsnittet vil beskrive mulige feilkilder som vil ha kunnet påvirket resultatene for testene av systemets nøyaktighet. Den største årsaken til at systemet estimerer unøyaktige lokasjoner skyldes variasjoner i mottatt signalstyrke for de ulike mobile enhetene, som beskrevet tidligere. Dette anses ikke som en feilkilde. Ikke alle feilkildene beskrevet her er bekreftet at har

funnet sted, men det er en reell mulighet for at slike tilfeller kan ha oppstått under utviklingen og testingen av lokasjonssystemet.

Unøyaktig arbeid under innsamlingen av lokasjonsdata (jf. avsnitt 3.3) vil kunne lede til feil som kan påvirke nøyaktigheten av lokasjonssystemet. For eksempel kan det hende referansemaskinen ikke har befunnet seg på riktig sted i det registreringen av lokasjonsdata har foregått. Det vil da bli registrert misvisende fingeravtrykk for lokasjonen. En slik feil vil kunne lede til små avvik mellom reell og estimert lokasjon på opptil et par meter. Under innsamlingen av lokasjonsdata er det også en mulighet for at det har blitt tastet inn misvisende koordinater i forhold til den reelle lokasjonen. Dette vil da lede til at fingeravtrykkene tilhørende en lokasjon, viser til koordinatene for en annen lokasjon. Det har blitt forsøkt å identifisere slike feil ved at database entitetene med jevne mellomrom ble kontrollert under innsamlingen av lokasjonsdata.

Under testingen av systemets nøyaktighet ble det identifisert at Android-telefonen stadig mister forbindelse til det trådløse nettverket. Dette er en feilkilde som ikke vil påvirke nøyaktigheten av lokasjonssystemet, men som degraderer systemets ytelse. Nettverkskann vil kunne gjennomføres uten at telefonen er koblet til det trådløse nettverket, men for å kunne kommunisere med lokasjonsserveren kreves det nettverksforbindelse. Om en befinner seg på akkurat samme lokasjon vil nettverksforbindelsen som regel opprettholdes, men i det en beveger seg med telefonen vil forbindelsen bli brutt. Under testingen av systemets nøyaktighet måtte en derfor ofte manuelt koble til det trådløse nettverket i det en ankom en ny lokasjon. Dette reduserer nytteverdien av lokasjonssystemet betraktelig. Samtidig oppstår ikke slike problemer for iPhone, dette kan tyde på at problemet vil kunne løses via en programvareoppdatering for Android-telefonen.

Det har vist seg å være en stor spredning i nøyaktigheten av lokasjonsestimatene (jf. tabell 4.8 og 4.11). Når avstanden mellom reell og estimert lokasjon blir lang, blir det vanskelig å beregne denne. Dette kan lede til at denne avstanden for noen lokasjonsestimat har blitt registrert unøyaktig. Det vil i så fall påvirke resultatene av testene for systemets nøyaktighet.

4.4 Identifikasjon av signalstyrkeintervall

Kamol Kaemarungsi et al. beskriver at en vil oppleve mindre fluktuering i mottatt signalstyrke for svake signaler[15]. Dette kan tyde på at signaler som

er velegnet for kommunikasjon ikke nødvendigvis er egnet for lokalisering. I dette avsnittet vil denne hypotesen bli testet, og det vil bli forsøkt å identifisere et intervall for mottatt signalstyrke som vil gi mer stabile fingeravtrykk. Om det finnes et slik intervall vil nøyaktigheten av et lokasjonsfingeravtrykk system kunne økes om det dannes fingeravtrykk ut i fra aksesspunkt med signalstyrker innenfor dette intervallet. Siden det er ønskelig at lokasjonssystemet skal kunne tas i bruk ved hjelp av ulike mobile enheter, vil det være viktig at et slikt intervall vil være uavhengig av hva slags mobil enhet som benyttes. Det er ikke funnet tidligere forskning hvor det har blitt undersøkt om det å danne fingeravtrykk fra aksesspunkt som gir svakere signalstyrker, vil kunne være med å øke nøyaktigheten av et lokasjonssystem.

I forsøk på å identifisere et slik intervall har det blitt gjennomført et sett med tester. Testene gikk ut på å måle variasjonen i mottatt signalstyrke fra ulike aksesspunkt, ved bruk av ulike mobile enheter. Testene har blitt gjennomført med den bærbare datamaskinen, iPhonen og Android-telefonen benyttet under arbeidet med denne oppgaven. For hvert aksesspunkt har det blitt gjennomført 50 målinger av mottatt signalstyrke for hver mobile enhet, totalt ble signalstyrken fra syv aksesspunkt undersøkt. Det har blitt forsøkt å velge ut aksesspunkt registrert med ulike RSSI verdier, slik at fluktuering for både sterke og svake signalstyrker blir identifisert. Normalt vil en innenfor dekningsområdet for dette lokasjonssystemet kunne registrere mellom 8 og 12 aksesspunkt. Resultatene av testene er gjengitt i tabell 4.13. Verdiene er oppgitt som RSSI, og sortert ut i fra den gjennomsnittlige RSSI verdien registrert med iPhonen.

	iPhone		**Android**		**Datamaskin**	
	Gjennomsnitt	*Standardavvik*	*Gjennomsnitt*	*Standardavvik*	*Gjennomsnitt*	*Standardavvik*
AP 1	-54.92	7,67	-56,06	2,98	-48,28	4,17
AP 2	-61,78	6,15	-64,74	3,77	-53,06	6,70
AP 3	-77,30	3,68	-79,56	3,00	-70,22	3,09
AP 4	-80,14	3,69	-86,94	1,90	-71,60	2,96
AP 5	-80,66	2,26	-80,88	1,89	-74,74	2,48
AP 6	-89,58	1,38	-89,28	1,14	-83,66	0,77
AP 7	-90,64	2,35	-91,66	2,15	-81,67	0,78

Tabell 4.13: Fluktuering i mottatt signalstyrke.

Standardavvikene fra tabell 4.13 bekrefter at en generelt vil oppleve mindre fluktuering for svake signaler. Årsaken til at dette fenomenet oppstår er mest sannsynlig et resultat av den omfattende "multipath" signalpåvirkningen en vil oppleve i en innendørsomgivelse (jf. avsnitt 2.1.4). Sterke mottatte

signalstyrker, fra omlag -55 dBm og oppover, vil bli registrert når det er "line-of-sight" mellom aksesspunkt og mobil enhet. Samtidig vil en i et slikt senario også kunne motta signaler fra det samme aksesspunktet som har tatt en annen vei, enn den direkte veien mellom sender og mottaker. I de tilfellene der dette har skjedd vil signalet ha blitt dempet som et resultat av refleksjon, penetrering av vegger og liknende. Dette vil resultere i en betraktelig svakere mottatt signalstyrke. Dermed vil en kunne oppleve relativt stor fluktuering i mottatt signalstyrke for den samme lokasjonen. For svakere signaler fra omlag -70 dBm og nedover, vil det ikke være "line-of-sight" mellom aksesspunktet og den mobile enheten. I en innendørsomgivelse vil det i et slik senario normalt være en vegg mellom sender og mottaker, dette leder til at en vil få en konstant demping i mottatt signalstyrke. På grunn av denne konstante dempingen vil noen signaler, som blir utsatt for ytterligere demping som en følge av veien de tar mellom sender og mottaker, ikke være sterke nok til å bli registrert av mottaker. På denne måten blir det eliminert bort en del av de mulige veiene signalet kan ta mellom aksesspunktet og den mobile enheten, og resultatet blir dermed en mindre fluktuering i mottatt signalstyrke.

Selv om standardavvik for mottatt signalstyrke ser ut til å bli mindre desto svakere signalet blir, vil ikke aksesspunktene som gir de aller svakeste signalstyrkene være mest egnet for å danne fingeravtrykk i et lokasjonssystem. Årsaken til dette er at når mottatt signalstyrke blir tilstrekkelig svak vil en risikere at en ved et vilkårlig tidspunkt ikke hører dette aksesspunktet ved den samme lokasjonen. Om en i et lokasjonssystem danner fingeravtrykk fra aksesspunkt med så svake signalstyrker vil en kunne oppleve at når en bruker gjennomfører et nettverksskann, som en del av et lokasjonsestimat, blir noen av aksesspunktene som i databasen definerer brukerens lokasjon ikke identifisert. Dette vil kunne lede til at lokasjonssystemet estimerer en unøyaktig lokasjon. **AP 6** og **AP 7** i tabell 4.13 var ikke hørbare for alle målingene. Tabell 4.14 viser i prosent hvor ofte disse aksesspunktene ikke var hørbare, beregnet ut i fra de 50 målingene gjennomført for hvert aksesspunkt med hver mobile enhet. Da **AP 6** og **AP 7** ikke alltid var hørbare har gjennomsnitt og standardavvik, presentert i tabell 4.13, blitt beregnet ut i fra færre målinger for disse aksesspunktene enn for de resterende aksesspunktene (**AP 1** til **AP 5**).

	iPhone	**Android**	**Datamaskin**
AP 6	18%	6%	0%
AP 7	10%	30%	76%

Tabell 4.14: Prosentandel av målingene hvor aksesspunktene ikke var hørbare.

4.4.1 Evaluering av signalstyrkeintervall

Dette avsnittet vil forsøke å identifisere et intervall for signalstyrker som vil være gunstige å benytte i et lokasjonssystem basert på fingeravtrykk.

Nedre grense

Den nedre grensen i signalstyrkeintervallet har som hensikt å forsikre om at det ikke dannes fingeravtrykk ut i fra aksesspunkt som ikke alltid er hørbare. Tabell 4.14 viser at to av aksesspunktene ikke var hørbare ved en del av målingene. Ved å studere gjennomsnittlig mottatt signalstyrke fra disse aksesspunktene, for de ulike mobile enhetene, kan det konkluderes med at smarttelefonene (iPhone og Android) vil kunne registrere lavere signalstyrker enn den bærbare datamaskinen. Den nedre grensen for signalstyrke intervallet bør derfor settes ut i fra hvor svake signalstyrker den bærbare datamaskinen vil kunne identifisere. Da **AP 6** og **AP 7** ikke alltid var hørbare, bør den nedre grensen settes høyere enn de gjennomsnittlige mottatte signalstyrkene for disse aksesspunktene. **AP 5** er aksesspunktet med den svakeste gjennomsnittlige signalstyrken som var hørbart ved alle målingene, for alle de mobile enhetene. For den bærbare datamaskinen var gjennomsnittlig signalstyrke for **AP 5** -74,74 dBm, og gjennomsnittlig signalstyrke for **AP 7** -81,67 dBm. Den nedre grensen for signalstyrkeintervallet bør derfor befinne seg mellom disse verdiene. For å være forsikret om at en ikke danner fingeravtrykk ut i fra aksesspunkt som ikke alltid vil være hørbare, antas en RSSI verdi på -75 dBm å være et fornuftig valg som nedre grense i signalstyrkeintervallet.

Øvre grense

Den øvre grensen i signalstyrkeintervallet har som hensikt å forsikre om at det ikke dannes fingeravtrykk ut i fra aksesspunkt som vil gi stor fluktuering i mottatt signalstyrke for en lokasjon. Samtidig vil det være essensielt at en øvre grense ikke blir satt så lavt at for få fingeravtrykk blir knyttet opp mot hver lokasjon. En ser ut i fra tabell 4.13 at det blir registrert store standardavvik for **AP 1** og **AP 2** for iPhonen og den bærbare datamaskinen, sammenliknet med de andre aksesspunktene. For Android-telefonen ble det generelt registrert relativt små forskjeller i standardavvik for de ulike aksesspunktene, sammenliknet med de andre mobile enhetene. **AP 3** ble registrert med den tredje sterkeste gjennomsnittlige signalstyrken, og et betraktelig

mindre standardavvik enn for **AP 1** og **AP 2**. Ut i fra dette kan vi konkludere med at den øvre grensen for signalstyrkeintervallet bør ligge mellom den gjennomsnittlige mottatte RSSI verdien for **AP 2** og den gjennomsnittlige mottatte RSSI verdien for **AP 3**. For å unngå et for snevert signalstyrkeintervall antas det å være fornuftig å sette den øvre grensen nærme den laveste gjennomsnittlige RSSI verdien registrert for **AP 2**. Den antas derfor at en RSSI verdi på -65 dBm vil være en fornuftig øvre grense for signalstyrkeintervallet.

Bruk av signalstyrkeintervall

Grensene for signalstyrkeintervallet identifisert i dette avsnittet må ses som myke, og de må kanskje forandres noe avhengig av hvor lokasjonssystemet skal implementeres. Å benytte et slik signalstyrkeintervall vil kun være hensiktmessig i områder der den trådløse infrastrukturen er godt utbygget, og en vil høre en rekke aksesspunkt ved hver lokasjon en ønsker å inkludere i lokasjonssystemet. Det bør tas i betraktning at hvor svake signaler de mobile enhetene kan registrere vil kunne variere fra enhet til enhet. Det er derfor essensielt å kontrollere at den nedre grensen av signalstyrkeintervallet ikke er så lav at det blir dannet fingeravtrykk fra aksesspunkt som ikke vil være hørbare for noen mobile enheter.

Den øvre grensen må kontrolleres i forhold til hvor mange aksesspunkt som er hørbare for hver lokasjon. Hvor den øvre grensen settes, må dermed ses i sammenheng med hvor godt utbygget den trådløse infrastrukturen er i bygningen/området lokasjonssystemet skal implementeres. Om den øvre grensen settes for lavt vil en kunne risikere å knytte for få fingeravtrykk til hver lokasjon. Dette vil gjøre det vanskelig å estimere lokasjoner, og lokasjonssystemet vil kunne ende opp som unøyaktig. Hvis dette er tilfellet vil signalstyrkeintervallet fungere mot sin hensikt. Om det viser seg at intervallet vil gi for få fingeravtrykk for å identifisere en lokasjon, vil det være mer hensiktmessig å øke den øvre grensen enn å senke den nedre grensen. Dette for å unngå å definere en lokasjon ved hjelp av aksesspunkt som ikke alltid er hørbare. Om innsamlingen av lokasjonsdata skal gjennomføres med en mobil enhet som registrer signalstyrke fra omliggende trådløst nettverk svakere enn de andre mobile enhetene, slik som det er identifisert for Android-telefonen i denne oppgaven, vil det kanskje være hensiktmessig å forskyve både den øvre og den nedre grensen av signalstyrkeintervallet litt nedover.

Kapittel 5

Evaluering

Dette avsnittet vil evaluere lokasjonssystemet utviklet i forbindelse med denne masteroppgaven. Det vil fokuseres på hvordan systemet tilfredsstiller kravspesifikasjonen. I etterkant av evalueringen vil det bli foreslått noen forbedringer av lokasjonssystemet.

5.1 Funksjonelle krav

Dette avsnittet vil evaluere hvordan lokasjonssystemet tilfredsstiller de funksjonelle kravene utledet i forkant av implementasjonen.

FK1: *Det skal utvikles et lokasjonssystem for WLAN basert på lokasjonsfingeravtrykk, beregnet for innendørs bruk.*

Dette kan ses som et overordnet kravet til systemet. Det beskriver på et overordnet plan hva som skal implementeres, og hva slags teknologi som skal benyttes. Implementasjonen av lokasjonssystemet står klart i samsvar med dette kravet.

FK2: *Systemets design skal være distribuert, og lokasjonsestimeringen skal foregå på serversiden av systemet.*

Dette er et krav som beskriver hvordan systemet skal designes. Lokasjonssystemet er utviklet i samsvar ved med dette kravet.

FK3: *Systemet skal levere lokasjonen til brukeren innenfor systemets dekningsområde. Lokasjonen skal oppgis i form av koordinater og grafisk representert på et kart.*

Testene for å identifisere systemets nøyaktighet (jf. avsnitt 4.3) ble utført innenfor systemets dekningsområde, og det ble for samtlige lokasjonsestimat returnert en lokasjon. En estimert lokasjon blir markert i brukergrensesnittene til klientløsningene ved hjelp av en rød prikk på et kart som viser hvor brukeren befinner seg (se figur 3.7 og 3.9). Årsaken til at den estimerte lokasjonen skal oppgis som koordinater er at det da blir enklere å utvikle tjenester basert på lokasjonssystemet. Grunnet relativt små skjermer vises ikke koordinatene for den estimerte lokasjonen i brukergrensesnittet, men klientløsningene mottar den estimerte lokasjonen som koordinater fra serveren. Disse koordinatene benyttes for å bestemme hvilke piksler i kartene som tilsvarer brukeres lokasjon, og dermed hvor den røde prikken skal plasseres. Klient programvaren er dermed i besittelse av koordinatene returnert for et lokasjonsestimat, og eventuelle tjenester basert rundt lokasjonssystemet vil kunne få tilgang til disse koordinatene. Dette kravet anses dermed som tilfredstilt.

Om systemet benyttes utenfor dekningsområdet vil lokasjonen innenfor dekningsområdet som best matcher dataene lest inn fra omliggende trådløse nettverk bli returnert. Dette er såfremt minst to av de tre aksesspunktene med sterkest RSSI verdi, identifisert under nettverksskannet, definerer en eller flere lokasjoner i databasen. Om dette ikke er tilfellet vil det ikke bli estimert en lokasjon for brukeren.

FK4: *Systemet skal være nettsentrert.*

Lokasjonsdataene har blitt samlet inn systematisk ved å følge et kart over dekningsområdet med rutenett. Hver rute definerer en lokasjon (jf. avsnitt 3.3). De innsamlede lokasjonsdataene er lagret i en database som vil administreres av netteier. Systemet har ingen funksjonalitet som gir brukere mulighet til å registrere nye lokasjoner, eller justere allerede eksisterende lokasjonsdata. Systemet anses dermed som nettsentrert.

FK5: *Systemets design skal være plattformuavhengig. Slik at det tilrettelegges for bruk av alle mobile terminaler/bærbare datamaskiner.*

Det stilles tre krav til en mobil enhet for at den skal kunne få estimert lokasjoner av lokasjonssystemet:

1. Det må være mulig å utvikle og kjøre programvare på den mobile enheten.
2. Den mobile enheten må ha mulighet til å gjennomføre nettverkskann. Dette vil kreve at den har WLAN utstyr.
3. Den mobile enheten må kunne sette opp en socketforbindelse til lokasjonssystemets server.

Disse kravene henger i stor grad sammen. For at det skal være mulig å bestemme når den mobile enheten skal gjennomføre et nettverksskann må krav 1 være oppfylt. Hvis krav 1 og 2 er oppfylt er krav 3 i de aller fleste tilfeller også tilfredstilt. Alle nyere bærbare datamaskiner tilfredsstiller disse kravene. Samtidig vil de fleste andre typer nyere mobile enheter, slik som smarttelefoner og PDAer også tilfredsstille disse kravene. Dette innebærer blant annet iPhone og Android-telefoner. Lokasjonssystemet er dermed designet slik at det skal kunne benyttes uavhengig av de mobile enhetenes plattform.

FK6: *Systemet skal være dynamisk og skalerbart med hensyn på utvidelse av dekningsområde og funksjonalitet.*

Systemets virkemåte er uavhengig av hvor det benyttes. For å inkludere nye dekningsområder må det gjennomføres en innsamling av lokasjonsdata for dette området. Om det nye dekningsområdet er utenfor rekkevidden av kartverket, må også kartverket byttes ut. Om dette er gjennomført vil systemet kunne fungere overalt hvor det finnes trådløse nettverk. Verktøyet laget for å registrere nye lokasjoner vil alltid kunne benyttes under innsamlingen av lokasjonsdata (jf. avsnitt 3.7).

Ny funksjonalitet kan legges til i systemet ved at det skrives kode for denne funksjonaliteten. Systemets arkitektur er bygget opp slik at funksjonalitet er tilhørende forskjellige komponenter. Ny funksjonalitet kan implementeres som en ny komponent, uten at det er nødvendig med store forandringer i allerede eksisterende kode. Nye komponenter vil kunne benytte allerede eksisterende funksjonalitet, ved å gjennomføre metodekall på komponenter med den ønskede funksjonaliteten.

FK7: *Lokasjoner estimert av systemet skal kunne oppgis som GPS standard koordinater.*

Denne funksjonaliteten er forsikret ved at det er definert et GPS referansepunkt i systemets koordinatsystem. Dette punktet er koordinatsystemets origo, og er lokalisert inntil et vindu. Det vil dermed være mulig å stå inntil eller å gå på utsiden av dette vinduet for å registrere GPS-koordinatene for dette punktet. Om GPS-koordinatene som tilsvarer koordinatsystemets origo er kjent, vil en enkelt kunne regne ut hvilke GPS-koordinater en lokasjon returnert av systemet tilsvarer.

FK8: *Systemet skal være modulært oppbygget for å muliggjøre gjenbruk av funksjonelle komponenter.*

Systemets funksjonalitet er fordelt i komponenter. Hver komponent utgjør en essensiell del av et lokasjonsestimat. Systemets funksjonelle komponenter kan

fordeles til de tilhørende serversiden av systemet og de tilhørende klientsiden av systemet. Serversiden inneholder i grove trekk funksjonelle komponenter for å aksessere databasen, estimere lokasjoner, og håndtere socketforbindelser med klientene. Klientsiden inneholder i hovedsak funksjonelle komponenter for å gjennomføre nettverksskann og kommunisere med serveren. Disse funksjonalitetene er separert fra hverandre slik at de enkelt vil kunne flyttes over i ny programvare, dette er forutsatt at den nye programvaren utvikles ved hjelp av samme programmeringsspråk. Dette kravet anses dermed som tilfredstilt.

FK9: *Systemet skal ha handlingsbasert lokasjonsestimering. Med dette menes at en bruker må gjennomføre en handling for å motta sin lokasjon. Dette vil gjøre det enklere å teste systemets virkemåte etter ferdig implementasjon. Samtidig vil også en handlingsbasert lokasjonsuthenting kreve minst mulig ressurser av brukerutstyret. Dette skal ikke begrense muligheten for senere å implementere automatisk lokasjonsoppdatering.*

Dette er implementert, og kan ses ut i fra klientløsningenes brukergrensesnitt vist i figur 3.6 og 3.8. Handlingsbasert lokasjonsestimering er realisert ved at et lokasjonsestimat blir gjennomført når brukeren trykker på en knapp. Det vil være enkelt å senere implementere automatisk lokasjonsestimering. Dette kan for eksempel realiseres via en tråd som kjører i bakgrunnen av systemet, og etterspør lokasjonsestimat ved faste tidsintervall.

FK10: *Systemet skal utvikles med hensyn på å kreve minst mulig batterikapasitet av brukerutstyret.*

På grunn av handlingsbaserte lokasjonsestimat vil nettverksskann kun bli gjennomført når en bruker ønsker å få estimert sin lokasjon, og selve nettverksskannet tar relativt kort tid. Samtidig blir det også sendt minimalt med data mellom klient og server per lokasjonsestimat. Å gjennomføre et lokasjonsestimat anses dermed som lite krevende med hensyn på de mobile enhetenes batteri kapasitet. For at lokasjonssystemet skal fungere må de mobile enhetene være koblet til det trådløse nettverket, slik at de vil kunne kommunisere med lokasjonssystemets server. Dette vil være faktoren som i størst grad vil påvirke batteribruken for de mobile enhetene. For iPhone oppgis det at batterikapasiteten vil støtte opptil 9 timer Internett bruk via Wi-Fi[1]. Det ble ikke funnet noe slik konkret mål for HTC Hero, men batterikapasiteten antas og være omtrent det samme. Dette tilsier av lokasjonssystemet vil kunne brukes aktivt i omtrent 9 timer for de to smarttelefonene.

FK11: *Systemets funksjonalitet skal tilrettelegges slik at den enkelt skal kunne benyttes av tredjeparts utviklere.*

[1] http://www.apple.com/batteries/iphone.html

Det har under implementasjonen av lokasjonssystemet ikke blitt aktivt tilrettelagt for at systemets funksjonalitet skal kunne benyttes av tredjeparts utviklere. Det vil allikevel være mulig å gjennomføre dette. Det vil derfor her bli beskrevet noen mulig tilnærminger for hvordan dette kan løses.

- Den enkleste måte å gjennomføre dette vil være å integrere koden fra klientløsningene i eventuelle nye applikasjoner. Dette er allikevel noe tungvindt om det skal utvikles en rekke applikasjoner fra flere ulike utviklere.
- Android-plattformen tillater flere applikasjoner å kjøre i parallell, og det er tilrettelagt for at applikasjoner skal kunne kommunisere med hverandre. Dette kan for eksempel gjøres ved at en applikasjon sender ut en *Broadcast Intent* med informasjonen den vil formidle til andre applikasjoner, og de andre applikasjonene tar i mot denne Intent meldingen ved hjelp av en *Broadcast Reciever* (jf. avsnitt 2.5). På denne måten vil lokasjonssystem applikasjon kunne formidle koordinatene mottatt fra et lokasjonsestimat til applikasjoner laget av tredjeparts utviklere.
- iPhone OS tillater ikke å kjøre flere applikasjoner i parallell. Løsningen beskrevet over for Android-plattformen vil dermed ikke fungere for iPhone. En måte å løse denne utfordringen for iPhone vil dermed være å tilgjengeliggjøre funksjonaliteten av lokasjonssystemet via biblioteker.
- Det er også en mulighet å tillate tredjeparts applikasjoner å kommunisere direkte med lokasjonsserveren, og på denne måten få tilgang den mobile enhetenes lokasjon. Hvis denne løsningen velges vil det være essensielt at det implementeres noen sikkerhetstiltak, slik at tredjepart applikasjoner kun vil få tilgang til lokasjonen for den enkelte mobile enhet.

5.2 Ikke-funksjonelle krav

Dette avsnittet vil evaluere hvordan lokasjonssystemet tilfredsstiller de ikke-funksjonelle kravene, utledet for å beskrive den ønskede kvaliteten av lokasjonssystemets tjeneste.

IFK1: *Dekningsområde: Systemet utviklet i forbindelse med denne oppgaven skal være en test på om lokasjonsfingeravtrykk er en egnet måte å skape et lokasjonssystem i store bygninger slik som ved NTNU. I den anledning er det essensielt at dekningsområdet for denne prototypen er representativt for slike*

bygninger. Dekningsområdet skal derfor dekke typiske innendørsområder en vil finne på campus, da disse antas å være representative for store bygninger. Dette innbærer:

- *Åpne arealer, et eksempel på dette er Glassgården i El-bygget.*
- *Avlukkende rom, dette innbærer blant annet lese og datasaler.*
- *Korridorer.*

Samtidig skal dekningsområdet være over to etasjer. Ved å ta i bruk et slikt dekningsområde vil utfordringer relatert til bygningsmessige omgivelser bli identifisert.

Figur 3.1 og 3.2 viser lokasjonssystemets dekningsområde. Ut i fra disse figurene ser en at dekningsområdet er lokalisert over to etasjer og omfatter alle de arealene som ble definert som typiske for campus. Alle arealtypene er også representert flere ganger. Det har blitt registrert 54 ulike aksesspunkt tilhørende eduroam nettverket i systemets database. Dette er kun aksesspunktene som har blitt observert med en av de tre sterkeste RSSI verdiene, for en eller flere lokasjoner. Det er sannsynlig at en under innsamlingen av lokasjonsdata har observert flere aksesspunkt en dette. Det antas dermed at systemet har avdekket hvordan et lokasjonssystem, basert på lokasjonsfingeravtrykk, vil fungere i store bygninger med omfattende trådløs infrastruktur.

IFK2: *Responstid: Systemet skal gjennomføre et lokasjonsestimat i løpet at ett sekund.*

Resultatene identifisert i avsnitt 4.2 viser at den gjennomsnittlige tiden det tar å gjennomført et lokasjonsestimat er 2133.80 millisekunder for iPhone og 1968.76 millisekunder for Android-telefonen. Disse gjennomsnittene inkluderer tiden klientløsningen trenger for å grafisk vise den estimerte lokasjonen i brukergrensesnittet, noe som har vist seg å være tidkrevende. De mobile enhetene mottar den estimerte lokasjonen i forkant av at brukergrensesnittet genereres, dermed vil tiden det tar å gjennomføre et lokasjonsestimat også kunne identifiseres som 1342.58 millisekunder for iPhone og 1104.22 millisekunder for Android-telefonen. Disse gjennomsnittlige tidene er ikke i regnet tiden det tar å generere brukergrensesnittene for de mobile enheten.

Ut i fra dette ser en at kravet til lokasjonssystemets responstid ikke er tilfredstilt. Det er allikevel registrert en vesentlig forbedring i systemets responstid sammenliknet med resultatene fra prosjektarbeidet gjennomført høsten 2009. I prosjektarbeidet ble den gjennomsnittlige tiden nødvendig for å gjennomføre et lokasjonsestimat identifisert til å være 2929 millisekunder[32]. 95.6% av den gjennomsnittlige tiden for et lokasjonsestimat gikk da med til å utføre

nettverksskann. Ytelses forbedringen skyldes derfor at nettverkskann, for å identifisere omliggende trådløs infrastruktur, blir gjennomført raskere for iPhonen og Android-telefonen enn for den bærbare datamaskinen slik det var implementert i [32].

IFK3: *Nøyaktighet: 85 % av lokasjonsestimatene skal returnere en lokasjon med maksimal feilmargin på 5 meter. 75 % av lokasjonsestimatene skal returnere en lokasjon med maksimal feilmargin på 3 meter.*

I avsnitt 4.3.3 ble nøyaktigheten av lokasjonssystemet identifisert, resultatene viste at:

- **iPhone**
  - 80.0% av lokasjonsestimatene returnerte en lokasjon med maksimal feilmargin på 5 meter.
  - 68.57% av lokasjonsestimatene returnerte en lokasjon med maksimal feilmargin på 3 meter.
- **Android**
  - 71.63% av lokasjonsestimatene returnerte en lokasjon med maksimal feilmargin på 5 meter.
  - 53.35% av lokasjonsestimatene returnerte en lokasjon med maksimal feilmargin på 3 meter.

Ut i fra disse resultatene ser vi at kravet ikke er tilfredstilt. Å øke nøyaktigheten av lokasjonsestimatene når det skal tilrettelegges for bruk av ulike mobile enheter, har vist seg å være hovedutfordringen for lokasjonssystemet. I avsnitt 5.3.1 blir det diskutert en mulig tilnærming til hvordan en kan øke lokasjonssystemets ytelse med hensyn på nøyaktighet.

IFK4:*Nøyaktighet 2: Systemets nøyaktighet skal ikke påvirkes av hva slags mobil enhet som benyttes.*

Resultatene presentert over viser at systemet gjennomsnittlig er mer nøyaktig for iPhonen enn for Android-telefonen, dette kravet er dermed ikke tilfredstilt. I avsnitt 4.3.3 ble det identifisert at denne ytelses forskjellen kan ha en sammenheng med at vektingsalgoritmen benytter Euklids avstand for lokasjonsestimat der flere lokasjoner har oppnådd en maksimal vekting.

IFK5: *Pålitelighet: Systemet skal alltid oppgi en lokasjon så lenge brukeren befinner seg innenfor systemets dekningsområde, og den mobile enheten er koblet til det trådløse nettverket.*

For alle testene av systemets ytelse beskrevet i kapittel 4 ble det returnert en lokasjon. Dette kravet er dermed oppfylt.

IFK6: *Systemet skal takle 1000 brukere per time, uten at dette leder til en forverring i systemets responstid.*

Lokasjonssystemets server er implementert slik at flere brukere skal kunne benytte seg av systemet samtidig. Dette er realisert ved at det opprettes en ny tråd for hvert etterspurte lokasjonsestimat (jf. avsnitt 3.5.3 under ClientHandler). Det har blitt kontrollert at denne funksjonaliteten fungerer ved å etterspørre lokasjonsestimat for de to smarttelefonene samtidig, men det har ikke blitt tid til å identifisere hvordan lokasjonssystemets server takler en stor brukergruppe. Det er dermed usikkert hvorvidt kravet er tilfredstilt. Årsaken til at det ble nedprioritert å teste hvor stor brukergruppe lokasjonssystemet kan håndtere, er at dette ikke er direkte relatert til lokasjonssystemet funksjonalitet. Samtidig bør det være relativt trivielt å implementere en velfungerende serverløsning som kan håndtere store brukergrupper, da dette er realisert i en rekke allerede eksisterende IT-løsninger.

5.3 Forslag til forbedringer

Ut i fra evalueringen ser en at systemet i stor grad tilfredsstiller de funksjonelle kravene, mens de ikke-funksjonelle kravene i mindre grad er tilfredstilt. Generelt konkluderes det derfor med at lokasjonssystemet leverer den ønskede tjenesten, men ikke med kvaliteten det er ønsket for tjenesten.

I de kommende avsnittene vil det bli lagt frem noen forslag til hvordan kvaliteten av lokasjonstjenesten kan forbedres.

5.3.1 Nøyaktighet

Den største utfordringen for lokasjonssystemer basert på lokasjonsfingeravtrykk er i denne oppgaven identifisert til å være at ulike mobile enheter oppfatter signalstyrkene fra WLAN-aksesspunkt forskjellig. Dette gjør det vanskelig å identifisere en lokasjonsbestemmende algoritme som vil gi nøyaktige lokasjonsestimat, uavhengig av hva slags mobil enhet som benyttes.

I avsnitt 4.3.1 ble det identifisert at en algoritme som i større grad ser på forholdet i registrert signalstyrke fra ulike aksesspunkt vil gi mer nøyaktige lokasjonsestimat, sammenliknet med en algoritme som utelukkende benytter

konkrete RSSI verdier for å estimere en lokasjon. Det ble derfor i avsnitt 4.3.3 foreslått at ved å inkludere flere fingeravtrykk for å definere hver lokasjon, vil en i større grad kunne benytte forholdet mellom aksesspunktenes signalstyrke til å estimere en lokasjon. Dette vil kunne øke lokasjonssystemets nøyaktighet. Videre ble det i avsnitt 4.4 identifisert en mindre fluktuering for svake signaler enn for sterke signaler. Dette indikerer at å danne fingeravtrykk fra aksesspunkt som leverer sterke signalstyrker for en lokasjon, vil kunne degradere nøyaktigheten av lokasjonssystemet. Det ble derfor forsøkt å identifisere et intervall for hvilke signalstyrker som er gunstige å benytte i et lokasjonssystem basert på fingeravtrykk. Dette intervallet ble satt til å være fra -75 dBm til -65 dBm. Disse grensene må anses som myke, og må kanskje forandres avhengig av den trådløse infrastrukturen i området der lokasjonssystemet skal implementeres.

Å benytte fingeravtrykk innenfor dette intervallet, og samtidig ta i bruk flere fingeravtrykk for å definere en lokasjon, antas dermed som er fornuftig tilnærming til hvordan lokasjonssystemets nøyaktighet kan forbedres. Dette forutsetter at det benyttes en lokasjonsbestemmende algoritme som ser på forholdet mellom aksesspunktenes signalstyrke, tilsvarende som vektingsalgoritmen beskrevet i avsnitt 3.5.2.

5.3.2 Responstid

Slik lokasjonssystemet er implementert i dag utgjør genereringen av brukergrensesnitt 44% av den gjennomsnittlige tiden brukt på et lokasjonsestimat for Android-telefonen, og 37% av den gjennomsnittlige tiden brukt på et lokasjonsestimat for iPhonen. Dette tilsvarer henholdsvis 864.54 og 791.22 millisekunder. Brukergrensesnittene er dermed svært ineffektive med hensyn på tidsbruk. Den enkleste måten å oppnå en bedre responstid for lokasjonssystemet vil dermed være å effektivisere tidsbruken for brukergrensesnittene.

Samtidig har det blitt vektlagt at brukergrensesnittene ikke utgjør en del av selve lokasjonssystemet, men heller er en tileggstjeneste for å enklere kunne vise brukerens lokasjon. I avsnitt 4.2.2 ble det identifisert at innlesning av informasjon fra omliggende trådløs infrastruktur utgjør 37% av den gjennomsnittlige tiden brukt på et lokasjonsestimat for Android-telefonen, og 53% av gjennomsnittlige tiden brukt på et lokasjonsestimat for iPhonen. Selv om denne fasen av et lokasjonsestimat er betraktelig forbedret med hensyn på tidsbruk sammenliknet med implementasjonen i [32], vil det være mulig å ytterligere minimere tiden det tar å gjennomføre denne fasen av et lokasjonsestimat. Dette kan realiseres ved at det utføres nettverksskann med jevne

mellomrom, og resultatene av disse lagres i klient programvaren. På denne måten er informasjonen nødvendig for å estimere en lokasjon tilgjengelig i det brukeren ønsker at det skal bli gjennomført et lokasjonsestimat. Dette vil kunne påvirke nøyaktigheten av et lokasjonssystemet. Om brukeren forflytter seg fra en lokasjon til en annen, og det ikke har blitt utført et nytt nettverksskann før det blir gjennomført et lokasjonsestimat, vil lokasjonen bli estimert basert på lokasjonsdata tilsvarende en tidligere lokasjon brukeren har befunnet seg ved. Dermed vil den estimerte lokasjonen kunne oppfattes som misvisende, selv om lokasjonsserveren estimerte rett lokasjon basert på de mottatte lokasjonsdataene.

Det har i denne oppgaven blitt identifisert at den største utfordringen ved lokasjonssystemet er å identifisere hvordan nøyaktigheten kan forbedres. Det anses dermed som lite gunstig å implementere slike automatiske nettverksskann, før lokasjonssystemet har en tifredsstillende ytelse med hensyn på nøyaktigheten av lokasjonsestimat. Automatiske nettverksskann vil også lede til at lokasjonssystemet vil ha et høyere batteri forbruk. Årsaken til dette er at det sannsynligvis vil bli gjennomført flere nettverksskann når disse utføres med jevne mellomrom, enn når nettverksskann kun blir gjennomført i det brukeren anmoder et lokasjonsestimat.

Kapittel 6

Konklusjon

I forbindelse med denne masteroppgaven har det blitt designet, implementert og testet en nettsentrert lokaliseringsløsning for WLAN. Løsningen baserer seg på innsamling av lokasjonsdata i form av lokasjonsfingeravtrykk. Hensikten med oppgaven har vært å undersøke hvorvidt en lokaliseringsløsning basert på fingeravtrykk vil være en fornuftig måte å implementere et innendørs lokasjonssystem i store bygninger slik som ved NTNU campus. Lokasjonsløsningen har et omfattende dekningsområde i EL-bygget ved NTNU Gløshaugen på 856 kvadratmeter, fordelt over to etasjer. Dekningsområdet inneholder alle innendørsarealer som har blitt identifisert som typiske ved campus, og anses dermed som representativt for store bygninger.

Kjernen av lokasjonssystemet er en server som inneholder programvare samt en database med de innsamlede lokasjonsdataene. Klienter som ønsker å få estimert sin lokasjon, må gjennomføre et nettverkskann for å samle inn lokasjonsdata fra omliggende trådløse nettverk. Deler av disse lokasjonsdataene vil bli sendt til serveren over en socketforbindelse. Lokasjonsserveren vil basert på de mottatte lokasjonsdataene estimere klientens lokasjon, og returnere denne lokasjonen til klienten. Dette er en universal lokaliseringsløsning som vil kunne fungere i alle trådløse nettverk, og som baserer seg på allerede eksisterende trådløs infrastruktur. For klientene er løsningen utelukkende programvare basert. Alle klienter med WLAN-støtte og mulighet for å kjøre programvare vil kunne benytte seg av lokasjonstjenesten. Dette er bevist ved at det er implementert klientløsninger for to ulike typer smarttelefoner, for iPhone og den Android baserte HTC Hero.

Ulike mobile enheter vil oppfatte signalstyrker fra aksesspunkt forskjellig for samme lokasjon, avhengig av blant annet antennen og designet av de mobile

enhetene. Samtidig vil en i en innendørsomgivelse oppleve stor fluktuering i mottatt signalstyrke som en følge av blant annet refleksjon og multipath. Dette gjør det til en krevende oppgave for lokasjonsserveren å estimere riktig lokasjon, uavhengig av hva slags mobil enhet som benyttes. Euklids avstand har tidligere blitt ansett som en svært egnet lokasjonsbestemmende algoritme, i en lokaliseringsløsning basert på fingeravtrykk. Det har i denne oppgaven blitt bevist at når lokaliseringsløsningen skal gi støtte for bruk av ulike mobile enheter, gir Euklids avstand ikke lenger tilfredsstillende resultater med hensyn på nøyaktighet. Dette skyldes at algoritmen tar utgangspunkt i konkrete signalstyrkeverdier, som vil variere avhengig av hva slags mobil enhet som har registrert disse. Det har derfor i stedet blitt implementert en alternativ lokasjonsbestemmende algoritme som benytter et vektingssystem, basert på forholdet mellom mottatt signalstyrke fra ulike aksesspunkt, for å estimere klientenes lokasjon. Denne vektingsalgoritmen økte ytelsen av lokaliseringsløsningen sammenlinket med Euklids avstand.

I etterkant av implementasjonen ble det gjennomført tester for å identifisere lokaliseringsløsningens ytelse med hensyn nøyaktighet og tidsbruk. Den gjennomsnittlige tiden det tar å estimere en lokasjon for Android-telefonen er 1104 millisekunder, mens en det i gjennomsnitt tar 1343 millisekunder å estimere en lokasjon for iPhone. Disse gjennomsnittsverdiene er ikke iberegnet tiden det tar å grafisk presentere lokasjonen i et brukergrensesnitt.

Nøyaktigheten av et lokasjonsestimat har blitt funnet ved å måle avvik i luftlinje mellom estimert og reell lokasjon. Det gjennomsnittlige avviket mellom estimert og reell lokasjon ble for iPhonen funnet til å være 3,46 meter, mens for Android-telefonen var dette 4,62 meter. Selv om vektingsalgoritmen økte lokaliseringsløsningens ytelse med hensyn på nøyaktighet, anses nøyaktigheten fortsatt ikke som tilfredsstillende. Det har derfor blitt forsøkt å identifisere en måte en vil kunne øke nøyaktigheten av lokaliseringsløsningen.

Det har blitt identifisert at det vil oppleves en større fluktuering i mottatt signalstyrke for sterke signaler enn for svake signaler. Dette tilsier at det er mer egnet å benytte svake signalstyrker i en lokaliseringsløsning basert på fingeravtrykk. Det har derfor blitt forsøkt å identifisere et intervall for signalstyrker som vil være ideelle å benytte i en lokaliseringsløsning. Intervallet ble satt til å strekke seg fra -75 dBm til -65 dBm. Grensene satt i dette intervallet må anses som myke, og må kanskje forandres noe avhengig av den trådløse infrastrukturen i området der lokaliseringsløsningen implementeres. Ved benytte flere fingeravtrykk for å definere en lokasjon i serverens database, vil det være flere signalstyrkeforhold å sammenlikne for vektingsalgoritmen.

Det antas derfor at ved å øke antallet fingeravtrykk, og samtidig danne fingeravtrykk ut i fra signalstyrker som befinner seg innenfor det identifiserte signalstyrkeintervallet, vil en kunne øke nøyaktigheten av lokaliseringsløsningen.

Dette arbeidet beviser at det er mulig å utvikle en innendørs lokaliseringsløsning basert på fingeravtrykk, som kan benyttes av mobile enheter med ulik underliggende hardware og plattform. Testresultatene viser imidlertid at det må undersøkes nærmere hvordan en kan øke nøyaktigheten for et slikt system, før det kan benyttes til kommersiell bruk.

6.1 Videre arbeid

Dette avsnittet vil legge frem noen forslag til videre arbeid, som ytterligere vil kunne belyse hvordan en lokaliseringsløsning basert på fingeravtrykk vil fungere i store bygninger.

Den største fremtidige utfordringen vil være å identifisere hvordan en kan øke nøyaktigheten og samtidig tillate bruk av ulike mobile enheter. Det har i denne oppgaven blitt foreslått en fremgangsmåte for hvordan dette kan løses. Det vil være interessant å se resultatene av en implementasjon der denne fremgangsmåten har blitt benyttet. Det vil i så fall være essensielt at dekningsområdet for denne implementasjonen har en tilsvarende størrelse som benyttet i denne oppgaven, for at det skal være mulig å sammenlikne resultatene. Om det viser seg at en slik tilnærming ikke gir en vesentlig ytelses forbedring med hensyn på nøyaktighet, vil det være essensielt å identifisere nye tilnærminger til hvordan nøyaktigheten skal kunne forbedres.

Bibliografi

[1] IEEE Standards Association. Standard for wireless lan medium access control (mac) and physical layer (phy) specifications. Standard, IEEE, 2007.

[2] Paramvir Bahl and Venkata N. Padmanabhan. Radar: An in-building rf-based user location and tracking system. *Proc. IEEE Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'00)*, pages 775–784, Mar 2000.

[3] M. Baldauf, S. Dustdar, and F. Rosenberg. A survey on context-aware systems. *Int. J. Ad Hoc and Ubiquitous Computing*, 2:263–277, 2007.

[4] Mauro Brunato and Csaba Kiss Kalló. Transparent location fingerprinting for wireless services. In *in Proceedings of Med-Hoc-Net 2002*, 2002.

[5] Guanling Chen and David Kotz. A survey of context-aware mobile computing research. 2004.

[6] C. H. Edwards and David E. Penney. *Elementary Linear Algebra*. Prentice Hall, 1988.

[7] Shih-Hau Fang, Tsung-Nan Lin, and Po-Chiang Lin. Location fingerprinting in a decorrelated space. *IEEE Transactions on Knowledge and Data Engineering*, 20:685–691, 2008.

[8] Emory Fry and Leslie Lenert. Mascal: Rfid tracking of patients, staff and equipment to enhance hospital response to mass casualty events. *AMIA Annu Symp Proc.*, page 261–265, 2005.

[9] Brian Hall. *Beej's Guide to Network Programming*. Jorgensen Publishing, 2009.

[10] Jeffrey Hightower and Gaetano Borriello. Location systems for ubiquitous computing. *Computer*, 34(8):57–66, 2001.

[11] Cisco Systems Inc. Cisco aironet antennas and accessories. Technical report, Cisco Systems, 2009.

[12] Todd D. Jick. Mixing qualitative and quantitative methods: Triangulation in action. *Administrative Science Quarterly*, 24(4):602–611, 1979.

[13] Kamol Kaemarungsi. Efficient design of indoor positioning systems based on location fingerprinting. 2005.

[14] Kamol Kaemarungsi and Prashant Krishnamurthy. Modeling of indoor positioning systems based on location fingerprinting. 2004.

[15] Kamol Kaemarungsi and Prashant Krishnamurthy. Properties of indoor received signal strength for wlan location fingerprinting. *Mobile and Ubiquitous Systems, Annual International Conference on*, 0:14–23, 2004.

[16] Masakatsu Kourogi, Nobuchika Sakata, Takashi Okuma, and Takeshi Kurata. Indoor/outdoor pedestrian navigation with an embedded gps/rfid/self-contained sensor system. *Technical report of IEICE. PRMU*, 106(73):75–80, 2006.

[17] David Ley. Ubiquitous computing. *Emerging technologies*, 2:64–79, 2007.

[18] Binghao Li, Yufei Wang, Hyung Keun Lee, Andrew Dempster, and Chris Rizos. A new method for yielding a database of location abstract fingerprints in wlan, 2005.

[19] Reto Meier. *Professional Android Application Development.* Wrox Wiley, 2009.

[20] Hirokazu Miura, Kazuhiko Hirano, Noriyuki Matsuda1, Hirokazu Taki, Norihiro Abe, and Satoshi Hori. Indoor localization for mobile node based on rssi. *Knowledge-Based Intelligent Information and Engineering Systems*, 4694:1065–1072, 2007.

[21] Sindre Paulsrud Moe. Design and evaluation of a user-centric information system. Master's thesis, Norwegian University of Sience and Technology, 2009.

[22] William S. Murphy and Willy Hereman. Determination of a position. Technical report, 1999.

[23] Truls Ostbye, David F. Lobach, Dianne Cheesborough, Ann Marie M. Lee, Katrina M. Krause, Vic Hasselblad, and Darryl Bright. Evaluation of an infrared/radiofrequency equipment-tracking system in a tertiary care hospital. *J. Med. Syst.*, 27(4):367–380, 2003.

[24] K. Pahlavan, Xinrong Li, and J.P. Makela. Indoor geolocation science and technology. *Communications Magazine, IEEE*, 40(2):112–118, Feb 2002.

[25] P. Prasithsangaree, P. Krishnamurthy, and P. Chrysanthis. On indoor position location with wireless lans. *Personal, Indoor and Mobile Radio Communications, 2002. The 13th IEEE International Symposium on*, 2:720–724, Sept 2002.

[26] Nissanka B. Priyantha, Anit Chakraborty, and Hari Balakrishnan. The cricket location-support system. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 32–43, New York, NY, USA, 2000. ACM.

[27] Theodore S. Rappaport. *Wireless Communications: Principles and Practice. Second Edition.* Prentice Hall PTR, 2002.

[28] John C. Stein. Indoor radio wlan performance part ii: Range performance in a dense office environment. *Intersil Corporation.*

[29] Andrew S. Tanenbaum. *Computer Networks.* Prentice Hall PTR, 2003.

[30] F. van Diggelen. Indoor gps theory & implementation. *Position Location and Navigation Symposium, 2002 IEEE*, pages 240–247, 2002.

[31] Angela M. Wicks, John K. Visich, and Suhong Li. Radio frequency identification applications in hospital environments. *Hospital Topics*, 84(3):3–9, 2006.

[32] Gerhard Wiese. Utvikling av en lokaliseringsløsning ved hjelp av trådløse nettverk. E-post for kopi: gerhardc@stud.ntnu.no, 2009.