NTNU
Det skapende universitet

Lokasjons- og kontekstbaserte tjenester fra et informasjonsforvaltningsperspektiv.

FUMBLE - Et trådløst innendørs lokasjonsbasert informasjonssystem for studenter ved NTNU Gløshaugen.

Geir Marius Gjul

Master i informatikk
Oppgaven levert: Mai 2006
Hovedveileder: Ingeborg Torvik Sølvberg, IDI

Norges teknisk-naturvitenskapelige universitet
Institutt for datateknikk og informasjonsvitenskap

Forord

Denne oppgaven er skrevet i forbindelse med min masteroppgave, ved Institutt for Datateknikk og Informasjonsvitenskap ved NTNU.

Oppgaven har vært en lang, og tidvis svært krevende prosess, men interessant nok for "å holde hjulene igang". Dette spesielt fordi jeg ofte ser nye muligheter, og ikke problemer. Derfor er prototypen slik den er implementert en forenklet løsning på et ferdig system, selv om jeg gjerne skulle sett at den var implementert med mer funksjonalitet.

Jeg vil spesielt takke min veilederen, professor Ingeborg T. Sølvberg, for kyndig veiledning, og ikke minst: For å ha holdt meg på rett kurs. Det siste har ikke vært enkelt da oppgaven spenner over flere fagområder, og det til tider har vært vanskelig å se veien videre.

En stor takk går til mine foreldre, spesielt min far for korrekturlesing av oppgaven.

En takk går også til min medstudent Bjørn Egil Haugvik for samarbeidet gjennom et helt skoleår.

Til slutt vil jeg benytte anledningen til å takke Kristin Tyldum Kjøglum, Mona Knarvik og Simen Nordskog som tok seg tid til å teste prototypen, og for å ha gitt meg uvurderlig tilbakemelding.

Trondheim, 1. Juni 2006

Geir Marius Gjul

Sammendrag

Denne oppgaven presenterer en felles struktur for kontekstavhengige systemer som relaterer lokasjon og informasjon. Strukturen baserer seg på AROUND-arkitekturen, men har blitt tilpasset slik at den støtter et trådløst posisjoneringssystem som benytter "Fingerprinting" som strategi.

Resultatene beviser at posisjonering basert på trådløse signaler er mulig ved NTNU Gløshaugen, og at det er mulig å utvikle en felles struktur som knytter lokasjon, beskrevet ved signalstyrke målinger, til informasjon. Strukturen utnytter hvordan trådløse signaler blir lagret for å forenkle prosessen med å velge ut kontekstavhengig informasjon som er relevant.

Det ble utviklet en prototype som implementerer strukturen som er foreslått. Prototypen ble implementert i to deler: en lokaliseringsmodul og en informasjonsmodul. Lokaliseringsmodulen er basert på Nibble, et innendørs posisjoneringssystem for mobile enheter, som gjør posisjonering mulig vha. det trådløse nettet ved NTNU Gløshaugen. Informasjonsmodulen er utviklet for å støtte kontekstavhengig informasjon basert på lokasjonen foreslått av lokaliseringsmodulen.

Oppgaven diskuterer også hvordan kontekstavhengig informasjon kan presenteres basert på tilstand. Dette gjelder integrering av eksterne informasjonssystemer i et kontekstavhengig informasjonssystem.

Hovedkonklusjonen er at:

- Prototypen beviser at det er mulig å utvikle en felles struktur som utnytter en trådløs lokaliseringsmodul for å utvikle et kontekstavhengig informasjonssystem.
- Integreringen med andre informasjonssystem vil være avgjørende for et kontekstavhengig informasjonssystem.

Videre arbeid med dette prosjektet inkluderer å videreutvikle både lokasjonsdelen og informasjonsdelen. Lokasjonsdelen med tanke på å forbedre nøyaktigheten i posisjoneringsalgoritmen. Informasjonsdelen med tanke på å integrere eksisterende informasjonssystemer. Det er også en mulighet å implementere en Gazetteer og å implementere en kartmodul. Til slutt vil det være viktig å lage og identifisere lover som vil gjelde for et kontekstavhengig system av denne karakter.

Abstract

This thesis presents a generic structure for context-aware systems that relate location and information. The structure is based on the AROUND architecture, but has been modified such that it supports wireless positioning systems.

The results prove that positioning based on wireless signals is possible at NTNU Gløshaugen, and that it is possible to develop a general structure that relates location, described by signal strength measurements, to information. The structure exploits how wireless signal is stored to simplify the process of chosing which content dependent information is relevant.

A prototype was developed that implements the suggested structure. The prototype was implemented in two modules: a location module and an information module. The location module is based on Nibble, an indoor location system for mobile devices, that enables positioning by utilizing the wireless network at NTNU Gløshaugen. The information module is developed to support context-aware information based on the location suggested by the location module.

The thesis also discusses how context-aware information can be presented based on state. This applies to the integration of external information systems in a context-aware information system.

The main conclusion is that:

- The prototype proves that it is possible to develop a generic structure that utilizes a wireless location module to develop a context-aware information system.
- The integration between the context-aware system and external systems is essential for the success of such systems.

Future directions for this project include continuing the development of both the location module and the information module. The location module should be improved to increase accuracy in the positioning algorithm. The information module should be integrated with existing information systems. There is also the posibillity of integrating a Gazetteer and implementing a map module. Finally, it will be important to develop and identify the laws that will apply in a context-aware system of this nature.

Innhold

Figurer

Tabeller

1 Innledning

Denne oppgaven ser nærmere på hvordan informasjon kan knyttes til geografisk informasjon, og hvordan denne relasjonen kan utnyttes i et kontekstavhengig (context-aware) system. Informasjon som skal knyttes til en geografisk lokasjon (det være seg et punkt eller et område) har mange fellestrekk da denne (informasjonen) eksisterer i et kunstig område som avspeiler det fysiske området. En prototype er utviklet med tanke på hvordan en best kan utnytte denne relasjonen, og implementert med tanke på å teste ut noe av funksjonaliteten til et kontekstavhengig system. I tillegg kan prototypen fungere som en plattform for andre, da problemområdene ved et slikt system spenner over mange interessante fagområder.

1.1 Problembeskrivelse

Vi befinner oss i informasjonsalderen, og mengden informasjon som er tilgjengelig på et vilkårlig tidspunkt er enorm. Å finne tilbake til informasjon kan også være en vanskelig oppgave, spesielt hvis man har lite tid tilgjengelig. Og ser vi på hva som tilbys av trådløse enheter, blir det ikke enklere å navigere i informasjonsrommet. Å finne informasjon, eller å finne tilbake til informasjon, er en lang prosess. En må vite hvor, eller hvilket system man skal bruke, hva slags søkeord, frase og syntaks som kan gi ønsket resultat og vurdere resultatene som kanskje kan gi den informasjonen som var ønsket. Hva om det fantes et system som kunne presentere informasjon før du har tenkt på at du trenger informasjon, eller presenterer mulige startpunkt i søkeprosessen? Å benytte lokasjon for å avgrense eller tilby informasjon er en interessant tanke. For å kunne utnytte lokasjon kommersielt mangler ennå mange modeller og mye forskning. Dette gjelder spesielt innendørs da de mest utbredte trådløse teknologiene kun virker utendørs. Et slikt system vil kunne gi svar på hva som finnes i omgivelsene, og\eller hva slags kontekst brukeren befinner seg i.

Det er spådd at konkurransen blant leverandørene av mobile tjenester vil øke betraktelig. Derfor er det viktig å kunne designe, utvikle og implementere kontekstavhengige systemer enkelt og raskt. Slike systemer er ofte omtalt som allestedsnærværende systemer (ubiquitous computing) i litteraturen, og grunnen til at slike systemer er spådd en lovende framtid, er at de kan være med på å skape økt verdi (added value) for slutt-brukeren.

Ved NTNU Gløshaugen finnes det ikke et kontekstavhengig system, men behovet, modellene[1] og teknologien er tilstede. Gjennom samtaler med nye og gamle studenter, viser det seg at mange ønsker seg, og ville ha benyttet seg av, et kontekstavhengig system dersom det hadde eksistert. Begrunnelsen for dette er at informasjonsbehovet ved oppstart av et nytt studie er stort, men til tider vanskelig å tilfredsstille. I tillegg til dette er det avgjørende hvor du befinner deg og hvor

[1] Modell for lokasjonssystemer og informasjonssystemer finnes, oppgaven prøver å kombinere disse.

mye tid du har tilgjengelig. Det finnes flere eksempler (se avsnitt 2) som anvender forskjellige modeller for å innføre kontekstavhengige tjenester.

Kontekstavhengige systemer er avhengig av teknologi som kan varsle om en endring i kontekst. Ofte krever dette spesiallaget utstyr eller tilleggsutstyr. "Trådløs Trondheim" er et prosjekt under utvikling. Prosjektet er ledet av Professor Arne Sølvberg. Målet er å gjøre deler av Trondheim, i første omgang Midtbyen, campusområdene (Gløshaugen og Dragvoll) og traseen fra byen via Gløshaugen til Dragvoll trådløst innen studiestart høsten 2006, og dermed gjøre byen til et laboratorium for forskning og utvikling av mobile datatjenester"[3]. Dette vil være teknologi som kan benyttes i et kontekstavhengig system, uten å måtte installere tilleggsutstyr.

1.1.1 Mål for oppgaven

Oppgaven består av flere delmål:

1. Er posisjonering ved hjelp av trådløse signaler[2] mulig å få til ved NTNU Gløshaugen?
2. Er det mulig å skape en relasjon mellom romlig informasjon (spatial information) og lokasjon som utnytter sammenhengen mellom informasjonsrommene (information spaces) og de fysiske rommene, og i tillegg utnytter måten lokasjonsdata er organisert på?
3. Kan et kontekstavhengig system være til hjelp for studenter i studiehverdagen?

1.1.2 Oppgaveformulering

Oppgaven tar utgangspunkt i NTNU Gløshaugen og identifiserer problemer ved et kontekstavhengig system. Oppgaven kan deles i to hovedproblemer, heretter beskrevet som lokasjonsdel og informasjonsdel. Fokus i denne oppgaven vil være informasjonsdelen, men lokasjonsdelen er likevel helt nødvendig i et kontekstavhengig system[18]. Oppgaven vil derfor også gå igjennom tema som knyttes til lokalisering.

Det har vært gjennomført flere prosjekter når det gjelder innendørs posisjonering (se kapittel 2), men få beskriver sammenhengen mellom posisjon og informasjon. Geografiske informasjonssystemer (GIS) kobler geografisk referert informasjon til lokasjon, men disse fungerer bare utendørs. Oppgaven har mange likheter med tradisjonelle GIS-systemer og den har brukt ideer fra slike systemer, men det er ikke her snakk om et slikt system.

[2]Med trådløse signaler menes her alle typer trådløse signaler som blir benyttet i innendørs datakommunikasjon.

Lokasjonsdelen er den delen av prototypen som tar seg av posisjonering. Mye arbeid er tidligere blitt gjort for å få innendørs posisjonering så presis som mulig. Likevel er dette et området som mangler analytiske modeller som kan brukes som et rammeverk for å designe og utvikle posisjoneringssystemer[17]. Oppgaven foreslår en informasjonsmodell som kan være med på å støtte kontekstavhengige systemer.

Informasjonsdelen ser nærmere på hva slags informasjon som kan være aktuell, hvordan man best kan lagre slik informasjon, integrering mot andre informasjonssystemer og hvordan man kan utnytte den relasjonen som finnes mellom lokasjon og informasjon. Romlig informasjon har spesielle egenskaper som gjør det mulig å knytte slik informasjon til en fysisk lokasjon. Et kontekstavhengig system prøver å modellere virkeligheten slik at overgangen fra den fysiske verden til et informasjonsområde blir enklest mulig.

Oppgavens hovedproblemstilling kan oppsummeres slik:

Er det mulig å utvikle et innendørs kontekstavhengig system ved hjelp av eksisterende teknologi og modeller som kan hjelpe studenter ved NTNU Gløshaugen i studiehverdagen?

1.2 Scenario

For å gi et lite innblikk hva kontekstavhengige systemer er, oppsummeres dette i en fiktiv historie om to personer, Mona og Kari. Noen av funksjonene er implementert, andre er kan tenkes å implementeres i et ferdig system.

Det er mandag og Kari blir vekt av alarmen på mobilen klokka 08.00. Forelesningen hennes denne dagen begynner ikke før 10.15, så hun unner seg en lang frokost. En time senere er hun vel fremme på Gløshaugen, og får en melding på mobilen at forelesningen er flyttet til auditorium "Kjel1" fordi administrasjonen hadde gjort en feil. Siden Kari ikke vet hvor dette er ber hun systemet vise henne et kart hvor det aktuelle rommet er avmerket. Da det enda er en time til forelesning, så hun bestemmer seg for å ringe Mona, men hun svarer ikke. Kanskje fordi hun ikke hører telefonen tenker Kari, og søker henne opp på mobilen. Mona befinner seg på Sito og vil gjerne snakke med noen, så Kari bestemmer seg for å gå dit for å slå av en prat. Under samtalen kommer det frem at Mona ikke har fått med seg at forelesningen er flyttet da hun enda ikke har meldt seg på distribusjonslistene for kurset.

Etter forelesning får Kari en ny melding at det deles ut studentpakker på Stripa, så både Kari og Mona skynder seg over dit for å sikre seg en pose. Etter å ha sett igjennom posen og funnet ut at den var akkurat lik som fjorårets, går venninnene hver til sitt. Kari finner seg en leseplass, men før hun begynner ber hun systemet om at hun ikke vil forstyrres, og dette slår samtidig av lyden på mobilen.

Mona finner seg et ledig rom med trådløsdekning, og begynner å jobbe på bær-

baren. Etter en liten stund kommer det opp en melding om at det er informasjon tilgjengelig som kan være interessant. Det viser seg at det avholdes kurs i presentasjonsteknikk neste uke. Siden Mona snart er ferdig med sin mastergrad, bestemmer hun seg for at dette er noe som er verdt å være med på, da hun snart skal presentere oppgaven sin. Programmet viser også en oversikt over nye meldinger på Innsida, og innboksen hennes. Ved noen raske klikke er både meldingene og e-postene lest, uten at hun måtte logge seg inn i hvert enkel system - dette tar programmet seg av.

Etter skoletid forlater Kari Gløshaugen, og mobilen skrur på lyden og varsler om at hun har nye meldinger. Hun lar meldingene ligge og bestemmer seg for å se på de senere, da ingen av meldingene var merket som viktige.

Mona avslutter dagen med å legge inn en hyggelig melding til Kari, som Kari vil få når hun kommer på skolen neste dag.

1.3 Oppgavens struktur

Oppgaven er delt opp i følgende kapittel:

Kapittel 2 Går igjennom State of the Art. Her diskuteres tidligere arbeider innenfor lokasjonssystemer og informasjonssystemer, generelle strukturer og kontekstavhengige systemer.

Kapittel 3 Presenterer "Mitt system", slik det ideelt kunne ha blitt implementert. Dette kapitlet inneholder en kravspesifikasjon og et overordnet systembeskrivelse.

Kapittel 4 Beskriver hvordan prototypen ble implementert.

Kapittel 5 Dette kapitlet diskuterer hvordan prototypen har blitt testet, og presenterer resultatene fra disse.

Kapittel 6 Til slutt følger en oppsummering av oppgaven.

2 State of the art

Dette kapittelet inneholder relevant teori for å utvikle et innendørs kontekstavhengig systemet basert på trådløse signaler. Et kontekstavhengig system er i [17] definert som:

> Location-aware services are based on some form of positioning techniques. Positioning systems enable context aware computing with location awareness.

Oversatt betyr dette at lokasjonsavhengige systemer er basert på en eller annen form for posisjoneringsteknikk. Posisjoneringssystemer muliggjør kontekstavhengig databehandling ved hjelp av posisjonsoppmerksomhet.

Avsnitt 2.1 går igjennom generelle systemer som kan brukes i et kontekstavhengig system. Først diskuteres systemer som kan reagere på en endring i lokasjon og de mest brukte måtene dette blir gjort på (avsnitt 2.1.1). Så diskuteres noen systemer som gjør informasjon tilgjengelig for brukere (avsnitt 2.1.2). Avsnitt 2.2 inneholder viktige strukturer som kan brukes i lokasjons- og informasjonssystemer. Relasjonen mellom disse strukturene er beskrevet i avsnitt 2.3.6 og i kapittel 3. Til slutt i dette kapittelet ser oppgaven nærmere på eksisterende systemer innenfor lokasjons- og informasjonsbaserte systemer:

- Avsnitt 2.3.1 - Nibble
- Avsnitt 2.3.2 - RADAR
- Avsnitt 2.3.3 - GUIDE
- Avsnitt 2.3.4 - Wireless Campus LBS
- Avsnitt 2.3.5 - Grimstad Museum
- Avsnitt 2.3.6 - AROUND

Avsnitt 2.5 oppsummerer kapittelet.

2.1 Generelle systemer

Et kontekstavhengig system er som nevnt en kombinasjon av to enkeltstående systemer. For det første er et kontekstavhengig system basert på et system som kan reagere på en endring i omgivelsene (kontekst), i dette tilfellet en endring i lokasjon. Men det er også mulig å reagere på andre kriterier som feks. tilstand (se avsnitt 3.6.3). For det andre kan et kontekstavhengig system bestå av et, eller flere, informasjonssystem. Kombineres disse to systemene får man et kontekstavhengig system, som i teorien kan gi økt verdi for brukerne[24, 8, 18, 19].

2.1.1 Lokasjonssystemer

Lokalisering ved hjelp av radiosignaler er ikke noe nytt. Utendørs finnes allerede systemer som GPS[33] og snart europeiske Galileo[32]. I tillegg til dette er det i dag mulig å triangulere GSM signaler for å lokalisere mobiltelefoner. Men også systemer som virker innendørs har begynt å komme i den senere tid. Disse systemene baserer seg på teknologier som: RFID[36], Bluetooth[30], WiFi[38] eller ustandardiserte teknologier som RadioEye[10].

Trådløse nett kan altså brukes til mer enn bare dataoverføring. All kommunikasjon via radiosignaler er avhengig av et signal som er sterkt nok til å kunne tolkes. De fleste produsenter legger derfor inn mulighet til å lese av hvor god signalstyrken er der du befinner deg. Det er denne funksjonen som kan utnyttes i et innendørs posisjoneringssystem.

Det finnes i dag tre forskjellige metoder for å posisjonere mobile enheter: Angel of arrival (Triangulering), Trilateration (Trilaterasjon), og Fingerprinting (Fingeravtrykk). Felles for innendørs posisjonering er at de fleste systemer benytter fingerprinting som strategi. De neste punktene (2.1.1, 2.1.1) og 2.1.1 diskuterer fordeler og ulemper ved de forskjellige strategiene.

Angle of Arrival

AoA er en teknikk for å bestemme posisjonen til en mobil enhet ved hjelp av retningen til trådløse signaler. Dette blir gjort ved at et signal fra en mobil enhet registreres ved forskjellige basestasjoner. Disse basestasjonene må være utstyrt med tilleggsutstyr for å kunne beregne vinkelen til det innkommende signalet. Ved å kalkulere retningen ved minst to basestasjoner gir dette et krysningspunkt som kan beskrives med lengdegrad\breddegrad.

For at AoA skal kunne beregnes kreves at tidspunktet når signaler sendes og mottas registreres nøyaktig, og at alle enheter benytter samme tid. Fordelen med AoA er at det kan tas i bruk uten kjennskap til signalstyrken i området, da det er kun når signalet oppfattes som teller. En annen fordel er at det ikke kreves “line of sight” til basestasjonene. Ulempen er at det kreves ekstra utstyr ved basestasjonene, og det er nødvendig å kjenne avstanden mellom minst to basestasjoner (som er med i beregningen) for å kunne finne en posisjon.

AoA er den teknikken som blir benyttet i lokalisering av GSM telefoner, feks. ved 911\113 anrop.

Selv om denne teknikken er utbredt når det gjelder utendørs posisjonering, er det vanskelig å implementere en lignende strategi innendørs ved hjelp av WiFi. Hovedgrunnen til dette er at det kreves ekstra utstyr ved alle basestasjonene, og avstanden mellom basestasjonene må være kjent for at strategien skal fungere. Siden avstand mellom basestasjonene i et innendørs miljø sjelden er kjent, er ikke denne teknikken egnet for innendørs posisjonering.

Trilateration

Å forstå trilaterasjon kan være vanskelig, derfor går denne oppgaven igjennom 2-D trilaterasjon (to dimensjonal trilaterasjon), som er enkelt å forstå. Interesserte lesere henvises til [23] for en forklaring på 3-D trilaterasjon.

2-D trilaterasjon baserer seg på å finne minst tre punkt (basestasjoner) og avstanden til disse. Først beregnes avstanden til et punkt, og i et to dimensjonalt plan vil du da kunne befinne deg i en sirkel med avstanden som radius rundt dette punktet. Ved å bruke avstanden til et annen punkt vil det dannes to sirkler som overlapper hverandre og du kan befinne deg i krysningspunktene til disse to sirklene. Å kjenne avstanden til et tredje punkt vil dermed avgjøre hvilket av disse to krysningspunktene du befinner deg i. Trilaterasjon bruker dermed tre eller flere kjente lokasjoner og måler avstanden mellom den mobile enheten og lokasjonene. Måten avstand blir beregnet på er en detaljert prosess, men går kort fortalt ut på å måle hvor lang tid det tar for et signal å komme fram og ut i fra dette beregne avstanden. Det er dette som gjør at det kreves "line of sight" for at trilaterasjon skal fungere, da signalet må bevege seg i en rett linje for at det skal være mulig å beregne avstand.

Trilaterasjon er den teknikken som blir benyttet i lokalisering av GPS enheter.

Denne teknikken er globalt tilgjengelig allerede idag (GPS), men den virker altså kun utendørs. Dette fordi innendørs vil vegger reflektere signalet og gjøre avstandsmålingen upålitelig. Det jobbes med å få denne teknikken til å fungere innendørs, men det har ikke vært mulig å finne ut hvordan dette løses. For trilaterasjon gjelder det samme som AoA at det kreves eget utstyr, men som nevnt krever denne teknikken også "line of sight", og dermed er en slik teknikk ikke egnet innendørs.

Fingerprinting

Fingerprinting (fingeravtrykk) er en teknikk som baserer seg på å danne flere avtrykk i området lokaliseringen skal foregå, enten ved et fast rutenett eller ved interesseområder. Et avtrykk inneholder signalstyrken til alle tilgjengelig aksesspunkt ved en lokasjon. Ved å først bygge en oversikt som inneholder nok avtrykk til at interesseområdene er dekket, kan den relative posisjonen til en mobil enhet beregnes ved et senere tidspunkt. Å bygge en slik oversikt er ofte referert til som en "offline-fase", mens gjenfinningsdelen refereres til som en "online-fase".

For å illustrere fingerprinting med et enkelt eksempel: Sett at det finnes to aksesspunkt og avstanden mellom disse er 100 meter, noe som er omtrent rekkevidden til trådløse signaler innendørs. Ved å stå ved det ene aksesspunktet vil signalet fra dette være veldig godt, mens fra det aksesspunktet som er 100 meter unna vil signalet være veldig dårlig. Det er dermed mulig å konkludere med at du befinner deg i nærheten av det aksesspunktet som har best signal. I dette eksempelet er det ikke mulig å posisjonere brukeren nøyaktig, da to aksesspunkt ikke er nok for å bestemme absolutt posisjon. Eksempelet viser likevel et interessant område ved fingerprinting, og det er at det selv med få tilgjengelige aksesspunkt er mulig å

fastslå at en mobil bruker befinner seg "i dette området".

Går man videre med forrige eksempel og legger til et tredje aksesspunkt som ligger 100 meter unna begge de forrige aksesspunktene, vil de tre aksesspunktene danne en trekant. Innenfor denne trekanten er det mulig å posisjonere en mobil bruker svært nøyaktig, da signalstyrken fra aksesspunktene er unik for forskjellige posisjoner. Det er dette fingerprinting utnytter når lokasjon skal beregnes.[3]

Fingerprinting krever ikke eget utstyr, men kan benytte et hvilket som helst trådløst nettverk for posisjonering. Det kreves heller ikke "line of sight" til antennene. Ulempen er at signalstyrke (avtrykkene) må registreres i hele området der posisjonering skal sies å være mulig.

Fingerprinting er den teknikken som er brukt i Wireless Campus LBS[18], RADAR[24], Nibble[21] og til en viss grad GUIDE[8] (se også avsnitt 2.3).

Denne teknikken kan sies å kun virke innendørs, da utstyret er kun ment for innendørs bruk, og at teknologien (ihvertfall ennå) kun er installert innendørs. Det kan likevel være mulig å legge inn lokasjoner som befinner seg utendørs fordi trådløse signaler går igjennom vegger, og derfor vil signalstyrke kunne registreres på utsiden av bygg som har trådløs dekning.

2.1.2 Informasjonssystemer

Informasjonssystemer omhandler mennesker og maskiner, og metoder for å samle inn, bearbeide, overføre og spre informasjon. De fleste informasjonssystemer i dag er tilgjengelig via web. Dette vil si at brukere kan få tilgang til systemene via en nettleser og en datamaskin med internettilkobling. Før var det vanlig å ha en dedikert applikasjon for hvert informasjonssystem, noe som krevde at du måtte ha en spesiell programvare for å få tilgang til systemet.

Selv om Internett har gjort det enklere å få tilgang til informasjonssystemer, er problemet det samme som da informasjonssystemene krevde egen programvare. Informasjonen finnes (eller kan finnes) i flere systemer, og en bruker må ofte sjekke flere systemer for informasjon. Resultatet er at en bruker er nødt til å bruke flere systemer for å få med seg alle endringer. En student ved NTNU må feks., i tillegg til å logge seg inn i skolens datasystem, logge seg inn i Innsida, It's:Learning, og webmail for å få med seg alle meldinger. I tillegg kommer meldinger fra Guru-[4] og Orakel-tjenesten[5], Tekna[6], institutt spesifikke sider[7] osv.

Systemene har ulike formål og struktur, og dette fører til at brukerne må lære seg

[3] Eksemplene gitt her er svært enkle, og representerer ikke virkeligheten. I virkeligheten må man ta hensyn til støy og refleksjoner av signaler. Aksesspunkt har ikke en fast avstand seg imellom, osv., men prinsippet er det samme.

[4] `http://guru.idi.ntnu.no`

[5] `http://www.orkel.ntnu.no`

[6] `http://www.ntnu.tekna.no`

[7] `http://www.idi.ntnu.no`, `http://www.ime.ntnu.no`

hvordan de forskjellige systemene virker. Dette kan være et hinder, spesielt for nye brukere. I tillegg til å vite hvordan systemene skal brukes, må brukerne ha kjennskap til hvor (url), hvilket system som brukes til hva og når man bør sjekke for oppdateringer.

For å gi noen eksempler på informasjonssystemer, går denne oppgaven igjennom It's:Learning og Innsida. Begge er systemer som er i bruk ved NTNU, men benyttes også av andre universiteter. Disse systemene dekker de fleste aspekter ved et informasjonssystem. Oppgaven går senere igjennom hvordan disse systemene kan integreres (avsnitt 3.6.3) og i tillegg hvordan webmail kan integreres i et kontekstavhengig system (avsnitt 3.6.3).

It's:Learning

It's:Learning[8] er NTNU's E-læringssystem. Dette vil si at det er her studenter får informasjon om fag, timeplaner, forelesninger, gruppearbeid, prosjekter osv. It's:Learning er i sin tredje versjon, som er en stor forbedring fra forrige versjon. Spesielt er det lagt vekt på å forbedre brukergrensesnittet. Likevel var det noen studenter som mente at den nye versjonen var mindre oversiktlig enn den forrige. Dette er ikke en ideell situasjon da det kan hindre studenter i å bruke It's:Learning.

It's:Learning er et fora som lar forelesere holde kontakten med studentene og omvendt. Studenter som melder seg opp i et fag vil få et eget område under "Emner" som de blir knyttet til. Her kan de finne oversikt over kurs, forelesninger osv. Kort fortalt et område der det samles informasjon om faget.

Men It's:Learning tilbyr også at studenter kan kommunisere med hverandre. Studenter som jobber i felles grupper kan opprette egne områder som kun gruppemedlemmene har tilgang til. Dette gjør det enkelt å spre informasjon, endre, og samarbeide om oppgaver selv om studentene befinner seg på forskjellige lokasjoner. Det finnes også mulighet for "live-chat", der studenter kan snakke sammen via meldinger som øyeblikkelig dukker opp på mottakerens skjerm. Det er også mulig å legge igjen meldinger, både private og til hele kurs, en tjeneste som er ganske redundant da E-mail som oftest blir brukt til dette.

It's:Learning tilbyr også filoverføring og versjonskontroll. Dette gjør det enkelt for studenter å samarbeide, og holde oversikten over nye filer, evt. se hvilke filer som er endret siden sist (markeres med rødt). Brukerne har også tilgang til et privat område der det er mulig å legge inn notater, oppdatere en personlig kalender og lagre filer. I It's:Learning kan man også velge å bli varslet når nye meldinger blir lagt inn, både via E-mail og SMS[9].

Et problem med It's:Learning er at det ikke finnes en adresse som brukerne kan skrive inn for å få tilgang. For å benytte It's:Learning må man først logge seg inn i Innsida (se avsnitt 2.1.2). Her finnes det en link som tar brukeren videre til systemet.

[8] `http://www.its-learning.com`

[9] SMS koster 1 krone pr. mottatte melding.

Se også avsnitt 3.6.3.

Innsida

Innsida[10] er skolens interne meldingssystem. Her legges det inn meldinger til studenter og ansatte, og meldinger til hele NTNU. Det kan være meldinger om kurs, ledige stillinger, foredrag, åpningstider osv. I tillegg til dette finnes det masse informasjon knyttet til det å være student eller ansatt ved NTNU.

Et eksempel på en melding som legges ut på Innsida kan være:

```
Studentmeldinger:

Store dokumenter: Word II.
Fredag 21/10 kl. 1215-1400 i auditorium D13.
NB! Endret auditorium NB!.
Lars Meisingseth, ITEA/Orakel
```

Her er meldingen lagt ut i god tid, men meldingen har blitt oppdatert med et nytt auditorium like før kurset skulle holdes. Studenter som har lest den første meldingen kan overse oppdateringen, og dermed møte opp i feil auditorium. Dette er et typisk eksempel på at informasjon ikke alltid når frem til studentene.

Problemet er ikke å få tilgang til informasjonen, men å få informasjonen til rett tid. Kurs, møter, foredrag osv. har ofte en kort tidsramme, og for å få med seg denne må man sjekke Innsida ofte. Mange studenter gjør ikke dette, og kan derfor gå glipp av endret informasjon.

Se også avsnitt 3.6.3.

2.2 Viktige strukturer

Et kontekstavhengig informasjonssystem inneholder store mengder data. I tillegg er det flere forskjellige typer data som skal beskrives for at brukerne skal ha noen nytte av systemet. For at både brukere og systemet skal kunne gjenfinne data, er det nødvendig å kunne strukturere dataene. Et viktig begrep er datastrukturen "metadata", se avsnitt 2.2.3. Metadata kan brukes til å beskrive alle former for data, men måten det er gjort på varierer. Oppgaven har valgt å bruke metadata sammen med en hierarkisk struktur, som er en system-struktur (avsnitt 2.2.1) og gazetteers (avsnitt 2.2.4). En hierarkisk struktur er foreslått i AROUND-arkitekturen[16], mens gazetteers er valgt fordi dette kan være med på å beskrive geografiske lokasjoner og informasjon knyttet til disse.

[10] `https://innsida.ntnu.no`

2.2.1 Hierarkisk organisert data

I en hierarkisk data modell (hierarchical data model) er data organisert i en tre struktur på en slik måte at den ikke kan ha for mange relasjoner. Attributter er knyttet til en instans av en post, og alle postene inneholder de samme attributtene. Denne strukturen tillater repeterende informasjon ved å bruke foreldre\barn-relasjoner (parent\child relations). Instanser er relatert til hverandre ved å bruke 1:N (leses: en til mange) relasjoner. En slik struktur har ofte en instans på toppen av treet, kalt en rotnode.

Et eksempel på hierarkisk organisert data er: En organisasjon har mange ansatte. Organisasjonen vil da organisere de ansatte i en tabell (instans) "Ansatte". Tabellen vil inneholde kolonner (attributter) for fornavn, etternavn, telefon, adresse osv. En organisasjon kan ha flere ansatte (forelder), mens en ansatt kan kun tilhøre en organisasjon (barn), og det er disse to delene som lager et hierarki. I dette eksemplet er det organisasjonen som er rotnoden.

Fordelen med å lage hierarkiske relasjoner mellom forskjellige typer data er at det kan gjøre det enkelt å besvare noen spørsmål. Ulempen er at det kan bli svært vanskelig å svare på andre. I tillegg må man overholde en til mange relasjonen, hvis ikke blir hierarkiet et nettverk.

2.2.2 Tre-lags modellen

"Tre lags modellen" (Three-tier eller Three-layer) er ofte brukt i systemutvikling som en metode for å skille brukergrensesnittet, forretnings logikk og datalagring, hhv. presentasjonslag, logikklag og datalag (derav navnet tre-lags modellen). Disse er ofte implementert som selvstendige moduler, og derfor også utviklet og vedlikeholdt uavhengig av de andre lagene. Modellen er en klient-server arkitektur, der de forskjellige lagene kan kjøre på forskjellige plattformer. Tre-lags modellen er ansett som å være en software arkitektur (software architecture) og et software design mønster (software design pattern).

Bortsett fra de vanlige fordelene med modulær software, med godt definerte grensesnitt, er intensjonen med tre-lags modellen at det skal være mulig å oppgradere eller bytte ut lagene uavhengig av hverandre. Velger man å bytte database, skal dette kun påvirke datalaget. Noen mener at det å dele opp programmer på denne måten er en ulempe, da en enkel databasespørring fra grensesnittet må igjennom logikklaget for å kunne kommunisere med datalaget. Det er mulig å spørre direkte fra presentasjonslaget, og dermed slippe ekstra kode i både logikklag og databaselag.

I enkelte tilfeller velger man å dele opp logikklaget i flere lag. Dette kalles en multilag arkitektur (n-tier architecture).

2.2.3 Metadata

Metadata er ofte beskrevet som "data om data", som er en dårlig beskrivelse av hva mentadata egentlig er. Metadata er data som beskriver et annet sett av data feks. kolonnenavn i en tabell, eller spesielle tagg-er som beskriver teksten den omrammer (som feks. XML-dokumenter). En annen viktig side ved metadata er sammenhengen, eller relasjonen, mellom data. Et eksempel på dette er relasjonsdatabasen, som bygger på relasjonsmodellen[11]. Denne sammenhengen beskriver hvordan data står i forhold til hverandre, feks. i et hierarki.

[4] definerer metadata som:

> Metadata is information on the organization of the data, the various data domains, and the relationshipt between them. (...) For instance, in a database management system, the schema specifies some of the metadata, namely, the name of the relations, the fields or attributes of each relation, the domain of each attribute, etc.

Oversatt betyr dette at metadata er informasjon om organisering av data, forskjellige data domener, og relasjonen mellom de. I et database system spesifiserer feks. skjemaet noe av metadataene, det vil si navn på relasjoner, felt eller attributter ved hver relasjon, domenet for hver attributt osv.

Metadata blir brukt for å [12]:

- *Forbedre tilgang til informasjonsobjekter:* Søk kan bli forbedret gjennom bruk av metadata, og for å kunne søke i forskjellige samlinger.
- *Lagre kontekst om objekter:* Bibliotek og museum inneholder ikke bare objekter, men prøver også å lagre relasjonen mellom de, relasjonen til andre mennesker, plasser, bevegelser og hendelser.
- *Utvide bruksområdene til objekter:* Avstand og\eller plassering kan gjøre det vanskelig for mennesker spredt over et stort område å få tilgang til et fysisk objekt. Metadata prøver å fjerne denne barrieren ved å beskrive objektet på en slik måte at det ikke er nødvendig være i besittelse av dette. Dette kan realiseres gjennom tekst, bilder, video, audio, 3D-representasjon osv.
- *Håndtere versjoner:* Over tid vil det eksistere flere versjoner av informasjonsobjektene, og metadata blir brukt til å relatere versjonene til hverandre, men samtidig ta vare på versjonene.
- *Ta vare på rettslige spørsmål:* Bruk av objekter kan være pålagt retningslinjer, som feks. ved kopiering. Eierskap er også en viktig attributt som kan endre seg over tid.

[11] `http://en.wikipedia.org/wiki/Relational_databases`

- *Konservere:* Metadata er utviklet for å overleve endringer i hardware, software og nye informasjonssystemer.
- *Forbedre systemer og økonomi:* Metadata kan bli brukt for å evaluere og teste systemer, enten for å forbedre gamle systemer eller utvikle nye.

En vanlig form for metadata er assosiert med tekst der metadata beskriver forfatter, tittel, antall sider, dato for utgivelse osv. Kartoteket du finner i bibliotek er et konkret eksempel på metadata. Katalogkortene inneholder informasjon om boken og hvor den befinner seg i biblioteket.

2.2.4 Gazetteers

En Gazetteer er en geografisk ordbok, og en viktig kilde til informasjon om steder. En slik ordbok inneholder hvordan land, regioner eller kontinenter er oppbygd, sosial statistikk og fysiske kjennetegn, som feks. fjell, elver og veier.

Verden Gazetteers (world gazetteers) inneholder ofte en alfabetisk liste over land, med statistikk tilknyttet hvert enkelt. Andre gazetteers tar for seg informasjon om enkelte byer, landsbygder eller bosetninger.

Mindre gazetteers (short-form gazetteers) er ofte brukt i digitale relasjoner eller GIS-systemer, og kan bare inneholde en liste av navn sammen med lengde- og breddegrad eller andre referansesystem.

En generell Gazetteer beskriver et sted ved navn, lokasjon og steds-type. Et navn er et geografisk navn, feks. “Oslo”. Lokasjon kan representeres på forskjellige måter, feks. ved hjelp av GPS koordinater eller X og Y koordinater på et kart. Steds-type sier noe om stedes natur, feks. “by”. Dette er de tre kjerneelementene som må være med for å danne en Gazetteer.

Gazetteers kan tilpasses og utvides ut i fra hvilke behov man har. [25] har tatt utgangspunkt i ADL Gazetteer GCS[28], og tilpasset denne for kunne søke i et digitalt billedgalleri.

2.2.5 Bayesian Network

Nibble (se avsnitt 2.3.1) bygger et “Bayesian network” for å kunne dedusere posisjon. Dette er et alternativ til en hierarkisk struktur nevnt i avsnitt 2.2.1.

The Interchange Format for Bayesian Networks[12] oppsummerer et “Bayesian network” med:

> The goal of the current effort is to specify a XML-based format that is very simple to understand and yet can represent directed acyclic

[12] `http://www.cs.cmu.edu/afs/cs/user/fgcozman/www/Research/InterchangeFormat/`

graphs with probabilistic relations, decision variables and utility values.

Oversatt betyr dette at et mål for nåværende arbeid er å spesifisere et XML-basert format som er svært enkelt å forstå og som likevel kan representere rettede asykliske grafer med sannsynlige relasjoner, slutnings variable og tilleggs verdier.

Å gi et konkret eksempel på dette er vanskelig, da selv med få lokasjoner generer mye data. For en oversikt over strukturen lister Figur 1 formatet. Nibble lagrer lokasjoner i et nettverk med navnet "WaveLan Location System", som kan best forklares med en slags rotnode. Under denne defineres lokasjonene med navn, adresse og signalstyrke som spesifisert i bif.dtd (Document Type Definition). Dette skaper sannsynlige relasjoner (probalistic relations) og skal gjøre Nibble i stand til å dedusere posisjon utover hva man kan forvente av et system som sammenligner signalstyrke parvis med lagrede lokasjoner.

```
<!ELEMENT bif ( network )*>
      <!ATTLIST bif version CDATA #REQUIRED>
<!ELEMENT network ( name, ( property | variable | definition )* )>
<!ELEMENT name (#PCDATA)>
<!ELEMENT variable ( name, ( outcome | property )* ) >
      <!ATTLIST variable type (nature|decision|utility) "nature">
<!ELEMENT outcome (#PCDATA)>
<!ELEMENT definition ( for | given | table | property )* >
<!ELEMENT for (#PCDATA)>
<!ELEMENT given (#PCDATA)>
<!ELEMENT table (#PCDATA)>
<!ELEMENT property (#PCDATA)>
```

Figur 1: bif.dtd

2.3 Kontekstavhengige systemer

Med kontekstavhengige systemer forsøker man å integrere digitale enheter i miljøet personer beveger seg i. Et av målene er å gjøre enheter i stand til å registrere endringer i miljøet rundt dem og automatisk tilpasse seg og reagere ut i fra endringene basert på bruker behov og preferanser.

Systemene som er nevnt under baserer seg på en eller flere av teknologiene og strukturene diskutert i dette kapittelet. Felles for disse er at programmene reagerer på forandring i omgivelsene, hovedsaklig en endring av posisjon.

2.3.1 Nibble

Nibble er et system som gjør det mulig å posisjonere trådløse klienter innendørs. Prototypen er utviklet av Multimedia Systems Laboratory ved Department of Computer Science, UCLA[21], og bygger på MUSE[6] som er en mellomvarearkitektur for sensorbaserte systemer. Nibble er et selvstendig program som ved hjelp av IEEE 802.11 standarden deduserer posisjonen til en trådløs klient ut i fra tilgjengelige trådløse signaler. I teorien kan systemet støtte andre trådløse teknologier, men dette er enda ikke implementert.

Nibble benytter seg av standard IEEE 802.11 trådløst utstyr. Den eneste begrensningen i hardware er at klienten må benytte et Lucent Orinoco (eller WaveLAN) IEEE 802.11 trådløst kort. Dette fordi Nibble krever at Lucent Client Manager kjører på klienten som skal posisjoneres.

Arbeidet med Nibble argumenterer for at tidligere systemer kun har benyttet seg av naboskapsalgoritmer (nearest-neighbor techniques). Måten Nibble deduserer posisjonen til trådløse klienter på, er ved hjelp av probabilistic modeling based on Bayesian networks". Dette er raskt gjennomgått tidligere i avsnitt 2.2.5.

Ved å la en av forskerne alltid ta med seg en bærbar pc hver gang han beveget seg bort fra kontoret sitt, ble det generert data som dannet grunnlaget for resultatene. Disse viste at Nibble alltid klarte å skille mellom konferanserommene i bygningen, men tok feil enkelte ganger når det kom til mindre kontorer.

Det konkluderes med at Nibble er nøyaktig helt ned til ca. 3 meter (10 feet).

Arbeidet med Nibble nevner flere områder der det er mulig for forbedringer, og de fleste går på å øke presisjonen ved posisjonering. Profilering går ut på at enheter eller brukere har større sannsynlighet for å befinne seg på faste plasser, og dette kan brukes til å forbedre resultatene. Filtrering kan være med på å hjelpe ved plutselige endringer i signalet ved å ta i bruk for eksempel et Kalmanfilter[34] for å jevne ut signalet. Interesserte lesere henvises til [15]. Naboskapsmatrise (adjacency matrix) kan være med på å utelukke områder som ikke er tilgjengelig fra en bestemt posisjon, altså rom det er umulig å forflytte seg til fra det rommet brukeren befinner seg i. Rent teknisk har UCLA Multimedia Systems utviklet et kort som måler temperatur, luftfuktighet, retning og akselerasjon, dette i den tro at det er mulig å forbedre presisjonen. Det jobbes også med å kunne støtte heterogene nettverk, som feks. Bluetooth og WiFi, men nåværende versjon støtter altså kun IEEE 802.11 standarden.

Arbeidet presenterer et fungerende system som benytter seg av trådløse nettverk for å kunne posisjonere mobile enheter. Arbeidet går igjennom flere aspekter ved lokasjonsbaserte tjenester, og beviser at lokalisering vha. WiFi i et innendørs miljø er mulig. Det foreslås ingen modell for hvordan posisjon kan relateres til informasjon eller tjenester.

2.3.2 RADAR

RADAR[24] ble presenter under "Conference on Computer Communication, Proceedings IEEE INFOCOM 2000" som et eksperiment om hvordan tilby lokasjonsavhengige tjenester vha. trådløst utstyr. Arbeidet viser at det ved hjelp av trådløse aksesspunkt og en bærbar pc med et trådløst nettverkskort, er mulig å lokalisere brukere innendørs. Den tar for seg forskjellige algoritmer for beregning av posisjon, samt flere måter å lagre denne informasjonen på. Det vises også til flere typer funksjonalitet knyttet til et kontekstavhengig informasjonssystem, som feks. at utskrifter blir skrevet ut på nærmeste skriver.

RADAR er utviklet for å kunne kjøre på WaveLAN trådløse nettverkskort, men uavhengig av hva slags aksesspunkt som er implementert. En empirisk metode[31] ble brukt for å posisjonere en bruker. Kort fortalt går dette ut på å samle inn data på forhånd slik at det finnes et sett med punkter som senere brukes til å posisjonere en bruker. Denne metoden ble testet opp mot kun sterkeste signal og random posisjonering. For lagring av posisjonsdata diskuterer arbeidet flere metoder og algoritmer (R-Tree, X-Tree osv.), men valgt modell er en implementering av en lineær søkealgoritme. "Active Directory" er foreslått som bindeleddet mellom lokasjon og tilgjengelige tjenester, men ikke diskutert. Interesserte lesere henvises til [2].

Rapporten viser til gode resultater når det gjelder å posisjonere brukere innendørs. I gjennomsnitt hadde systemet en feilmargin på 2,94 meter, som er tilsvarende det resultatet rapportert i 2.3.1, og med de forbedringer som artikkelen foreslår, er det teoretisk mulig å få enda bedre presisjon. Andre interessante resultater i rapporten er at retning spiller en avgjørende rolle når det gjelder posisjonering, og at støy ikke er en brukbar funksjon når det gjelder posisjon.

For å forbedre presisjonen til systemet undersøkes bruker-mobilitets profiler (user-mobility profiles). Gjennom sannsynlighet og historie kan slike data være med på å støtte lokaliseringsprosessen.

Rapporten presenterer et system for å lokalisere brukere innendørs. Ved å benytte seg av algoritmer, datastrukturer og modeller kan systemet lokalisere brukere ned til ca. 3 meter, noe som er nok for å plassere en mobil bruker i feks. et kontor. Relasjonen mellom lokasjon og informasjon bygges vha. "Active Directory", også utviklet av Microsoft. Teknologien prøver å finne riktig og nok informasjon for brukere i store systemer, men er vanskelig å implementere fordi den krever mye kunnskap om, ikke bare selve teknologien, men også om miljøet den skal implementeres i.

2.3.3 Guide

The Guide Project[8] hadde som mål å utvikle og realisere et kontekstavhengig turistsystem. Målet ble oppnådd ved at en prototype ble tatt i bruk i byen Lancaster, England. Systemet lar en turist som er i besittelse av en Guide enhet (unit)

ta virtuelle reiser gjennom byen. Det er verdt å merke seg at her er det ikke snakk om et system som til enhver tid vet hvor brukeren befinner seg, men som vet om en bruker befinner seg i et gitt område (celle) og kan gi instrukser om hvordan komme seg til neste område (celle).

Guide benytter seg av celler som en fastsettelse for lokasjon. En celle inneholder informasjon om omgivelsene, og en bruker befinner seg bare i en celle av gangen. En celle er av størrelsesorden en sirkel med radius 100 meter, og en bruker kan befinne seg hvor som helst i denne sirkelen.

Valget av celler gjør det enkelt å lage en informasjonsmodell. Guide har identifisert fire distinkte typer informasjon: 1; Informasjon som er skreddersydd gjeldene celle 2; Geografisk informasjon 3; Hypertekst og 4; Profil informasjon (omtalt som "active components" i arbeidet).

The GUIDE Project kan vise til et system som fungerer i et reelt miljø, der turister kan be om å få låne en Guide enhet. Systemet ble godt mottatt av turistene, og tilbakemelding fra disse førte til forbedringer, spesielt med tanke på brukergrensesnitt.

Videre arbeid handler om å utvide bruksområdene for Guide til å omfatte undervisningsverktøy (educational tools) og spill. Forfatterne ser for seg at Guide feks. kan brukes til skattejakt. De nevner også at Bluetooth kan benyttes som teknologi for å kunne støtte innendørs kommunikasjon og bedre presisjon.

Det er spesielt informasjonsmodellen utviklet for Guide som er av relevans for denne oppgaven. Tilgjengelig informasjon presenteres i form av lenker, og lenker tilgjengelig endres etter hvilken celle brukeren befinner seg i. Dette er aktuelt da det på NTNU Gløshaugen allerede eksisterer store mengder informasjon i form av webbaserte tjenester.

2.3.4 Wireless Campus LBS

Wireless Campus LBS (Location Based Services)[18] er et prosjekt ved University of Twente, i Nederland. Prosjektet har undertittelen: "Building campus-wide location based services based on WiFi technology". Universitetet er et relativt lite universitet, med ca. 6000 studenter og 2500 ansatte (i 2004). Spredt utover det 140 hektar store universitetsområdet er det installert 650 individuelle trådløse aksesspunkt, som gjør dette til et av Europas største homogene trådløse nettverk. Dette betyr at alle med en PC, bærbar PC, PDA eller andre WiFi kompatible enheter har tilgang til universitetsnettverket og Internett, uten bruk av kabler.

Prosjektet utnytter det trådløse nettverket for å kunne tilby lokasjonsbaserte tjenester. Det trådløse nettverket er bygd i samarbeid med IBM Nederland og Cisco Systems. Det er det trådløse nettverket som står for både posisjonering og datakommunikasjon.

Ved hjelp av posisjon, håper prosjektet å kunne bygge flere tjenester som er po-

sisjonsavhengig. Eksempelet de bruker er "Friend Finder" (FF), som er en prototype utviklet ved samme universitet som gjør det mulig for personer å finne hverandre i to utvalgte bygninger. Målet er å utvide dette til å gjelde hele universitetet.

Målet med dette prosjektet er ikke å utvikle et trådløst lokasjonsbasert system, men å fungere som en sandkasse (testbed) for studenter og andre interesserte. Målet har derimot vært å finne ut hva slags infrastruktur som kreves for kontekstavhengige tjenester og hva slags tjenester som kan legges oppå et slikt system. Prosjektet kan ikke vise til noen resultater enda, siden det nettopp har startet, men med bakgrunn i "Friend Finder", som hadde gode resultater, skal det være mulig å utvikle et system som dekker hele universitetet.

Videre arbeid innebærer å få dannet et digitalt kart over universitetet, og å få avmerket de trådløse aksesspunktene på kartet. Dette for å kunne utvikle flere metoder for posisjonering. Prosjektet er også i dialog med flere forskjellige forskergrupper for å se om de kan bidra til den tekniske infrastrukturen som da kan tjene som en sandkasse for deres spesifikke fagfelt. Et mål er at systemet skal bli en dagligdags funksjon for mobile brukere ved universitetet.

Prosjektet presenterer flere gode ideer både når det gjelder lokalisering vha. av WiFi og hva dette kan brukes til. Samarbeidet mellom de forskjellige fagfeltene ved universitetet åpner for nye ideer og muligheter, som ikke er så lett å se ut i fra kun et fagfelt.

2.3.5 Grimstad Museum

"Anvendelse av metodikken Contextual Design for utvikling av lokasjonsbaserte tjenester ved Grimstad Museum"[22] er et prosjekt gjennomført ved Høgskolen i Agder. Prosjektet går ut på å utvikle en elektronisk museumsguide ved Grimstad Bymuseum ved å anvendte metoden "Contextual Design". Det ble utviklet en prototype som ble testet ved Ibsenhuset og Grimstad Bymuseum. Prosjektet baserer seg på CyberGuide[19] som er forgjengeren til GUIDE (avsnitt 2.3.3) presentert tidligere i denne oppgaven.

Oppgaven går grundig igjennom de mest aktuelle trådløse teknologiene som Blåtann (bluetooth), WLAN (Wireless local area network) og IR (Infrared). Prototypen benytter seg av Blåtann for kommunikasjon og posisjonering. I tillegg diskuterer den fordeler og ulemper med forskjellige typer PDA-er. Metodikken "Contextual Design" er en framgangsmåte som studerer brukerne i deres arbeidskontekst for å samle inn konkrete og reelle data.

Resultatet av arbeidet ble en fungerende prototype som dekket tre rom: Apoteket, Vaktrommet og Terje Vigenrommet. Gjester ved museet kunne fritt vandre mellom disse rommene og få presentert informasjon avhengig av hvilket rom de befant seg i. Det viste seg at Blåtann var en egnet teknologi for bruk i lokasjonsbaserte tjenester, og metoden "Contextual Design"var en god fremgangsmåte.

I videre arbeid er det spesielt et punkt denne oppgaven kan ha nytte av:

> Det må legges inn mekanismer slik at PDA'ene automatisk kobler seg opp til det sterkeste Blåtann aksesspunkt som finnes. Dette er for å støtte opp rundt lokasjonsavhengig informasjon. Det er naturlig at aksesspunktet i rommet som gjesten befinner seg gir fra seg sterkeste signalstyrke.[13]

I tillegg nevner oppgaven at informasjon bør legges inn i en database for bedre struktur, og at brukerne bør varsles når ny informasjon er tilgjengelig. Også brukergrensesnittet er gjenstand for videre arbeid, da størrelsen på skjermene til PDA-er er en utfordring.

Det er flere sider ved denne oppgaven som er relevant for denne oppgaven. Dette gjelder spesielt diskusjonen om tilgjengelige trådløse teknologier, og begrunnelsen for valg av Blåtann. I tillegg peker oppgaven på flere problemområder ved en elektronisk museumsguide, som kan overføres til et kontekstavhengig system. Dette går både på hvordan posisjonering er mulig, og hvordan informasjon bør lagres og knyttes til posisjon.

2.3.6 AROUND

"The AROUND architecture for dynamic location-based services"[16] foreslår to modeller for å knytte digitale tjenester til lokasjon, hvor den ene blir realisert gjennom to tilfellestudier (case studies).

Artikkelen har tatt utgangspunkt i GUIDE (avsnitt 2.3.3), men skiller seg fra dette prosjektet ved å benytte seg av en abstrakt modell som beskriver forholdet mellom en tjeneste og en lokasjon. Det ble utviklet to modeller: avstands-basert utvelgelse (distant-based selection) og omfangs-basert utvelgelse (scope-based selection). Den første ble foreslått, men ikke testet til fordel for den siste.

Resultatene viser at modellen fungerer, men åpner også for nye spørsmål. Siden det er snakk om en arkitektur (AROUND architecture) er det noen av disse spørsmålene som ikke går direkte på modellen, men rammeverket.

Noen få punkter er nevnt som videre arbeid, men det er spesielt et som er interessant. Måten områder (omtalt som scope) blir definert på, må løses på en fornuftigere måte. Det påpekes at flere administratorer kan ha tilgang til systemet og ha forskjellig oppfatning av hvordan slike områder skal utformes. Løsningen er å lage et intuitiv grensesnitt, slik at flere områder ikke overlapper hverandre, men at man heller gjenbruker det første området som ble laget.

Arbeidet presenterer en modell som kan knytte tjenester og\eller informasjon til lokasjon. Selv om modellen støtter innendørs lokasjonsavhengige tjenester er

[13] Det er en skrivefeil i teksten: "... som gjesten befinner seg *i* gir fra seg sterkeste signalstyrke."

det ikke foreslått hvordan posisjonering skal foregå innendørs. Systemet som er foreslått baserer seg på GSM og GPS, som begge fungerer dårlig innendørs. Da andre har foreslått løsninger for innendørs posisjonering, er det modellen som er interessant i denne oppgaven, og ikke posisjoneringsteknologien.

2.3.7 Oppsummering kontekstavhengige systemer

Kontekstavhengige systemer er ikke noe nytt. Det finnes allerede systemer som bruker GSM signaler for å lokalisere mobiltelefonbrukere. Dette kan brukes til å sende menyen til en restaurant hvis brukeren befinner seg i nærheten av denne. Andre systemer tilbyr sporing, feks. av personer, gjenstander eller flåtestyring innenfor feks. drosjenæringen eller spedisjonsfirma.

Systemene som er nevnt under avsnitt 2.3 har to viktige fellestrekk. For det første baserer de lokasjonsdelen på WiFi (Grimstad Museum brukte Blåtann, men teknikken er den samme). Men lokasjon alene er ikke nyttig hvis den ikke relateres til annen informasjon. Feks. "Du er her." sier ingenting hvis du ikke beskriver hvor "her" er i forhold til et kjent punkt. Det andre fellestrekket er at systemene prøver å relatere informasjon til posisjon (Nibble er feks. kun et posisjoneringssystem). Dette gjelder spesielt Guide og AROUND. Som det går frem av teksten oppstår det her et skille mellom systemer som 1) er gode posisjoneringssystemer og 2) er gode informasjonssystemer. Eksempel på dette er Nibble, som er et godt posisjoneringssystem, men som mangler en informasjonsmodell knyttet til lokasjonsdataene, og AROUND som har en god informasjonsmodell, men som overlater posisjoneringen til et subsystem (som man antar finnes).

Merk at det finnes flere prosjekter som bruker trådløse signaler i et kontekstavhengig system. Noen av disse er referert til i teksten, men ikke gjennomgått i dette kapitelet. Det er flere grunner til at disse ikke har blitt diskutert her. Hovedgrunnen er at disse systemene har benyttet tilleggsutstyr for å posisjonere mobile enheter. Denne oppgaven avgrenser seg til teknologi og utstyr som allerede er tilgjengelig. Andre grunner er at teknikken som har blitt brukt ikke kan benyttes med trådløse signaler innendørs, som feks. triangulering og trilaterasjon. Noen eksempler på prosjekter som ikke har blitt diskutert i dette kapittelet er: MUSE[6], CyberGuide[19], Cordis[26], VOR[20], FAMOUS[14] og "Sensor-Assisted WiFi indoor location system"[7]. I tillegg finnes det flere prosjekter som er under utvikling mens denne oppgaven ble skrevet, feks. Trådløs Trondheim.

2.4 Andre forhold

I tillegg til de strukturer, modeller og systemer som er nevnt til nå, er det andre forhold som er viktig å være klar over når det gjelder et kontekstavhengig system. Disse er listet under.

2.4.1 Lokasjons-sporing og posisjon-oppmerksomhet

(Location-tracking and position-aware)

I [27] diskuteres det at posisjoneringsteknologi kommer i to varianter: Lokasjons-sporing og posisjons-oppmerksomhet. Lokasjonssporingsteknologi foregår ved at en ekstern enhet (feks. nettverket) beregner posisjonen. RADAR benytter seg av denne metoden, da det er opp til aksesspunktene å avgjøre hvor en mobil klient befinner seg[24]. Posisjonsoppmerksomhetsteknologi gjør klienten i stand til selv å finne ut hvor den er (feks. GPS). Denne teknologien er benyttet i Nibble[21], GUIDE[8] og Wireless Campus [18].

Det spiller ingen rolle for hvilken løsning som blir valgt med tanke på presisjon eller informasjonsmodell. Forskjellen ligger i hvordan systemet blir implementert.

2.4.2 Service baserte systemer

Service baserte systemer tilbyr tjenester (services) gjennom en enkel portal. Tjenester defineres her som tjenester som vanligvis tilbys via flere systemer, men som vha. service baserte systemer kan nåes via et felles grensesnitt.

Det som er spesielt for denne oppgaven er at den definerer at en tjeneste er både informasjon (å kunne tilby informasjon) og tjenester (å kunne tilby tjenester). Service baserte systemer legger også vekt på å identifisere hva slags tjenester som skal inngå i systemet, for så å lage en abstrakt kobling mellom tjeneste og lokasjon. Denne koblingen skal helst gjenspeile virkeligheten, på en slik måte at overgangen fra et virkelig rom til et kunstig rom blir så enkel som mulig.

2.4.3 Screen scraping

Screen scraping[37] er en teknikk for å hente ut data fra skjermvisningen til et annet program. Forskjellen mellom tradisjonell datagjenfinning og screen scraping er at screen scraping henter ut informasjon som i utgangspunktet var beregnet på mennesker, i motsetning til datagjenfinning der informasjon er organisert slik at det er raskt og enkelt for andre programmer å hente ut informasjon (og kan dermed være nesten uleselig for mennesker).

Det finnes flere definisjoner på screen scraping, men i denne oppgaven er det HTML-scraping som er relevant. HTML-scraping parser en HTML-side, analyserer innholdet og tar avgjørelser basert på innhold eller endringer som ble funnet. Det er også mulig å velge ut noe av innholdet for å presentere dette for brukeren på en annen måte enn siden det ble hentet fra. HTML inneholder ofte mye informasjon som ikke blir presentert for brukeren, men som er avgjørende for hvordan informasjon blir presentert. Et eksempel er tag-en "<TITLE>": Denne gjør at teksten blir presentert i toppen av vinduet. At den skal plasseres her, er ikke viktig å vite for brukeren, men denne korte teksten inneholder ofte stikkord om hva

som befinner seg på siden. Dermed kan den danne grunnlaget for et effektiv søk (i motsetning til å måtte søke igjennom hele siden).

2.5 Oppsummering state of the art

Dette kapitelet prøver å gi en oversikt hva som skal til for å lage et kontekstavhengig system. Først ble generelle lokasjonssystemer og informasjonssystemer diskutert. Så ble datastrukturer som kan brukes i kontekstavhengige system diskutert, og til slutt ble det diskutert noen kontekstavhengige system.

Studiet av disse systemene danner grunnlaget for "Mitt system" (Kapittel 3).

3 Mitt system

Denne oppgaven foreslår å kombinere kontekstoppmerksomme systemer med tradisjonelle informasjonssystemer for så å utvikle et kontekstavhengig informasjonssystem vha. eksisterende teknologier og metoder. Her er et kontekstoppmerksomt system definert som et system som reagerer på endring i posisjon, mens et informasjonssystem er et system som tilbyr informasjon eller tjenester, ofte via web. Eksisterende teknologier og metoder omfatter utstyr, programvare og modeller som er fritt (gratis) tilgjengelige. Målet er å kombinere funksjonaliteten til et lokasjonssystem med funksjonaliteten til et, eller flere, informasjonssystem for å skape økt funksjonalitet.

Oppgaven prøver å besvare følgende spørsmål 1.1.1:

- Er posisjonering ved hjelp av trådløse signaler mulig ved NTNU Gløshaugen?
- Er det mulig å skape en enkel relasjon mellom lokasjonsdata og informasjonsdata, og i tillegg utnytte måten lokasjonsdata er organisert?
- Kan et kontekstavhengig system være til hjelp for studenter i studiehverdagen?

3.1 Det ideelle systemet

Systemet oppgaven foreslår inneholder en lokaliseringsmodul, en informasjonsmodell og en informasjonsmodul. Lokaliseringmodulen har som oppgave å fortelle informasjonsmodulen hvor brukeren av systemet befinner seg. Informasjonsmodulen bruker denne lokasjonen og informasjonsmodellen for å tilby kontekstavhengig informasjon til brukeren.

Det ideelle systemet inneholder lokaliseringsdata for hele skolens område, som gjør systemet i stand til å lokalisere en bruker hvis denne befinner seg på skolens område. Systemet vil være integrert med dagens datasystemer ved universitetet, og vil kunne tilpasse hvordan informasjon blir presentert basert på kontekst og brukerens ønsker.

For å benytte seg av systemet må brukeren installere en klient som kommuniserer med en server. Klienten har som oppgave å bestemme posisjon og videresende denne til serveren. Serveren har som oppgave å finne kontekstavhengig informasjon og sende denne tilbake til klienten. Kontekstavhengig informasjon er informasjon om sted (som er lagret i systemet) og informasjon hentet fra eksterne systemer (dvs. systemer som er integrert (se avsnitt 3.6.3)).

Oppgaven til et slikt system er å raskt kunne holde brukerne oppdatert basert på konteksten de befinner seg i.

3.2 Brukere av systemet

Systemet slik det er foreslått i denne oppgaven retter seg mot studenter. Studenter blir introdusert for flere systemer som inneholder forskjellig informasjon om studier, fag, fritidsaktiviteter, konserter, kurs, osv. I tillegg er nye omgivelser, rutiner, andre studenter, forelesere og universitetet kilder til informasjon.

Ved å forsøke å samle mest mulig informasjon på et sted, og samtidig tilpasse denne slik brukeren ønsker, prøver et kontekstavhengig system å imøtekomme informasjonsbehovet som studenter kan ha. Studentene kan ha behov for å vite hvor en spesiell forelesning er, eller når støttekurset i matte går.

Brukere av systemet er derfor *studenter*. Det er disse som vil benytte seg av systemet til daglig. Målet er å kunne tilby brukerrelevant informasjon. Dette er både informasjon som kan være relevant for studenten, basert på hvilket studie han\hun tar, og informasjon som er basert på hvilken kontekst studenten befinner seg i. Studentene vil også kunne gi viktig tilbakemelding om mulige forbedringer, problemer og foreslå ny funksjonalitet. Ved å gi slik tilbakemelding kan studentene ved å logge seg inn i systemet være sikker på at de alltid er oppdatert og har tilgang til informasjon som er knyttet til den kontekst de befinner seg i.

I tillegg vil systemet bli brukt av *forskere* for å utvikle ny funksjonalitet i et kontekstavhengig system. Forskere er også interessert i resultatene av systemet, og vil bruke dette i sine valg for videreutvikling.

For at et slikt system skal fungere er det også nødvendig med personer som har administrativ tilgang til dette, for å kunne oppdatere informasjon og tjenester. Dette er ikke en konkret bruker, men kan i fremtiden bli en ansvarlig enhet. Dette er nødvendig for drift av systemet. Det kan også tenkes at noen studenter påtar seg administrative roller, feks. ved å kunne oppdatere informasjon som eksisterer i systemet, men disse rollene er ikke diskutert i denne oppgaven.

3.3 Hva skal systemet tilby

Et kontekstavhengig system vil i utgangspunktet begrense seg til hvordan det definerer en endring i kontekst. Dette systemet definerer kontekst som en endring i lokasjon. En endring i lokasjon gjør det mulig å tilpasse informasjon og tjenester som er tilgjengelig basert på lokasjonen. Det er også nødvendig å ta hensyn til en endring i tilstand, for å kunne oppdage endringer i eksterne systemer, og dette er diskutert i avsnitt 3.6.3.

Det kan knyttes mye data til en lokasjon, og samtidig kan lokasjon brukes til å skape ny informasjon. En lokasjon kan beskrives med metadata, se avsnitt 2.2.3. Denne beskrivelsen kan inneholde navn på sted, alternative navn, dekningsområde, historikk osv. Gazetteer er en måte å strukturere slik informasjon på, se avsnitt 2.2.4. En Gazetteer inneholder attributter som kan sies å være søkbare, og som kan gi brukeren informasjon om lokasjonen. Søk er en viktig funksjon i in-

formasjonssystemer, men er ekstra viktig i kontekstavhengige systemer da dette tilbyr funksjonalitet som gjør brukeren i stand til å få tak i informasjon som ikke er en del av konteksten.

Et kontekstavhengig system trenger ikke bare å tilby informasjon, men kan også gi tilgang til andre datasystemer. Ved å integrere andre datasystemer får brukeren tilgang til disse gjennom et felles grensesnitt. Et eksempel kan være at en bruker tar en utskrift og systemet henter automatisk opp skriverkøen til skriveren. Brukeren ser da hvor i køen dokumentet befinner seg, og hvis systemet tillater det, kunne endre prioritet skriverjobben.

Systemet er ikke avhengig av andre tjenester for å få tilby økt funksjonalitet. Det kan utvikles egne moduler som kan implementeres i systemet. Et eksempel kan være å tilby en tjeneste som viser hvordan en bruker kan ta seg fram ved hjelp av et kart, eller hvor andre brukere befinner seg i forhold til deg.

Under oppsummeres noen av de punktene som er viktige ved et kontekstavhengige system:

- **Lokasjon:** I sin enkleste form opplyser systemet om hvor du befinner deg. I de fleste tilfeller er dette ikke spesielt nyttig for en bruker, men det er akkurat denne informasjonen et kontekstavhengig system basert på lokasjon benytter seg av.
- **Informasjon om lokasjon:** Ved å bruke informasjon om lokasjon, kan et kontekstavhengig system gi informasjon om hva som befinner seg "her". "Her" er da definert som hvor den lokasjonen systemet til enhver tid befinner seg på. Dette kan være romnummer, navn på rom, historisk informasjon, størrelse på rom, innhold, tilgjengelig tjenester osv.
- **Informasjon om brukere:** Ved at systemet kjenner posisjonen til alle brukerne av systemet, er du mulig å spørre systemet om hvor en bestemt bruker befinner seg.
- **Veibeskrivelse:** Systemet kan også bruke lokasjon til å finne ut hvor du skal gå. Dette er nyttig for studenter som ikke er kjent ved området de skal bevege seg i.
- **Tilstand:** Et kontekstavhengig system kan reagere på andre kriterier enn en endring i lokasjon, feks. tilstand. En endring i tilstand er definert ved at det skjer en endring i innhold feks. på en webside. Slike endringer er ofte viktige, og ved å la et kontekstavhengig system overvåke websider er det mulig å varsle brukerne.

3.4 Sammendrag kravspesifikasjon

Det er identifisert seks brukerkrav ved et kontekstavhengig system. Dette avsnittet inneholder et kort sammendrag av disse. Komplett kravspesifikasjon finnes i

Appendix B.

Brukerkrav 1

Dette kravet beskriver hvordan systemet skal besvare spørsmål av typen: "Hvor er"-spørsmål. Studenten får oppgitt å møte i en forelesningssal, men vet ikke hvor denne befinner seg. Her kan systemet enten gi et kart med et referansepunkt, eller i hvilket bygg rommet befinner seg.

Systemet vil også benytte dette for å lokalisere studenten, og presentere hva som finnes i omgivelsene, se Brukerkrav 2.

Brukerkrav 2

Dette kravet beskriver hvordan systemet skal besvare spørsmål av typen: "Hva inneholder"-spørsmål. Studenten trenger å vite om et rom inneholder, eller er tilknyttet en printer, eller om et rom inneholder en projektor.

Systemet skal kjøre en slik spørring automatisk når det skjer en endring av lokasjon, og presenterer resultatene for studenten.

Brukerkrav 3

Dette kravet beskriver hvordan systemet skal besvare spørsmål av typen: "Hvor går jeg"-spørsmål. Studenten stiller systemet et spørsmål om hvordan komme til et angitt sted. Systemet vil da ta hensyn til hvor studenten befinner seg, hvor han vil, og presentere et kart der studenten og målet er avmerket. Alternativt kan systemet gi svaret i form av en veibeskrivelse.

Brukerkrav 4

Dette kravet beskriver hvordan systemet skal besvare spørsmål av typen: "Hvor befinner"-spørsmål. Studenten spør systemet om hvor en annen student befinner seg, og systemet presenterer enten et kart eller en beskrivelse.

Brukerkrav 5

Dette kravet beskriver hvordan systemet skal besvare spørsmål av typen: "Hva er"-spørsmål. Studenten spør systemet om det finnes informasjon knyttet til en lokasjon eller objekt. Systemet henter frem informasjonen og presenterer den for brukeren.

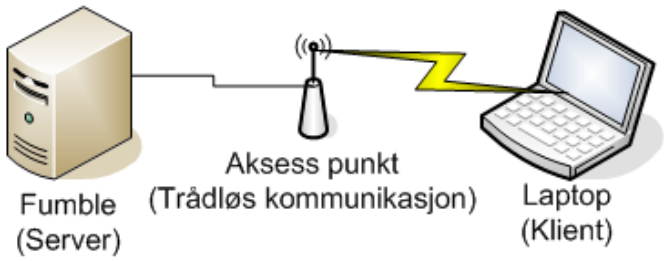
Systemet skal kjøre en en slik spørring automatisk når det skjer en endring av lokasjon, og presenterer resultatene for brukeren.

Brukerkrav 6

Dette kravet beskriver hvordan systemet reagerer på en endring i kontekst. Dette gjelder både en endring i lokasjon og tilstand. Oppdager systemet at det skjer en endring, vil systemet automatisk kjøre spørringer som beskrevet i Brukerkrav 1, 2 og 5. Måten informasjonen blir presentert på er avhengig av kontekst og brukerprofil.

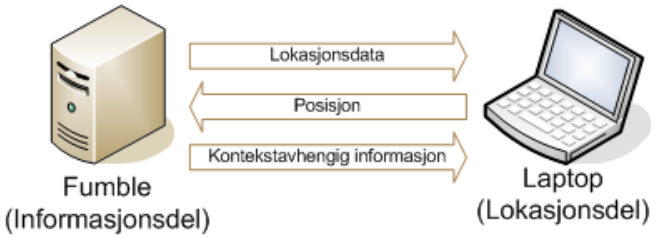
3.5 Overordnet systembeskrivelse

Systemet vil bestå av en klient-server løsning. Brukerne kommuniserer med systemet gjennom en lokal klient, som igjen kommuniserer med en sentral server. Systemet vil utnytte det trådløse nettverket ved NTNU Gløshaugen for å kunne kommunisere med serveren, uansett hvor brukeren måtte befinne seg og samtidig bruke signalstyrken til å posisjonere brukeren (se Figur 2).



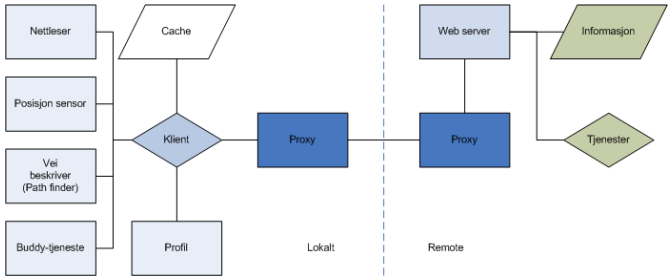
Figur 2: Server-klient løsning

Innledningsvis i dette kapitlet defineres et kontekstavhengig system som to separate moduler, og en struktur som gjør at disse er sterkt knyttet til hverandre. For å kunne illustrere dette viser Figur 3 en forenklet løsning på systemet. Lokasjonsdelen vil kjøre på klientsiden av systemet. Dette er fordi det er klienten som har tilgang til de trådløse signalene, selv om lokasjonsdata vil ligge sentralt (på serveren). Disse vil bli overført til klienten ved oppstart. Har klienten siste versjon av lokasjonsdataene vil ikke disse dataene bli overført. Informasjonsdelen vil kjøre på serversiden av systemet. Ved at klienten "sier ifra" hvor den tror den befinner seg, kan serveren bruke dette til å returnere kontekstavhengig informasjon.



Figur 3: Lokasjonsmodul og informasjonsmodul

Utfordringen blir dermed å skape en relasjon mellom lokasjonsdata og informasjonsdata. GUIDE[8] beskriver en klient-server systemstruktur, som har vist seg å fungere i virkelig miljø. Arkitekturen definerer klare oppgaver for hver enkelt komponent, samtidig som det eksisterer en godt definert grense mellom klient og server (se Figur 4). Det er derfor valgt å basere systemarkitekturen på denne. Merk at "posisjon sensor" er lokaliseringsmodulen, og "Web server" inneholder informasjonsmodulen.



Figur 4: Systemstruktur

I tillegg til systemstrukturen er det nødvendig med en datastruktur som knytter lokasjonsmodulen til informasjonsmodulen. AROUND-arkitekturen er en slik datastruktur som skisserer en struktur som knytter informasjon til forskjellige lokasjoner. AROUND-arkitekturen bygger en hierarkisk struktur (se avsnitt 2.2.1). Øverste nivå knytter informasjon til alle lokasjoner. Går man nedover i hierarkiet knyttes informasjon til færre og færre lokasjoner. Denne strukturen gjør at det enkelt kan knyttes informasjon til flere lokasjoner. Derfor er denne valgt som datastruktur.

AROUND-arkitekturen definerer to krav som må oppfylles for at en slik struktur skal kunne implementeres:

- ***Physical locality.*** *The system must provide a service selection mechanism that is effectively based on physical locality, thus allowing two physically co-located devices to discover the same services independently of their networks or administrative domains. Supporting this form of locality implies a discovery process capable of working in the wide area, over multiple network technologies, and across multiple administrative domains.*

  Oversatt betyr dette at systemet må tilby en utvelgelsesmekanisme som er basert på lokasjon og som tillater at to forskjellige enheter må kunne oppdage de samme tjenestene uavhengig av nettverk og administrative domener. For å støtte dette er det nødvendig med en lokaliseringsprosess som er i stand til å arbeide i et bredt område, over flere nettverksteknologier, og på tvers av flere administrative domener.

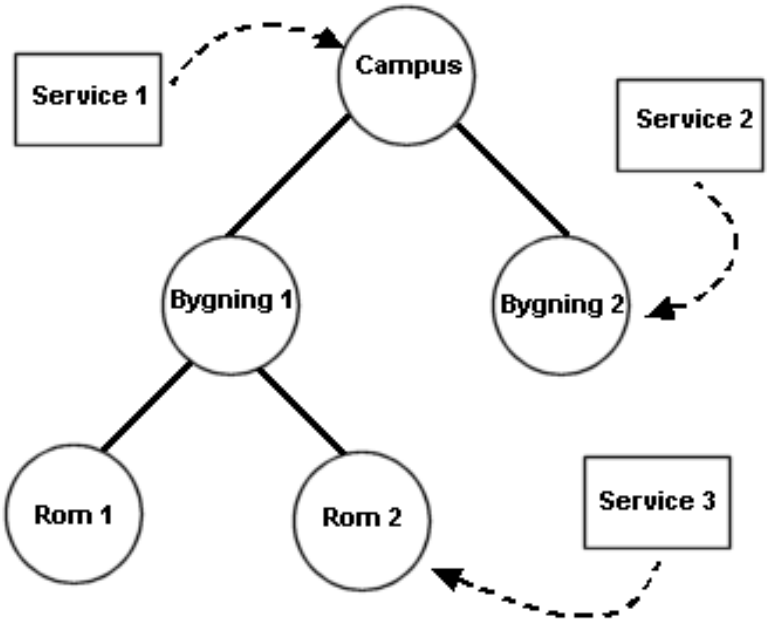
- ***Spatially distributed operation.*** *The service location space should allow queries to be scoped in spatial terms and be answered without having to search the entire offer space. Even though the system aims to be globally available, in the sense that it should be possible to discover local services anywhere, it does not aim to support global discovery, in the sense of searching for services available anywhere. This particular characteristic of locationbased services discovery should be explored in the design of a scalable global infrastructure for location-based services by creating a distribution model that reflects the spatial organisation of the service offer space.*

> Oversatt betyr dette at spørringer må kunne grupperes innenfor lokasjoner for å slippe å måtte søke igjennom all tilgjengelig informasjon. Dette selv om systemet prøver å være globalt tilgjengelig, og at det skal være mulig å oppdage lokale tjenester overalt. Systemet skal altså begrense søket ut i fra hvor det befinner seg. Dette er et særtrekk ved lokasjonsbaserte systemer som bør utforskes ved design av skalerbar global infrastruktur for lokasjonsbaserte tjenester ved å lage en distribuert modell som reflekterer den romlige organiseringen av tjenestene som tilbys.

AROUND-arkitekturen foreslår altså en struktur som gjør tjenester tilgjengelig der brukeren befinner seg. Den hierarkiske strukturen er med på å støtte "grupperte spørringer" (scope). Dette er best forklart ved et eksempel:

Uten gruppering må systemet stille spørsmålet: "Finn all informasjon som er knyttet til *lokasjon*.". Med gruppering kan systemet stille spørsmålet: "Hva slags informasjon er knyttet til *lokasjon*.". Forskjellen er at i det første spørsmålet må systemet søke igjennom all tilgjengelig informasjon, mens i det andre spørsmålet ser det kun på informasjon som er relatert til *lokasjon*.

AROUND-arkitekturen er illustrert i Figur 5.



Figur 5: AROUND-arkitekturen

3.6 Informasjonsmodell

I avsnitt 3.5 beskrives AROUND-arkitekturen. Arkitekturen beskriver hvordan det er mulig å utnytte romlig informasjon (spatial information) i et kontek-

stavhengig system. Figur 5 illustrerer modellen, men går ikke i detalj. Dette avsnittet beskriver hvordan denne oppgaven implementerer AROUND-arkitekturen slik at det er mulig å utnytte trådløse signaler.

Det er viktig å merke seg at med informasjonsmodell menes her både strukturen for lokasjons- og informasjonsdata. Der lokasjonsdelen eller informasjonsdelen beskrives, er disse beskrevet som hhv. lokasjonsdel og informasjonsdel. Informasjonsmodellen bygger på arbeidet gjort i GUIDE[8] og AROUND[16].

For å lettere se sammenhengen mellom lokasjonstruktur og informasjonstruktur beskrives først typer informasjon i et kontekstavhengig system, deretter beskrives lokasjonsdelen og informasjonsdelen hver for seg. Til slutt beskrives relasjonen mellom disse.

Informasjonsmodellen er presentert i Figur 9.

3.6.1 Design av informasjonsmodell

GUIDE[8] har identifisert fem typer informasjon:

1. **Geografisk informasjon:** Geografisk informasjon er fundamentet til et lokasjonsbasert system. Denne informasjonen kan være representert på forskjellige måter som feks. XY-koordinater eller GPS-data. Det ble imidlertid valgt å representere lokasjon vha. "fingeravtrykk" (se Appendix A). Et fingeravtrykk inneholder signalstyrken til tilgjengelige aksesspunkt, MAC-adressen til aksesspunktene, en unik identifikator og et navn. Signalstyrken er delt inn i to: Styrke og støy, og vha. av disse to kan Signal to Noise Ratio (SNR) deriveres, noe som har blitt brukt av andre systemer for lokalisering. Dette systemet benytter seg kun av signalstyrke (RSS). Det kan (bør) eksistere flere fingeravtrykk for samme posisjon. Fingeravtrykkene bør normaliseres og forskjellige fingeravtrykk kan ha forskjellig prioritet.

2. **Global informasjon:** Global informasjon er ikke knyttet opp til geografisk informasjon. Dette kan være informasjon eller tjenester som brukeren ønsker skal være tilgjengelige uavhengig hvor denne befinner seg. Et eksempel på dette kan være informasjon om studentpakker eller tjeneste "'Innsida". Knytter man derimot en bestemt lokasjon til slik informasjon, blir denne informasjonen kontekstavhengig, se punktet under.

3. **Kontekstavhengig informasjon:** Kontekstavhengig informasjon er nært knyttet opp til geografisk informasjon. Et eksempel kan være hjemmesiden til IT3000, som har forelesning i rom 123 på mandager kl. 13.15. Denne informasjonen bør alltid være tilgjengelig uansett tid eller sted, men bør knyttes opp til posisjonen til rom 123 feks. kl. 13.00. På denne måten har brukeren enkel tilgang til hjemmesiden (et klikk unna) hvis denne deltar i forelesningen.

4. **Informasjon.** Denne typen er ikke helt identifisert enda, men dreier seg om fast informasjon knyttet til et punkt. Med fast menes informasjon som sjelden endres, som bygningsinformasjon.

5. **Profiler:** Profiler er nødvendig for å kunne tilby individuell informasjon til forskjellige brukere. Feks. kan man i profilen ta vare på preferanser, tidligere bevegelser, fast arbeidsplass osv. På denne måten kan brukeren skreddersy sin kommunikasjon med systemet, og systemet kan bruke denne informasjonen til å øke verdien (ved å tilby kun bestemt informasjon og kunne øke nøyaktigheten).

Punkt en danner grunnlaget for lokasjonsdelen av modellen (se avsnitt 4.2.1), mens resten av punktene danner grunnlaget for informasjonsdelen (se avsnitt 4.3.1).

3.6.2 Lokasjonsdel

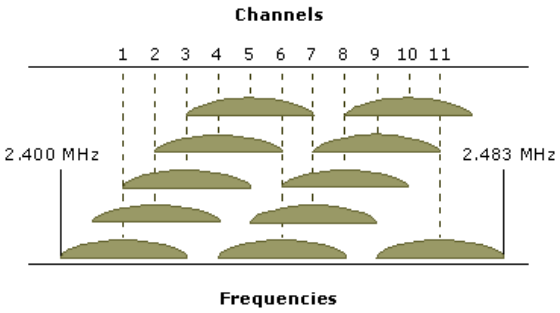
For lettere å forstå hvordan lokaliseringsmodulen fungerer er det nødvendig med bakgrunnskunnskap innenfor trådløse signaler. Dette avsnittet går kort igjennom de viktigste punktene for denne oppgaven:

Flere prosjekter[8, 18, 24] påpeker at plassering av aksesspunkt er viktig for å kunne posisjonere enheter nøyaktig. Dette gjelder ikke bare avstanden mellom, men også hvor høyt over bakken. Dette er fordi bygningsstrukturen er med på å avgjøre hvor langt et trådløst signal rekker, og en endring på få centimeter kan gi store forbedringer.

Det er ikke bare plassering som er med på å avgjøre hvor godt et trådløst nettverk vil fungere, men også valg av kanaler. I mesteparten av Europa er 11 kanaler tilgjengelig, men en av disse overlapper med nærliggende kanaler og forårsaker støy. Dette fører til at det kun er mulig å bruke tre kanaler (1, 6 og 11) uten å at kanalene skaper problemer for hverandre (se Figur 6).

Det er viktig å merke seg det er ikke gjort endringer med tanke på plassering eller valg av kanaler. Dette fordi dette er faktorer som ikke kan endres, men som er gitt fra før. Det nevnes likevel, fordi dette er faktorer som kan endres ved implementasjon for å forbedre både dataoverføring og presisjon i lokaliseringsmodulen.

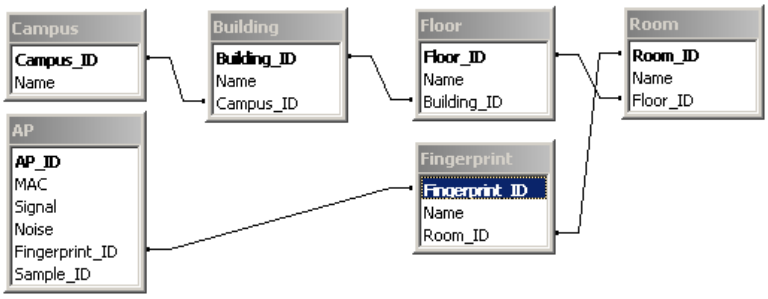
Ved IT-V finnes det 19 aksesspunkt (se Figur 1 i avsnitt 4.1.3), og med kun tre kanaler tilgjengelig er det vanskelig å ikke skape overlappende nettverk. Figur 26 (avsnitt 4.1.4) gir en oversikt over trådløse signaler ved arbeidsrom 263 ved IT-V. NTNU har valgt å benytte kanalene 1, 4 og 11, i motsetning til 1, 6 og 11 som er anbefalt. Av figur 6 ser vi at kanal 1 og 4 overlapper, og dermed skaper støy for hverandre. Hvorfor NTNU har valgt denne løsningen vites ikke på nåværende tidspunkt. I tillegg er det et aksesspunkt som bruker kanal 7, noe som kan være en feilkonfigurasjon eller at denne tilhører en annen bygning og dermed satt opp etter andre krav.



Figur 6: Mulige kanaler i Europa

Støy er en naturlig del av omgivelsene og påvirker hvor godt et trådløst signal oppfattes. I tillegg påvirker refleksjoner fra vegger, strømledninger, heiser, type trådløst utstyr osv. hvor godt et signal oppfattes. Denne oppgaven velger å se bort fra støy, og baserer seg kun på registrert signalstyrke for å dedusere posisjon. Et arbeid som tar hensyn til støy er Wireless Campus LBS[18].

Oppgaven har tidligere beskrevet AROUND-arkitekturen, men denne sier ikke noe om hvordan lokasjonene beskrives. Derfor er det nødvendig å utvide modellen. Modellen beskriver verden vha. "Campus", "Bygning" "Etasje", og "Rom", men overlater selve posisjoneringen til et underliggende subsystem. Figur 7 beskriver hvordan denne oppgaven utvider AROUND-arkitekturen for å tilpasse den et trådløst posisjoneringssystem.



Figur 7: Struktur i lokasjonsdel

AROUND beskriver modellen med "Campus", "Bygning" "Etasje", og "Rom", der rom er den miste lokasjonen som kan eksistere i modellen. Siden rom kan dekke et stort område, feks. en gang eller et auditorium, utvider denne oppgaven modellen med "Punkt" (definert som et "Fingeravtrykk"). Et punkt i denne oppgaven kan ikke sammenlignes med et punkt beskrevet ved hjelp av feks. GPS-koordinater eller X og Y koordinater i et kartsystem. Et punkt i denne oppgaven vil kunne beskrive et område, feks. en arbeidsplass. Dette fordi at trådløse signaler ikke er nøyaktige nok til å fastslå nøyaktig lokasjon. Dette er heller ikke et mål.

For å gjøre systemet så nøyaktig som mulig, ble det lagt inn støtte for at et punkt kan ha flere avtrykk. Attributten "Sample_ID" i tabellen AP (Figur 7) holder rede

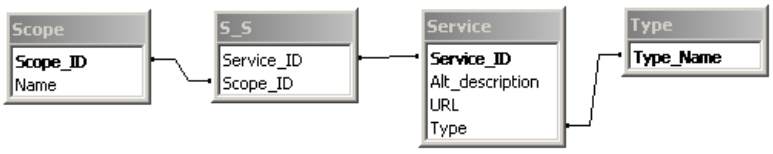
på hvilket avtrykk dette aksesspunktet tilhører, i tillegg til hvilket Fingeravtrykk (attributt Fingerpring_ID).

Ved å utvide AROUND-arkitekturen på denne måten, har systemet nå muligheten for å definere et punkt og i tillegg kunne knytte flere målinger opp mot dette punktet.

3.6.3 Informasjonsdel

Informasjonsdelen bygger på lokasjonsdelen og er blitt utformet med tanke på hva slags informasjon systemet skal forvalte, i tillegg til hvordan systemet skal integreres med andre systemer. Det er altså tre viktige kriterier som må oppfylles: 1) det må være en sammenheng mellom informasjonen og lokasjonen i modellen 2) hva slags informasjon systemet skal ta vare på, og 3) hvordan systemet skal integreres mot andre systemer.

AROUND-arkitekturen definerer informasjon som en tjeneste. Mitt system vil tilby både informasjon og tilgang til andre tjenester. Det er derfor nødvendig å utvide AROUND-arkitekturen. Figur 8 beskriver hvordan AROUND-arkitekturen utvides for å tilpasse modellen til å gjelde både tjenester og informasjon.



Figur 8: Struktur i informasjonsdel

Det er flere faktorer som har vært avgjørende for informasjonsdelen. For det første har GUIDE[8] identifisert fire typer informasjon som hører til i en informasjonsdel. For det andre har metadata vært viktig for å beskrive informasjonen. Til slutt har en generell gazetteer blitt brukt med tanke på at denne kan utvides på et senere tidspunkt (se avsnitt 2.2.4).

Figur 8 beskriver hvordan systemet beskriver informasjon, Informasjon er knyttet til en navne-type (tabell "Type") slik det er foreslått i avsnitt 3.5, samt informasjonen grupperes for å slippe å søke igjennom all tilgjengelig informasjon (tabell "Scope"). Modellen støtter dermed raske oppslag, samtidig som relasjonen mellom det fysiske rommet og informasjonsrommet er ivaretatt, se punktet under.

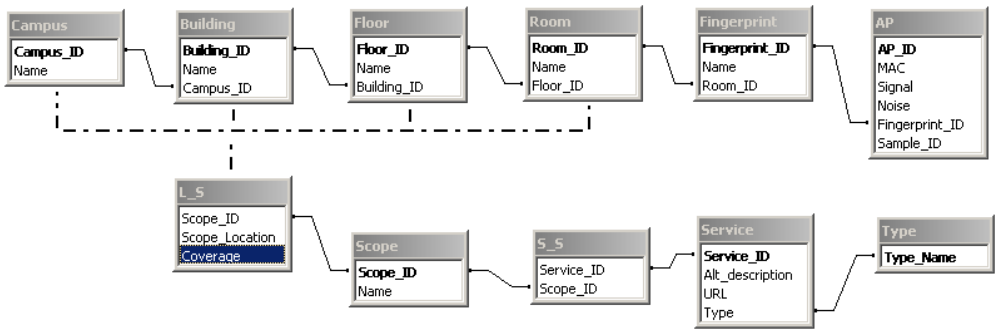
Relasjonen mellom lokasjonsdel og informasjonsdel

Et av målene med denne oppgaven er å skape en relasjon mellom lokasjon og informasjon. I et system der både lokasjon og informasjon er i konstant endring er det viktig å utvikle en modell som gjør det konseptuelt enkelt å forstå hvordan lokasjon og informasjon henger sammen[16, 22].

AROUND-arkitekturen foreslår en abstrakt relasjon mellom lokasjon og informasjon som knytter informasjon til lokasjon. I den fysiske verden er informasjon knyttet til en bestemt lokasjon, og målet er å knytte samme informasjon på en lignende måte til en eller flere informasjonsrom i systemet.

Et kontekstavhengig system vil inneholde store mengder informasjon. I tillegg vil denne informasjonen bestå av flere typer (se avsnitt 3.6.3). Som nevnt benyttes metadata for å beskrive denne informasjonen. Dette er nødvendig da informasjon kan være en adresse til en tjeneste som tilbyr informasjon, historisk informasjon, geografisk informasjon, osv.

Figur 9 foreslår en slik relasjon.



Figur 9: Informasjonsmodell

Lokasjonsdelen er her den øverste rekken med tabeller, mens informasjonsdelen er den nederste. Relasjonen er den stipla linjen som går mellom tabellen L_S (Location_Service) og tabellene "Campus", "Building", "Floor" og "Room". Tanken er her at det er attributten "Coverage" som forteller systemet hvilken lokasjon informasjonen er knyttet til, altså hvilken tabell i lokasjonsdelene systemet skal slå opp i. En slik relasjon kan opprettes så mange ganger som ønskelig, som beskrevet i avsnitt 2.2.1, og dermed kan informasjon eksistere for flere lokasjoner samtidig.

Hva slags informasjon skal systemet ta vare på og hvordan

Et kontekstavhengig system skal ikke ta over for eksisterende informasjonssystem, men er et system som skal fungere som en slags personlig assistent som har kjennskap til omgivelsene. Derfor skal det ikke i dette systemet lagres informasjon som allerede finnes andre steder, eller i andre systemet. Det som blir viktig for mitt system er å lagre pekere til andre informasjonskilder og tjenestetilbydere, og hvordan kommunikasjon mot disse skal foregå. Systemet må også kunne lagre diverse opplysninger om informasjonene, som rettigheter, tilgjengelighet (begrenset feks. av tid), alternative kilder, osv.

Oppgaven ser her bort fra data som er nødvendig i lokasjonsdelen, da dette er gjennomgått i avsnitt 3.6.2.

Figur 9 beskriver i tillegg til type, to andre attributter:

- *URL:* Adresse til ekstern informasjon eller tjeneste.
- *Alternative description:* Beskrivende tekst, hvis ikke adressen (URL) er tilgjengelig.

Informasjon som ikke finnes fra før, må derfor opprettes i andre systemet, evt. som en egen hjemmeside, og adressen (URL) kan da legges inn i systemet

Integrasjon mot andre systemer

Hvordan et kontekstavhengig system vil integreres mot andre systemer er et viktig spørsmål. Ved NTNU Gløshaugen finnes det allerede flere systemer som studentene benytter hver dag, feks. E-post[14], Student Web[15], Innsida[16] og It's:Learning[17]. Målet med mitt system er ikke å erstatte disse, men å fungere som en portal som kan forminske innsatsen som kreves for å holde seg oppdatert i hvert enkelt system (se avsnitt 2.4.2). Et problem som er felles for alle slike systemer er at det krever innlogging, og innsats fra brukeren sin side for å holde seg oppdatert. En student som skal sjekke etter endringer i alle systemene, må logge seg inn i alle systemene, ofte med forskjellige brukernavn og passord. Ved å legge innloggingsinformasjon i mitt system vil dette kunne varsle brukeren uten at denne må logge seg inn i de forskjellige systemene. System reagerer da på en endring i tilstand, og ikke lokasjon som nevnt tidligere.

Fordelen med dette er at brukeren slipper å bruke tid på å logge seg inn i flere systemer, men likevel være sikker på at alle meldinger kommer fram tidsnok. Det er også mulig å registrere endringer i flere systemer samtidig, lage et sammendrag og presentere dette for brukeren. Derfor er det viktig å integrere eksisterende system på en måte som oppleves som sømløs for brukeren (se avsnitt 3.7).

Et slikt system forutsetter at innloggingsinformasjon blir håndtert på en slik måte at det ikke er mulig for uvedkommende å få tilgang til informasjonen (se avsnitt 6.3).

Måten mitt system er integrert med andre systemer på, er gjennom en integrert nettleser. Alternativet ville vært å presentere eksisterende system i en ekstern nettleser (dette vil i praksis si standard nettleser på brukerens maskin). En integrert nettleser gir bedre kontroll over hva som skal presenteres til brukeren.

Ved å integrere disse systemene, vil en bruker kunne få beskjed når en endring har skjedd i eksterne systemer. Måten dette kan løses på er at en bruker må logge seg inn, og systemet vil da automatisk sjekke feks. Innsida og It's:Learning for endringer, og evt. gi tilbakemelding om dette. Dette gjør at brukeren slipper unna med en innlogging, for å få tilgang til feks. mail, Innsida, It's:Learning og Studweb, og brukeren blir varslet når det skjer en endring.

[14] `http://webmail.ntnu.no`
[15] `http://studweb.ntnu.no`
[16] `https://innsida.ntnu.no`
[17] `https://innsida.ntnu.no/sso/?target=itslearning`

I de neste punktene diskuteres hvordan en integrasjon mot E-post (avsnitt 3.6.3), Innsida (avsnitt 3.6.3) og It's:Learning (avsnitt 3.6.3) løses.

Integrering av E-post

E-post er tatt med her da det gir et enkelt eksempel på hvordan en integrasjon vil fungere. E-post er noe alle studenter bruker tilnærmet daglig, så nye meldinger oppdages relativt raskt. Likevel, innsatsen som kreves for å logge seg inn tar tid, og noen ganger er det unødvendig da det kanskje ikke fantes nye meldinger eller innboksen kun inneholdt spam.

Ved å legge inn innloggingsinformasjon i et kontekstavhengig system, kan brukeren varsles ut i fra hvordan dette er satt opp i brukerens profil. Brukeren slipper unna med kun en innlogging, og kan være trygg på at systemet gir et varsel når det har kommet en ny E-post. Måten systemet varsler brukeren på, kan endres feks. ved at en melding vises på skjermen eller at brukeren selv sjekker systemet for nye meldinger.

Studentenes nettbaserte E-postleser kalles "Webmail". Når systemet har lagret brukerinformasjon om E-mail kontoen, er det mulig for et kontekstavhengig system å registrere en endring, ved feks. å ta vare på tidspunktet for siste mottatte E-post. E-post har også en attributt som sier om en E-post er lest eller ikke. Dette kan kan utnyttes. Ved å sjekke for nye E-poster kontinuerlig, feks. hvert tiende minutt, kan et slikt system varsle om en eller flere nye E-poster.

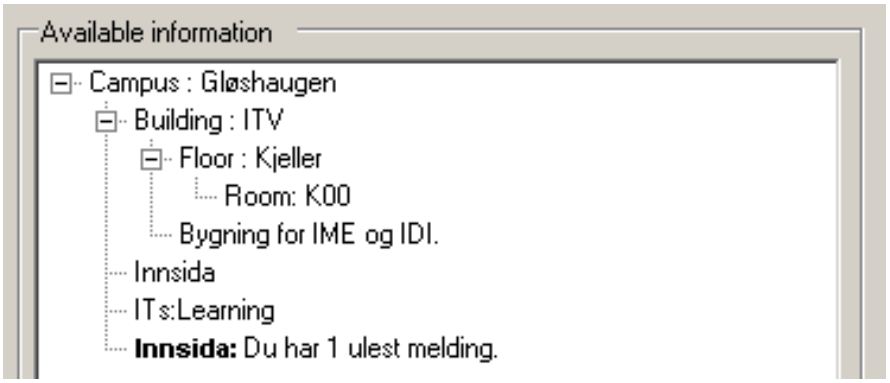
Når det har skjedd en endring, er det brukerens profil som avgjør om brukeren skal varsles eller ikke. Slik tilfellet er med E-post kan det også integreres et spam-filter, og E-post som gjenkjennes som spam vil ikke bli varslet. Men skjer det en endring, i dette tilfellet en ny E-post, vil systemet videresende brukeren til Webmail, og merke meldingen som lest. På denne måten ivaretas funksjonaliteten til E-postleseren i det kontekstavhengige systemet.

Integrering av Innsida

En integrering av Innsida i et kontekstavhengig system vil la systemet overvåke Innsida, og gi brukeren beskjed når det har skjedd en endring. I dette tilfellet at en ny melding har blitt langt inn. Systemet kan da velge å informere om dette aktivt ved å gi brukeren en melding, eller passivt ved å oppdatere sitt eget grensesnitt med en ny melding. I det siste tilfellet må brukeren åpne systemet for å få med seg nye meldinger. Et eksempel på dette er vist i Figur 10. Her har brukeren to valg: Det første er å la meldingen ligge. Det andre er å se på meldingen med en gang. Om brukeren velger det første vil ikke meldingen forsvinne før meldingen er lest. Hvis brukeren venter lenge med å lese meldingen, kan det dukke opp flere meldinger som da samles opp. Velger brukeren det siste, viderefører systemet kontrollen til Innsida og merker meldingen(e) som lest.

Utfordringen ligger i å plukke ut de meldingene som kan sies å være relevante for brukeren. På Innsida legges det ut informasjon som gjelder hele NTNU, noen er relevante for alle, mens de fleste er relevante for noen få. Ved å sette opp regler

i en profil, kan systemet gjøre en utvelgelse av meldinger som kan sies å være relevante for brukeren (se avsnitt 3.8).



Figur 10: Mitt system: Innsida.

Det finnes flere teknikker for å oppdage en endring feks. på en hjemmeside, og for å hente ut relevant informasjon. Siden dette ikke er en del av problemstillingen nevnes det for interesserte lesere at en teknikk som er mye brukt kalles "screen scraping" (se avsnitt 2.4.3). Kort fortalt henter screen scraping ut data fra informasjon som er lesbart for mennesker, i motsetning til vanlige parsere som baserer seg på data som er lagret for at maskiner lett skal kunne forstå innholdet.

Se også avsnitt 2.1.2 for en forklaring av Innsida.

Integrering av It's:Learning

En integrasjon mot It's:Learning er lik den for Innsida, men det unntak at i It's:Learning er de fleste nye meldinger studentspesifikke, dvs. at de er relevante i utgangspunktet. Utfordringen her blir hvordan et sammendrag av informasjonen skal presenteres. Det kan også være aktuelt å overføre kontrollen til It's:Learning for hver gang det oppdages en endring. Endring vil kunne oppdages på samme måte som i Innsida (se avsnitt 3.6.3).

Oppgaven går ikke nærmere inn på hvordan en slik integrering kan løses siden den er tilnærmet lik en integrering av Innsida, men diskuterer i avsnitt 3.7 hvordan en slik integrasjon kan løses hvis ikke kontrollen overføres til It's:Learning.

Se også avsnitt 2.1.2 for en forklaring av It's:Learning.

3.7 Presentasjon av informasjon

Informasjon kan presenteres på to måter. Den første overfører kontrollen til de eksterne systemene gjennom den integrerte nettleseren. Funksjonaliteten vil da bli lik for den som finnes i det eksterne systemet, og ingen implementering av funksjonalitet er derfor nødvendig. Fordelen med dette er at brukeren slipper å

logge seg inn i det eksterne systemet, og at det gis beskjed først når det har skjedd en endring.

Den andre måten er når kontrollen ikke overføres til det eksterne systemet. Her vil systemet hente inn informasjon, som vil behandlet utifra brukerens profil. Under diskuteres hvordan informasjon kan presenteres når kontrollen ikke overføres til det eksterne systemet.

Et kontekstavhengig system skal gi brukerne en rask oversikt over informasjon tilgjengelig. Siden informasjon kan komme fra flere eksterne kilder, er det nødvendig med en strategi for hvordan informasjonen skal presenteres. A9[1] er et system som kan vise til gode resultater der brukerne kan personalisere en startside vha. å legge til moduler. Disse modulene kan inneholde væroversikt, nyheter, E-mail, venner osv. På samme måte kan et kontekstavhengig system presentere informasjon ved å opprette en eller flere moduler for hvert eksternt system, og la brukeren selv bestemme hvor og hvordan informasjonen skal presenteres. Modulene vil inneholde den samme funksjonaliteten som i det eksterne systemet, som feks. les melding, lagre melding og slett melding. Selve lagringen skjer i det eksterne systemet, for å unngå dobbeltlagring.

Modulene skal enkelt kunne legges til, endres, slettes eller flyttes, ut i fra brukerens behov. Modulene danner en startside (notattavle\samleside), i likhet med den som er foreslått i [1].

3.8 Profiler

Profiler er foreslått for både å kunne øke presisjonen[21, 24], og for å kunne tilby brukerne muligheten til å "skreddersy" sin klient. I dette avsnittet diskuteres hva en profil er , for så å se nærmere på hvordan en profil kan være med å øke nøyaktigheten og til slutt hvordan en profil kan være med å tilpasse systemet til brukerens behov.

En profil er et sett med data som er knyttet til en bestemt bruker. Dette settet kan inneholde standard informasjon som: navn , adresse, telefon, stilling, osv. Det er også mulig å lagre informasjon om hvordan brukeren vil at systemet skal se ut, lagre preferanser, definere tidssone, lagre data om hvordan brukeren benytter systemet, osv. Det siste er for å kunne tilby brukeren valg, økt funksjonalitet - kort sagt la brukeren forme systemet på en slik måte at kommunikasjonen mellom bruker og system foregår så enkelt som mulig.

I et slikt system er det nødvendig å kunne posisjonere enheter så nøyaktig som mulig. Når man benytter seg av trådløse signaler er de gjenstand for store variasjoner med tanke på støy, bygningstopologi, forskjellig hardware og innblanding fra andre signaler. Det er derfor ikke alltid mulig å posisjonere en enhet så nøyaktig som ønsket. Det er her profiler kan være med på å forbedre nøyaktigheten. Ved å knytte data om posisjon opp mot en brukerprofil, vil det være mulig å bruke dette når systemet senere skal posisjonere brukeren. Dette er best forklart med et

eksempel: En bruker har et kontor i en etasje der det finnes mange andre kontor. Over tid vil det være registrert at brukeren befinner seg ofte på samme lokasjon feks. "Rom 123". Hvis systemet foreslår at brukeren befinner seg i "Rom 122", som er naborommet, kan man ved hjelp av lokasjonsdataene knyttet til profilen med stor sannsynlighet fastslå at brukeren faktisk befinner seg i rom 123. Her må det legges til grunn hvor stor forskjell det var i poengsum for hhv. rom 122 og 123, sett opp imot sannsynligheten for at brukeren kan ha befunnet seg i rom 122 for å diskutere noe med en kollega.

Data knyttet til profiler kan også være med å forbedre posisjonering der det befinner seg mange mennesker, noe som ofte er et problem med trådløse signaler siden menneskekroppen påvirker slike signaler. Ved å se på bevegelsesmønster i feks. en gang, kan man i et slikt system kunne forutsi hvor brukeren beveger seg. Dette vil være enten oppover eller nedover i gangen, men ikke feks. i etasjen over eller under. Dette er basert på hvor ofte målingene blir tatt kontra hvor fort en bruker kan fysisk bevege seg gjennom bygningen.

3.9 Oppsummering mitt system

I dette kapitlet er det foreslått et kontekstavhengig system som består av to moduler. En modul vil ta seg av lokalisering av klienter (lokasjonsmodul), mens den andre vil ta seg av presentasjon av informasjon basert på lokasjon og endring i tilstand (informasjonsmodul). Systemet baserer seg på tidligere systemer som GUIDE og Nibble, mens AROUND-arkitekturen fungerer som bindeleddet mellom lokasjon og informasjon. GUIDE har også bidratt til å identifisere hvilke typer informasjon som kan være aktuell i et kontekstavhengig system. I tillegg er erfaringer gjort fra RADAR, Wireless Campus LBS og Grimstad Museum, samt datastrukturer som metadata, Gazetteers og hierarkisk modellering, lagt til grunn for systemet.

3.9.1 Forutsetninger og valg

- Nibble danner utgangspunktet for lokaliseringsmodulen.
- AROUND-arkitekturen og GUIDE blir brukt i informasjonsmodellen og informasjonsmodulen.
- Informasjon skal hentes fra eksisterende systemer.
- Systemet har fokus på brukerne, ikke administrative roller.
- Systemet skal tilby: Lokalisering, informasjon basert på lokasjon og\eller tilstand og søk (etter sted, informasjon, og personer).

4 Prototyping

På bakgrunn av kapittel 3 foreslås prototypen "Fumble", et kontekstavhengig informasjons system. Målet er å kombinere funksjonaliteten til et kontekstoppmerksomt system med informasjonssystemer. Utfordringen er å skape en relasjon mellom disse som overfører den virkelige relasjonen (sammenhengen mellom det fysiske rommet og informasjonen) til en tenkt relasjon (sammenhengen mellom det fysiske rommet og informasjonsrommet[18]).

Utvikling av et kontekstavhengig system er en svært vanskelig oppgave da det ikke finnes en generisk modell som beskriver relasjonen mellom fysisk lokasjon og informasjon\tjenester[16, 17]. Det som skiller dette arbeidet fra systemene nevnt i avsnitt 2.3, er at trådløs posisjonering (som beskrevet i Nibble) er koblet tett til en informasjonsmodell (som beskrevet i AROUND og GUIDE) for å oppnå et innendørs kontekstavhengig system. For å klare dette har det vært nødvendig å lage en lokasjonsmodul og en informasjonsmodul. Lokasjonsmodulen lagrer lokasjonsdata på en slik måte at den kan sies å støtte informasjonsmodulen, og sammen skape et kontekstavhengig system.

Et kontekstavhengig system slik det er definert i denne oppgaven inneholder et utvalg av nødvendige brukerkrav for å vise hvordan et slikt system vil fungere. Kravspesifikasjonen inneholder altså ikke alle krav til et ferdig system. Alle kravene som ble identifisert ble heller ikke implementert (se avsnitt 4.4).

Et kontekstavhengig system er helt avhengig av et subsystem som kan registrere en endring i kontekst. Derfor beskrives først arbeidet med Nibble i avsnitt 4.1, både fordi dette arbeidet har vært avgjørende for prototypen og for å gi en generell forståelse for hvordan Nibble og kontekstoppmerksomme systemer kan fungere. Dette avsnittet tar for seg første delmål i oppgaven: Om posisjonering er mulig ved NTNU Gløshaugen. Andre delmål i oppgaven beskrives i avsnitt 4.5, mens avsnitt 5.4 diskuterer delmål tre.

Merk! For enklere å forstå hvordan lokasjonsdelen fungerer anbefales det å lese min definisjon på "sample" og "fingeravtrykk", se Appendix A.

4.1 Innledende testing

Høsten 2005 ble det gjennomførte flere tester for å finne ut om posisjonering er teknisk mulig ved NTNU vha. eksisterende WiFi utstyr. Programmet som ble valgt var Nibble (se avsnitt 2.3.1), og resultatene av disse testene er beskrevet i avsnitt 4.1.4. Arbeidet med Nibble ga mye kunnskap om kontekstoppmerksomme systemer basert på trådløse signaler.

[18] Information space.

4.1.1 Nibble Location System

Nibble (fullt navn: Nibble Location System) er som nevnt et innendørs system for posisjonering av mobile enheter utstyrt med et trådløst nettverkskort. Da Nibble ble utviklet ved Department of computer Science, UCLA[19] ble det laget støtte kun for trådløse nettverkskort av typen Lucent Orinoco. Grunnen til dette er at Lucent bruker et spesielt chipsett i sine trådløse kort som sammen med avansert programvare gjør det mulig å hente ut signaldata.

Det er flere grunner til at Nibble er aktuell for denne oppgaven. For det første er programmet åpen kildekode, dvs. at det er publisert under GPL (General Public License) og kan brukes og endres fritt så lenge betingelsene for GPL blir overholdt[11]. For det andre benytter Nibble seg av eksisterende trådløs dekning, dette er viktig fordi det allerede finnes et trådløst nettverk ved NTNU, og i tillegg er Trådløs Trondheim (TT) under utvikling når denne oppgaven skrives. Slik det ser ut nå blir TT basert på WiFi[5] og gjør at Nibble kan brukes direkte i det nye nettet. Den tredje grunnen er nøyaktighet når det gjelder posisjonering. Iflg. [21] kan klienter posisjonere helt ned til en feilmargin på bare 10 fot (3 meter), og dette er uten bruk av filter og profiler[20], noe som kan være med å øke nøyaktigheten ytterligere.

Ved å bruke det trådløse nettet støtter også Nibble dataoverføring som skalerer automatisk opp med den trådløse dekningen som finnes, da posisjoneringsteknologien er uavhengig av hastighet på trådløsnettet. En siste grunn er muligheten for å legge inn faste lokasjoner med en beskrivelse, dette med tanke på interesseområder.

4.1.2 Hvordan virker Nibble?

Figur 11 gir et eksempel på trådløs dekning i IT-Vest (ITV) som er en bygning på NTNU Gløshaugen. Figuren er forenklet og representerer ikke virkeligheten, men er ment å øke forståelsen for hvordan Nibble fungerer. I figuren er det tre forskjellige aksesspunkt (AP) og to brukere som begge har en bærbar datamaskin med trådløs tilkobling. Ser vi først på Bruker 1 befinner denne seg i rom 263 som er et arbeidsrom for studenter. Dette rommet er i hovedsak dekket av AP 2, med unntak av AP 3 som overlapper AP 2 i deler av rommet. Posisjonen til Bruker 1 kan i dette tilfellet være i hele dekningsområdet til AP 2, minus det området som AP 3 overlapper. Hvis bruker 1 kun ser AP 2, kan Nibble med stor sannsynlighet fastslå at Bruker 1 befinner seg i rom 263.

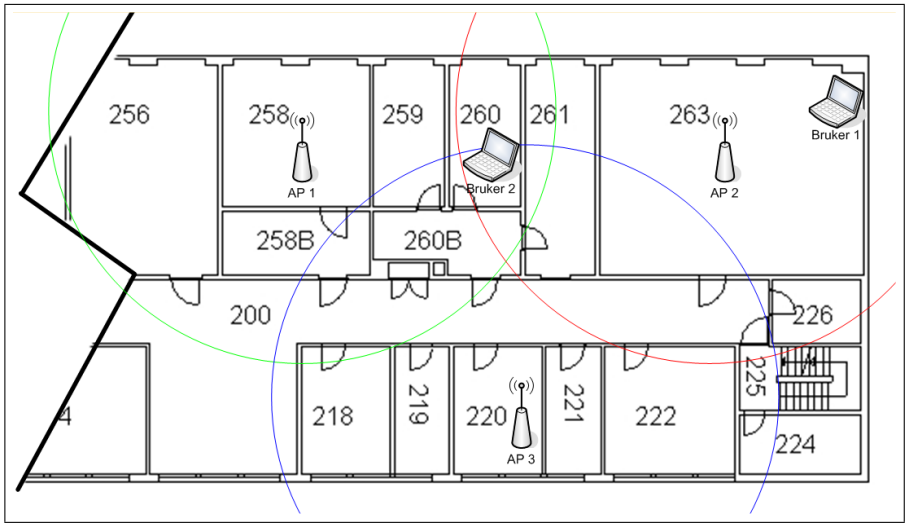
Et mer reelt eksempel er Bruker 2 som befinner seg innenfor dekningsområdet til både AP 1, 2 og 3. Dekningsområdet til WiFi er innendørs ca. 100 meter i alle retninger. Det vil si at her er det mulig å posisjonere ikke bare til et helt rom, men også hvor i rommet brukeren befinner seg. Dette er mulig fordi signalstyrken

[19] `http://www.cs.ucla.edu/`

[20] Filter brukes for å øke presisjonen på posisjonering. Et eksempel på profiler er gjennomgått i avsnitt 3.8.

vil være unik for alle de tre aksesspunktene i forskjellige deler av rommet.

Det skal nevnes at det i ITV finnes trådløs dekning i hele bygget. Dette gjør at det finnes flere tilgjengelige AP-er med varierende signalstyrke. Også signaler fra AP-er i etasjen over og under vil kunne hjelpe et slikt kontekstoppmerksomt system.



Figur 11: Eksempel på trådløsdekning i Nye Fysikk

4.1.3 Topografi ved IT-Vest

IT-byggene på NTNU Gløshaugen består av IT-Vest og IT-Sør. I IT-Vest finnes det 19 aksesspunkt fordelt på fem etasjer og plasseringen for disse er:

Etasje	**Rom**
Sokkel	000, 011, 060
1. etg.	100, 106, 143b, 160
2. etg.	200, 207, 258b, 243b
3. etg.	300, 306, 346b, 358
4. etg.	400, 406, 443b, 458

Tabell 1: Tabell: Oversikt over aksesspunkt ved IT-Vest

For 000, 100, 200, osv. er dette selve korridoren. AP er plasser i oasen[21] i hver etasje. Bygningsplan finnes på `http://www.ntnu.no/kart/no/gloshaugen.html`.

[21] Kaffekrok\myldrerom i hver etasje.

4.1.4 Test av Nibble

Testene ble gjennomført på en lørdag for å minimere støy fra omgivelsene. På forhånd ble det satt opp en testplan som hadde som mål å få svar på tre spørsmål: 1) Er posisjonering mulig vha. WiFi ved NTNU, 2) Kan Nibble skille mellom rom ved NTNU og i så fall 3) Hvor nøyaktig er Nibble ved NTNU?

Testplan:

1. **Er posisjonering mulig vha. WiFi ved NTNU?** Nibble skal i teorien kunne kjøre på hvilket som helst trådløst nettverk (inkludert Bluetooth og CDPD)[21], men dette var ingen garanti for at systemet vil virke ved NTNU Gløshaugen[22]. Ved å ta et par målinger på forskjellige steder og analysere innholdet i default.xml kan det avgjøres om Nibble er i stand til å hente ut signalstyrke ved det trådløse nettverket på Gløshaugen. Testen er vellykket om systemet henter ut forskjellig signalstyrke på de forskjellige stedene.

2. **Kan Nibble skille mellom rom ved NTNU?** Neste test er å se om systemet er i stand til å skille mellom to forskjellige rom. Posisjonene som ble valgt var: 1) oasen i 2 etg. IT-VEST og 2) Rom 263 IT-VEST. Oasen er et åpent rom (ingen dør) og har et aksesspunkt midt i rommet. Rom 263 er et arbeidsrom for studenter og inneholder ca. 20 pc'er og nærmeste aksesspunkt er i rom 258b (se `http://www.ntnu.no/kart/no/gloshaugen.html`). Testen er vellykket om systemet klarer å skille mellom oasen og rom 263.

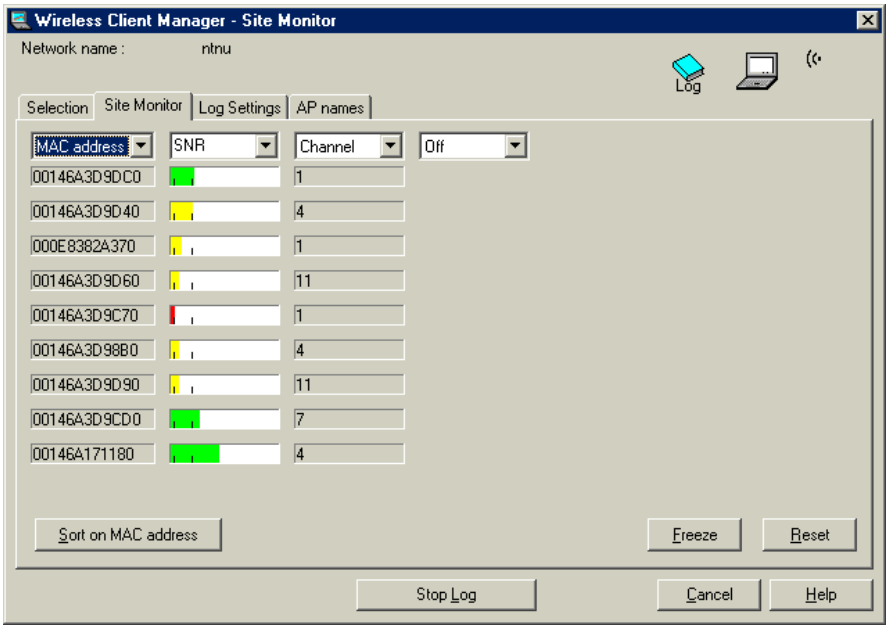
3. **Hvor nøyaktig er Nibble ved NTNU?** Til slutt ble Nibble testet for å se om det kunne skille mellom to forskjellige posisjoner i samme rom. Rommet som ble valgt var rom 263 som er ca. 100kvm stort, og posisjonene var ved døra (vest i rommet) og en arbeidsplass (øst i rommet). Testen er vellykket om systemet er i stand til å skille mellom disse to posisjonene.

Først ble posisjonene lagt inn, og alle ble forsterket en gang. Resultatet viser at systemet er i stand til å skille mellom posisjonene oppgitt i testen, men kan bli forvirret pga. endring i signalstyrke eller retning. Dette kan være pga. for få forsterkninger. Ved å forsterke posisjonene flere ganger ble Nibble mindre utsatt for forvirring. Siden feil likevel kunne forekomme, enn svært sjelden, så viser foreløpige resultater at Nibble har en feilmargin på ca. 3 meter ved rom 263 IT-VEST. Dette er i samsvar med resultatene i [21].

Nibble bestod alle testene, og dermed er det bevist at posisjonering er mulig ved NTNU Gløshaugen.

Grunnen til dette gode resultatet blir tydelig når vi ser på antall aksesspunkt som er tilgjengelig i rom 263 ved IT-VEST, se Figur 12.

[22]Selv om utstyr er merket med WiFi er det ikke garantert at de vil fungere sammen.



Figur 12: Signalstyrke i rom 263 IT-VEST

Hele ni aksesspunkt er tilgjengelig, mens teknikken brukt i Nibble trenger bare tre aksesspunkt for å kunne gi en nøyaktig lokasjon[21]. (Et aksesspunkt hadde en så svak signalstyrke at det ikke var tilgjengelig da skjermbildet ble tatt.)

4.1.5 Evaluering av Nibble

Nibble er et system som er tilgjengelig, og testene viser at det virker. Det hadde vært mulig å bruke Nibble som lokaliseringssystem, men det var flere grunner til at dette ikke ble gjort. Dette til tross for at Nibble kan vise til gode resultater ved NTNU Gløshaugen.

Nibble benytter seg av SNR (Signal-to-Noise Ratio), men det har i senere tid blitt antydet at SNR ikke er en egnet funksjon for lokalisering[24, 18, 17].

Hovedgrunnene til at Nibble ikke ble valgt er oppsummert under:

1. Mye av teknologien brukt i Nibble allerede så gammel at den er vanskelig å få tak i.

2. Inference algoritmen[23] er ikke inkludert som kildekode, og det kan da diskuteres om Nibble er et open source program.

3. Nibble er kodet i Java, et språk som er mindre egnet for rask utvikling enn feks. .NET[24].

[23] Algoritmen som beregner posisjon.

[24] Det er en allmenn oppfatning at .NET er svært godt egnet i prosjekter som har begrenset

4. Nibble er låst til en spesiell type trådløse nettverkskort, og det viste seg at det var vanskelig å få tak i både kortet og tilhørende programvare. Derfor er ikke dette egnet som system for brukere som har et bredt spekter av trådløse nettverkskort.

5. Nibble er avhengig av en ekstern DTD. Uten denne virker ikke programmet. Sagt på en annen måte: Er signalet for svakt for dataoverføring, vil ikke Nibble virke. Selv om dataoverføring ikke er mulig, er selv svake signaler gode nok for posisjonering.

6. Nibble lagrer lokasjonsdata i et Bayesian Network, en struktur som egner seg dårlig for denne løsningen. Å forandre dette ville krevd større arbeidsinnsats enn å lage et nytt system.

7. Den versjonen av Nibble som er tilgjengelig støtter ikke naboskap (adjecency).

4.2 Lokasjonsmodul

Basert på arbeidet med Nibble, ble det etterhvert klart at det ville bli både raskere og enklere å utvikle et eget system for posisjonering. Flere grunner til dette er nevnt i avsnitt 4.1.5. For dette systemet ble det satt en tidsramme på to uker og dette ble utviklingens første iterasjon: Å utvikle en lokasjonsmodul tilsvarende Nibble. Iterasjonen var ferdig når systemet kunne skille mellom to rom, eller tidsfristen gikk ut. Under beskrives hva slags teori som er nødvendig for å utvikle en lokaliseringsmodul:

For det første trengs det kunnskap om trådløse signaler. Dette er kort gjennomgått i avsnitt 3.6.2. For det andre må problemet med å hente ut disse signalene løses før arbeidet med selve prototypen kan begynne. Hvordan Fumble løser dette er beskrevet i avsnitt 4.2.1. Til slutt trengs det en strategi for å kunne dedusere posisjon. Heri Ramampiaro, Førsteamanuensis ved NTNU, foreslo en lineær algoritme. En lineær algoritme sjekker alle lagrede lokasjoner og beregner ut i fra dette hvilken som er mest sannsynlig basert på signalene prototypen mottar.

Det er også verdt å merke seg måten Guide løser posisjoneringen på kan være aktuell for traseen fra byen, via Gløshaugen, til Dragvoll, da det her vil være for få aksesspunkt til å få en nøyaktig posisjonering (se også avsnitt 6.3).

4.2.1 Valgt teknologi i lokasjonsdel

Valget om å utvikle en egen lokasjonsmodul åpnet for flere nye spørsmål. Blant annet har flere trådløse teknologier og teknikker vært diskutert i avsnitt 2.1.1. Likevel er det bare en teknologi og en teknikk som har vært aktuell for denne

med tid og resurser.

oppgaven og det er WiFi kombinert med fingerprinting. Fordelene og ulempene med disse er diskutert i kapittel 2.

- **WiFi**

  WiFi er en trådløs teknologi som gjør brukere i stand til å kommunisere med et datanettverk trådløst.

  Hovedgrunnen til at WiFi ble valgt er at det allerede finnes et trådløst nettverk basert på WiFi ved NTNU Gløshaugen. Dette er tilgjengelig for alle studenter som har en datamaskin, i de fleste tilfeller en bærbar pc, med et trådløst nettverkskort. Dette gjør at det ikke er noen ekstra kostnader knyttet til installasjon eller kjøp av ekstra utstyr for å implementere et kontekstavhengig system ved NTNU Gløshaugen.

- **Fingerprinting**

  Fingerprinting er en teknikk som registrerer tilgjengelige trådløse signaler ved en lokasjon og bruker dette for senere å kjenne seg igjen ved lokasjonen.

  Hovedgrunnen til at fingerprinting ble valgt er at de andre teknikkene krever ekstra utstyr i tillegg til det trådløse nettverket (her WiFi). Fingerprinting har den store fordelen at den kan benyttet på eksisterende trådløse nettverk uten ekstra utstyr, noe som var en forutsetning for oppgaven.

Det at WiFi ble valgt som teknologi for trådløs posisjonering, gjør at spørsmålet om hvordan disse signalene skal hentes inn fortsatt stod åpent. Nibble baserte seg på Lucent Orinoco Client Manager, men denne er låst til en bestemt type trådløse nettverkskort. Siden brukerne av et kontekstavhengig system vil ha svært forskjellige trådløse nettverkskort, er ikke dette en reell løsning. Løsningen kom i form av NetStumbler[25], et gratis program som støtter de fleste nye trådløse nettverskort.

- **NetStumbler**

  NetStumbler er et verktøy som gjør det mulig å oppdage Wireless Local Area Networks (WLANs) vha. 802.11x protokollene, og det kan brukes til å feilsøke trådløse nettverk.

  Hovedgrunnene til at NetStumbler ble valgt er at dette programmet er gratis, oppdateres jevnlig for å støtte nye trådløse nettverkskort, og at det er mulig å legge inn egendefinerte scripts som kan utføre bestemte handlinger. Det at NetStumbler støtter scripting gjør NetStumbler i stand til å bearbeide signalene og lagre disse i feks. en fil eller database. I første iterasjon av Fumble ble signalene skrevet til fil, men det viste seg at dette kunne skape inkonsistens i dataene, da det ikke var mulige å låse filen for lesing og skriving. Løsningen viste seg å være å lagre dataene i en database. Kommunikasjon

[25] `http://www.netstumbler.org`

mot databasen foregår ved hjelp av transaksjoner, og inkonsistens problemet blir dermed ikke-eksisterende.

Valget om å lagre signalene i en database ble gjort etter at første iterasjon var ferdig. Selv om det tidligere ble sagt at så snart en iterasjon var ferdig ble det ikke gjort endringer i denne iterasjonen, ble dette likevel gjort fordi det av og til oppsto problemer som fikk prototypen til å krasje. Grunnen til dette var at endringene som må til for å skrive til en database istedet for en fil ble ansett som minimale.

Den innebygde script-funksjonen i NetStumbler er svært kraftig. Kort fortalt har NetStumbler endel innebygde funksjoner som blir utløst når spesielle betingelser er møtt. Disse er utnyttet i scriptet for å ta vare på signalstyrken til alle tilgjengelige aksesspunkt, og når alle er oppdaget skrives disse til disk.

Når signalstyrken til alle tilgjengelige aksesspunkt er gjort tilgjengelig via databasen, er det opp til Fumble å nyttegjøre seg av denne informasjonen. Hvordan dette blir gjort er diskutert i avsnitt 4.2.2.

Nibble er som nevnt kodet i programmeringsspråket Java. Under diskuteres hvorfor VB.NET ble valgt som programmeringsspråk.

- **VB.NET**

  VB.NET (Visual Basic .NET eller VB .NET) er en versjon av Microsofts Visual Basic som er en del av selskapets .NET produkt gruppe. Et av målene er å forenkle utvikling av Windows-applikasjoner og Web-løsninger. VB.NET er den første versjonen av Visual Basic som kan sies å være Objekt Orientert[35].

  Det er allerede nevnt at VB.NET er et språk som er bedre egnet for rask utvikling enn feks. Java eller Python, men dette var ikke den viktigste grunnen. Hovedgrunnen er at VB.NET er XML orientert. .NET Framework Essentials [29] begrunner dette med: "... it (dataset class) allows reading and writing of data and schema in XML, and is tightly integrated with XmlDataDocument, ...". Fordelene med dette er blant annet at .NET støtter interoperabilitet, skalerbarhet og ytelse uten at dette er noe som må tas hensyn til. Prototypen benytter seg intensivt av dataset-klassen, ergo skal kommunikasjon med andre .NET applikasjoner være støttet. Dette åpner for at andre kan ta over prosjektet, men bruke feks. C# .NET eller C++ .NET som programmeringsspråk.

  En annen viktig grunn til at VB.NET ble valgt er at det er mulig å produsere brukergrensesnitt uten at programmet nødvendigvis virker. Dette åpner for muligheten til å simulere funksjonalitet hvis det skulle vise seg at prosjektet tar for lang tid, eller det viser seg at det ikke var mulig å implementere funksjonalitet som virker.

  Mitt system (kapittel 3) foreslår en klient-server løsning, men Fumble ble implementert som et selvstendig program, der både klient- og serverdelen er

en del av samme system. Dette til tross for at VB.NET tilbyr klient-server systemstrukturer via ASP.NET. Fordelen med å utvikle en prototype som en Windows Applikasjon er at dette er raskere, det trengs ingen dedikert server (for å kjøre webserveren), og en Windows applikasjon er enklere å implementere. Ulempen er at prototypen ikke vil kunne kommunisere med andre klienter, slik de vil gjøre i et ferdig system, og brukergrensesnittet til en Windows Applikasjon er ganske annerledes fra en Web løsning. Dette kan skape en vanskeligere overgang fra prototype til ferdig system.

4.2.2 Inference

Lokasjonsmodulen slik den er implementert inneholder lineær inference algoritme. Siden Nibble ikke la ved sin inference algoritme var det nødvendig å utvikle en ny. (Inference forklares best ved: handlingen som går fra statistisk data til generalisering, ofte med stor grad av sannsynlighet.) Dette forklares best ved et eksempel:

Anta at det i systemet er lagret to posisjoner. Ved hver posisjon er to aksesspunkt tilgjengelig, AP 1 og AP 2. Ved punkt A er det lagret et fingeravtrykk "Ved punkt A" som inneholder en sample som vist i tabellen under:

AP_ID	**Signal**
1	-75dBm
2	-10dBm

Tabell 2: Tabell "Ved punkt A"

Ved punkt B er det lagret et fingeravtrykk "Ved punkt B" som inneholder en sample som vist i tabellen under:

AP_ID	**Signal**
1	-15dBm
2	-80dBm

Tabell 3: Tabell "Ved punkt B"

La oss anta at brukeren befinner seg ett sted mellom punkt A og B, og signalstyrken ved dette punktet er for AP 1 -13dBm og for AP 2 -78dBm. Posisjon blir bestemt ut ifra poeng, og poengene beregnes som følger:

$$F_{score}(x) = \sum_{i=1}^{x} |AP(lagret)x + AP(gjeldene)x|$$

hvor AP(lagret) er aksesspunktet lagret i databasen og AP(gjeldene) det aksesspunktet med samme adresse ved gjeldende posisjon.

Inference algoritme

```
for each AP(gjeldende)
   for each AP(lagret)
      if AP(gjeldene).MAC = AP(lagret).MAC then
         score += abs(AP(gjeldende).signal + AP(lagret).signal
```

Algoritmen er svært forenklet med tanke på implementasjonen, men viser grunnprinsippet. MAC er en unik identifikator for hvert aksesspunkt og signal er signalstyrken rapportert fra NetStumbler. For enkelhets skyld er signalstyrken omgjort til positive verdier i utregningen under.

Punkt A får en score på 130 (abs(75 - 13) + abs(10 - 78)) og punkt B får en score på 4 (abs(15 - 13) + abs(80 - 78)). Her gjelder at lavest score gir høyest sannsynlighet, da en score på 0 er en eksakt match. Ut i fra dette kan det konkluderes med at brukeren befinner seg nærmest (i nærheten av) punkt B.

4.2.3 Oppsummering lokasjonsmodul

Valget om å utvikle en egen posisjoneringsmodul viste seg å være holdbar, da det tok 10 dager før en fungerende modell var ferdig utviklet og operativ. De resterende dagene som var avsatt til utviklingen ble brukt til å forbedre nøyaktigheten. Dette gikk kort fortalt ut på å lage støtte for flere sampler ved samme lokasjon, og å ta med i beregningen hva forrige lokasjon var.

Lokasjonsmodulen er testet i avsnitt 5.1.

4.3 Informasjonsmodul

Informasjonsdelen er et resultat av flere faktorer, og den delen som har hatt størst fokus gjennom oppgaven. For det første har AROUND-arkitekturen vært en inspirasjonskilde for å utvikle relasjonen mellom lokasjon og informasjon. For det andre har brukerkravene og brukergrensesnittet ført til at det ble implementert en nettleser i prototypen. For det tredje har metadata og Gazetteers påvirket hvordan informasjon ble lagret.

Informasjonsmodulen ble implementert gjennom flere iterasjoner. Første iterasjon var å kunne tilby informasjon basert på lokasjon. Andre iterasjon ble å integrere Fumble med eksterne systemer, og dermed også implementere en nettleser. Tredje iterasjon var å kunne tilby søk, mens siste iterasjon var å kunne tilby en "buddytjeneste".

Under beskrives valgene som ble tatt i forbindelse med informasjonsmodulen.

4.3.1 Valgt teknologi i informasjonsdel

I tillegg til valgene diskutert i avsnitt 4.2.1 er det tatt flere valgt med tanke på informasjonsdelen. Valgene tatt i lokasjonsdelen har påvirket valgene i informasjonsdelen, mens valget av VB.NET gjorde at også informasjonsdelen er utviklet i VB.NET. Teknologiene er diskutert i kapittel 2, men unntak av nettleseren som diskuteres under.

- **AROUND**

  AROUND er en arkitektur for å knytte tjenester til en eller flere lokasjoner. Arkitekturen er åpen med tanke på posisjoneringssystem eller teknologi.

  AROUND [16] foreslår en arkitektur for å oppdage og relatere tjenester til en geografisk posisjon. Artikkelen opererer med to framgangsmåter: a) avstands-basert utvelgelse (distant-based utvelgelse), og b) omfangs-basert utvelgelse (scope-based utvelgelse), som nevnt i avsnitt 2.3.6. Dette avsnittet ser nærmere på selve arkitekturen.

  Grunnen til at kun omfangs-basert utvelgelse er aktuell for denne oppgaven er at avstands-basert utvelgelse krever at det er mulig å avgjøre avstanden mellom to punkter. Dette er ikke mulig ved hjelp av posisjoneringsmodulen slik den er implementert i denne oppgaven, og heller ikke mulig med tilsvarende posisjoneringsteknikker basert på WiFi som er tilgjengelig. Selv om artikkelen argumenterer for at begge modellene burde implementeres i et kontekstavhengig system, er det i artikkelen også kun benytte seg av omfangs-basert utvelgelse, selv om feks. GPS tilbyr avstand mellom to punkter.

  Som forklart i avsnitt 4.2 er lokasjonsdata organisert i en trestruktur, der "Campus" er rotnoden. Bygninger, etasjer og rom er hhv. første, andre og tredje nivå under rotnoden. Grunnen til en slik enkel struktur er at den er med på å støtte omfangs-basert utvelgelse av tjenester. Ser vi på figur 9 er lokasjonsdata representert ved sirkler, og tjenester som rektangler. Omfang eksisterer allerede i lokasjonsdata avhengig av hvor vi kobler tjenestene i treet. En tjeneste knyttet til noden "Campus" vil være tilgjengelig uansett lokasjon, mens tjenester knyttet til node 1.etg. vil kun være tilgjengelig i første etasje i bygning IT-Vest. Modell tilbyr også å lage egne omfangs områder ved å kombinere flere lokasjoner, som feks første, tredje og femte etasje i en bygning. Det er heller ingenting i veien for å kombinere en hel bygning med noen få rom fra en annen bygning.

Noe av informasjonen som blir presentert i "Fumble" har dedikerte komponenter, som feks. kontekstavhengige meldinger og alternativ tekst. Dette er gjennomgått i avsnitt 4.7.2. Ved å velge å lese en melding overfører Fumble kontrollen til en intern nettleser, som overtar ansvaret for å presentere informasjonen.

- **Nettleser**

  En nettleser er et program som gjør brukere i stand til å lese websider. Det finnes flere konkurrerende nettlesere på markedet som Opera, FireFox, Netscape Navigator og Internet Explorer.

  Grunnen til at det ble implementert en nettleser er at dette gjør det enkelt å integrere andre systemer. Nettleseren gjør at systemer som Innsida, webmail og It's:Learning kan presenteres i prototypen, i tillegg til at prototypen kan fungere som et mellomlag mellom andre systemer og brukeren for å tilby økt funksjonalitet.

  VB.NET rammeverket inneholder en plugin som gjør det lett å integrere en nettleser i Windows Applikasjoner. Når dette er gjort oppfører den seg akkurat som en hvilken som helst annen nettleser.

Arbeidet mitt ved NTNU har gitt meg erfaring med flere måter å lagre informasjon på. Spesielt viktig for denne oppgaven er metadata og Gazetteers.

- **Metadata**

  Metadata (avsnitt 2.2.3) er viktig for at systemet skal kunne nyttegjøre seg informasjon. Fumble inneholder kun et utvalg av dataposter med tilhørende beskrivelser. Dette fordi prototypen er kun ment som et verktøy som har som oppgave å bevise at det er mulig å gjøre det på foreslåtte måte. Det kan nevnes at det er enkelt å utvide prototypen med flere dataposter hvis dette er ønskelig.

  Postene som er valgt implementert er i tillegg til en unik ID:

  – Alternativ beskrivelse: Dette er en beskrivende tekst som trer i kraft hvis systemet ikke har en databærer tilgjengelig eller at meldingen ikke er knyttet til en webside.

  – URL: Denne inneholder en webadresse hvis meldingen er knyttet til en webside. Se punkt Gazetteers under.

  – Type: Denne inneholder en klassifisering av informasjonen. Se punkt Gazetteers under.

- **Gazetteers**

  Systemet har implementert en generell Gazetteer (se også avsnitt 6.3).

  Selv om Fumble kun benytter en generell Gazetteer, var det planlagt å implementere en standard Gazetteer som feks. ADL Gazetteer. Dette er en standard som tillater bidrag til samme lokasjon fra flere bidragsytere. Noe som vil være reelt i et ferdig kontekstavhengig system. Attributtene i en generell Gazetteer, med tilhørende oppføringer, er listet under:

  – Geografisk navn inneholder: Stedsnavn, Gyldig navn (Boolean), Dato_fra og Dato_til. I Fumble ble kun Stedsnavn implementert.

- Navnetyper inneholder Navnetypeskjema, Navnetype, Gyldig, Dato_fra og Dato_til. I Fumble er navnetypeskjema og navnetype støttet.
- Lokasjonsinformasjon inneholder: Spatial representasjon (Avgrenset til en boks, med vest, øst, sør og nord som grenser), og gyldig. Her skiller Fumble seg fra generelle Gazetteers, fordi Fumble representerer posisjon ved hjelp av trådløse WiFi signaler. Derfor representeres også forskjellige forekomster annerledes (se forøvrig avsnitt 4.2).

I tillegg til disse elementene er de noen alternative elementer som er aktuell for denne oppgaven.

- Metadatainformasjon. Når forekomsten ble innført, og evt. endret. *Innført_dato og Endret_dato.* Ikke implementert.
- Navnevariant. Et stedsnavn kan ha flere navn. *Stedsnavnvariant, Gyldig, Dato_fra og Dato _til.* Ikke implementert.
- URL. Peker til en webside med informasjon om forekomsten. *Link_URL, Innført_dato.* Implementert.

4.3.2 Kontekst

I likhet med lokasjonsmodulen skjer det en utvelgelse i informasjonsmodulen. Forskjellen er at informasjonsmodulen ikke er basert på sannsynlighet. En tjeneste eller informasjon er enten en del av kontekst eller ikke. Måten informasjonsmodellen er bygd opp, spesielt hvordan informasjon eksisterer i forhold til en lokasjon, er beskrevet i avsnitt 4.5. Algoritmen, for å velge ut informasjon som er en del av konteksten, er listet som pseudokode under:

Finn kontekstalgoritmen

Merk! Lokasjon er beskrevet som fingeravtrykk.

```
Hent Fingeravtrykk ID basert på Fingeravtrykk
Hent Rom ID basert på Fingeravtrykk ID
Hent Etasje ID basert på Rom ID
Hent Bygnings ID basert på Etasje ID

Finn kontekst for Rom ID
Finn kontekst for Etasje ID
Finn kontekst for Bygning ID

Presenter global kontekst
Presenter bygnings kontekst
Presenter etasje kontekst
Presenter rom kontekst
```

Algoritmen er svært forenklet med tanke på implementasjonen, men viser grunnprinsippet.

Informasjonsmodulen er testet i avsnitt 5.2.

4.3.3 Oppsummering informasjonsmodul

Informasjonsmodulen baserer seg på AROUND-arkitekturen og GUIDE, og er implementert slik det ble foreslått i kapittel 3. Dette med unntak av integrasjonen med eksterne systemer. Fumble vil kunne henvise til eksterne systemer, men det er nødvendig for brukeren å logge seg inn manuelt. Fumble er heller ikke i stand til å hente ut og tilpasse informasjon fra eksterne systemer.

4.4 Implementasjon i forhold til identifiserte brukerkrav

Ikke alle brukerkravene som ble identifisert er implementert, tabell 4 viser en oversikt over hvilke krav som er implementert, implementert kun som eksempel eller ikke implementert.

Brukerkrav	**Implementert**	**Eksempel**	**Ikke implementert**
1 Hvor er			X
2 Hva inneholder	X		
3 Hvor går jeg			X
4 Hvor befinner		X	
5 Hva er	X		
6 Endring av kontekst	X		

Tabell 4: Tabell Brukerkrav

Prototypen er altså begrenset til funksjoner som kan brukes til å besvare spørsmålet i oppgaveformulering (se avsnitt 1.1.2). Disse funksjonene bygger også på kravspesifikasjonen (avsnitt 3.4). Valg av hvilke krav som ble implementert ble tatt på grunnlag av tidsrammen, og om kravene med høy sannsynlighet lot seg gjennomføre. De kravene som ble valgt er implementert på en måte som illustrerer noe av funksjonaliteten ved et slikt program, mens brukergrensesnittet er et resultat av GUIDE[8] og generell brukergrensesnittdesign. Utover dette er det ikke lagt vekt på brukergrensesnittet.

Prototypen er implementert med følgene funksjonalitet:

- **Fumble:** Grovt sett er Fumble delt inn i fem hoveddeler: Lokasjon, informasjon, nettleser, søk etter informasjon\tjenester og søk etter venner (se Figur 13). Disse blir gjennomgått i de neste punktene. I tillegg finnes det noen funksjoner som er globale, disse er:

- **Finn lokasjon:** Dette gjør det mulig å slå av eller på funksjonen som lokaliserer klienten. Den finnes under Fil-menyen som "Scan on". For å slå på denne funksjonen klikker man på menyen og en hake vil indikere at funksjonen kjører. I et ferdig implementert system er det denne funksjonen brukeren vil ha tilgang til som har med lokasjon å gjøre. All informasjon om lokasjon vil være lagret på forhånd. Informasjonen om hvor klienten befinner seg presenteres i "Current location", se Figur 13. I figuren befinner klienten seg i IT-Vest (ITV), 2. etasje, rom 263, fingeravtrykk nr. 1 og beskrivelsen "Geir". De siste tallene er klokkeslettet for da målingen ble gjort.
- **Finn kontekstavhengig informasjon:** Denne funksjonen kjøres når "Finn lokasjon" returnerer en lokasjon. Selv om dette er en global funksjon, er den så nært knyttet til "Tilgjengelig informasjon" at det ble valgt å beskrive den under "Informasjon", se punkt Informasjon under.
- **Signalstyrke:** Dette er en funksjon brukeren ikke kan endre, men er med på å gi viktig tilbakemelding, se "Signal" Figur 13. For det første gir denne funksjonen tilbakemelding i en utviklingssammenheng: Mottar prototypen signaler? For det andre gir den viktig informasjon til brukeren av systemet: Ved lav signalstyrke forventer brukeren mindre av systemet, i motsetning til et sterkt signal som gjør at brukeren forventer mer[8]. I figuren er det et svakt signal, noe som kan føre til unøyaktig posisjonering. Tallet "2" forteller at det er måling nr. to (av tre) som behandles, og "Updating signal" forteller at måling nr. 2 var bedre enn måling nr. 1, derfor blir dette lagret og det gamle slettet. Alternativet er "Keeping old signal" og det betyr at måling nr. 2 var dårligere enn måling nr. 1 og dermed blir måling nr. 2 forkastet.

Figur 13: Fumble består av fem hoveddeler.

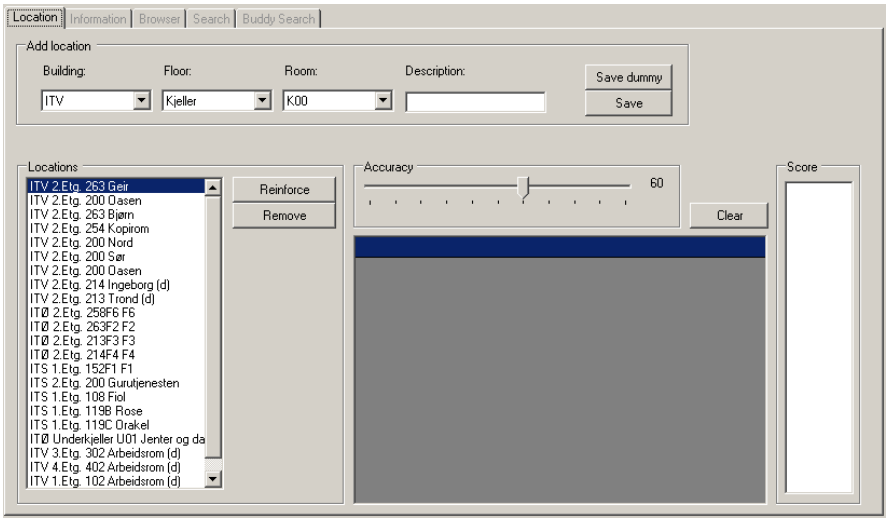
- **Lokasjon:** Prototypen er i stand til å lokalisere seg selv der det finnes flere trådløse nettverk. For å kunne støtte denne funksjonaliteten ble følgende funksjoner implementert:
  - **Legg til lokasjon:** Når klienten befinner seg på et sted der det kan være nyttig at klienten kan kjenne seg igjen, gir prototypen mulighet til å velge hvilken bygning, etasje og rom en befinner seg i, i tillegg til et beskrivende navn. Ved å legge til en ny lokasjon, tar prototypen flere målinger over seks sekunder og lagrer det beste i lokasjonsdelen

av databasen, se kapittel 4.2. Når dette er gjort, er prototypen i stand til å fortelle hvor brukeren er en annen gang når brukeren er på samme lokasjon.

- **Legg til en tom lokasjon:** Det er flere steder som er stengt for studenter, men som likevel vil eksistere i et kontekstavhengig informasjonssystem. Derfor ble det opprettet en funksjon som lagret kun bygning, etasje, rom og et navn, uten å ta målinger. Dette var nyttig for å ta skjermbilder, fordi det enkelt kunne legges inn mange lokasjoner. Disse blir ikke brukt når prototypen prøver å lokalisere seg selv. (Det skal nevnes at hvis man forsterker (se under) en slik tom lokasjon, vil den bli gyldig og tatt med i lokaliseringen.)
- **Forsterk lokasjon:** Ofte er det ikke nok å ta bare *en* måling på samme plass for å få et godt nok resultat. Støy i form av antall mennesker i bygningen, samt orientering, er med på å endre signalene. Derfor ble det valgt å implementere en mulighet for å forsterke en lokasjon. Denne gjør det samme som å legge til en lokasjon, bare at målingen blir lagt til i tillegg til de som allerede er lagt inn på samme lokasjon. Dette gir prototypen flere muligheter til å finne en måling som passer, noe som dermed er med på å øke nøyaktigheten.

  Det kan lønne seg å forsterke en lokasjon ved å peke klienten i alle himmelretningene, dermed vil posisjoneringen påvirkes av orientering i mindre grad. Det kan i tillegg være andre faktorer å ta hensyn til, som tid på dagen. Et avtrykk i skoletiden, vil skille seg fra et avtrykk tatt på kveldstid, derfor kan det være lurt å forsterke signalet på forskjellige tider av døgnet.
- **Slett lokasjon:** Denne funksjonen ble tatt med da det flere ganger ble gjorde feil ved å legge til feil lokasjon eller forsterke feil plass. All data knyttet til en lokasjon blir fjernet fra databasen. Denne funksjonen var også nyttig hvis det trådløse nettverkskortet ble byttet ut, og det var nødvendig å legge inn lokasjoner på nytt.
- **Juster nøyaktighet:** Forskjellige trådløse kort leser signalene ulikt. Derfor ble det valgt å implementere en mulighet for å justere nøyaktigheten manuelt. Å justere nøyaktigheten manuelt er enklere enn å justere den automatisk. I et ferdig system vil en slik løsning implementeres slik at systemet justerer denne nøyaktigheten selv. Denne funksjonen må også ta hensyn til hvordan "hva som er i nærheten" defineres. Sagt på en annen måte, graden av nøyaktighet avgjør hvor stort område en lokasjon skal dekke.

- **Informasjon:** Når lokasjon er kjent, er Fumble i stand til å presentere informasjon knyttet til lokasjoner. Dette er muliggjort gjennom informasjonsmodellen utviklet for denne type programmer (se kapittel 4.3.1). Informasjonen presenteres i en trevisning (Tree View) som oppdateres automatisk (se avsnitt 4.7.2).



Figur 14: Fumble : Lokasjon.

– **Type:** For enklere å kunne gruppere og søke i tilgjengelig informasjon har Fumble adoptert type-begrepet fra Gazetteers. Dette gjør at det er enkelt å finne alle tjenester som er knyttet til en type, og man trenger derfor ikke søke igjennom all informasjonen[16] for å få fram spesifikk informasjon. Funksjonen gjør det mulig å opprette en ny type som øyeblikkelig blir tilgjengelig for ny informasjon (se punktet under). Fumble implementerer kun et nivå med typer, da dette viser hvordan en slik oppdeling er med på å gruppere informasjonen. Gazetteers deler opp i flere nivåer av typer, der første nivå gir et bredt søk, mens nivå lengre ned gir et smalt søk.

– **Legg til informasjon:** Dette er den viktigste administrative funksjonen i informasjonsdelen. Ved å fylle ut feltene under "Add information" knyttes informasjonen som blir lagt inn til den lokasjonen som er oppgitt. Når en bruker befinner seg på angitt lokasjon, vil informasjonen være tilgjengelig for brukeren (se punktet under). For å legge inn informasjon, må brukeren avgjøre om denne er av en type som allerede eksisterer, eller om det må opprettes en ny type for denne tjenesten (se punktet over). Det viktigste steget i denne funksjonen er å avgjøre på hvilket nivå tjenesten skal være: Skal den alltid være tilgjengelig (global)? Eller skal den kun være tilgjengelig i et bestemt rom? Et feil valg her kan føre til at informasjon ikke kommer opp, eller kommer opp selv om brukeren ikke har nytte av den ut i fra gjeldene lokasjon. Det må også knyttes et navn til informasjonen, som skal beskrive hva slags område informasjonen er ment å dekke og hva slags type informasjon det er snakk om. Dette er tatt med for å forenkle gjenfinningsarbeidet når informasjon eksisterer i systemet. Et siste punkt som må være med er "Alternative description". Informasjon kan ikke eksistere i systemet uten en alternativ beskrivelse. Grunnen til dette er at hvis det ikke

knyttes en URL til informasjonen, er det denne beskrivelsen som dukker opp i systemet. Her er det mulig å skrive inn flere linjer med informasjon (ved å trykke enter i feltet), og for å kunne legge inn dekkende beskrivelser. URL, som er det siste feltet, er et valgfritt felt. Hvis det eksisterer en webside for informasjonen eller tjenesten, kan denne legges inn her, hvis ikke så kan den være blank. Hvis den er blank er det bare den alternative beskrivelsen som inneholder en beskrivelse av informasjonen. Knyttes det derimot en webside til informasjonen, kan brukeren enkelt klikke seg fram til denne (se punktet under). Denne koblingen til eksterne websider er viktig for at et slikt system skal integreres enkelt med eksisterende systemer. Det argumenteres for en slik kobling i [8].

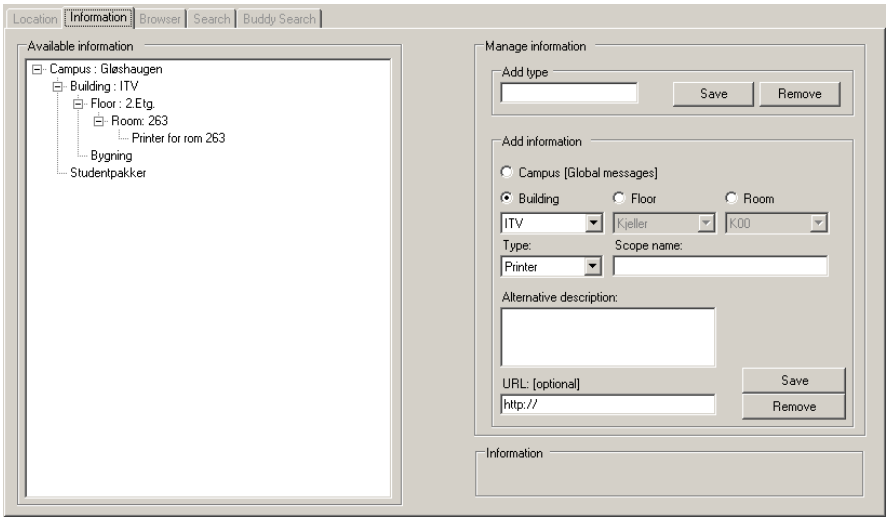
- **Tilgjengelig informasjon:** Dette er den viktigste brukerfunksjonen i informasjonsdelen, og det er denne funksjonen en bruker ville ha hatt tilgang til i et ferdig system. Mesteparten av metodene beskrevet her er implementert for å gjøre denne funksjonen tilgjengelig. Denne funksjonen gjør Fumble i stand til å presentere hva som befinner seg "her". Dette er altså hovedfunksjonen i et kontekstavhengig system.

  Det ble valgt en tre-visning (tree view) som grunnlag for presentasjonen, fordi dette er foreslått som den beste måten å vise hierarkisk data på i .NET. I tre-visningen finner vi igjen de samme nivåene som vi kjenner fra lokasjonsdelen, med unntak av "Campus" og lokasjonsnavnet. Grunnen til dette er at lokasjonsdelen antar at det er "Gløshaugen" det er snakk om som campus uansett, og informasjonsdelen opererer med relasjoner på romnivå, ikke innad i rommene. Begge disse forskjellene ville ikke eksistert i et ferdig system.

  Funksjonen får lokasjon fra lokasjonsdelen av programmet, og benytter dette til å avgjøre hva som skal presenteres som tilgjengelig informasjon. Global informasjon eller globale tjenester blir undersøkt først, det vil i praksis si at prototypen sjekker om det finnes noen relasjoner til "Campus"-nivået. Dette blir gjennomført uansett, altså uavhengig av om lokasjon er tilgjengelig eller ikke (avsnitt 6.4). Så sjekkes hvert nivå for relasjoner, først bygning, så etasje og til slutt rom. Relasjoner som oppfyller kravet til posisjon blir hentet fra databasen og presentert i en tre visning (Tree view) som vist i Figur 15.

- **Nettleser:** Siden det ble valgt å presentere informasjon vha. websider og integrere Fumble mot andre webbaserte system, ble en nettleser implementert. Alternativt ville ha vært å la gjeldende nettleser[26] på klienten starte for å presentere informasjonen\tjenestene. Ulempen ville da være at det kunne bli mange åpne nettlesere, og brukeren bli oversvømt av informasjon. Fordelen er at brukeren kan bruke den nettleseren de vanligvis bruker. Ved å integrere en nettleser har programmet full kontroll over hva som blir presentert.

[26] Gjeldene nettleser vil ofte være Internet Explorer, men kan ha blitt erstattet av feks. Opera eller FireFox.



Figur 15: Fumble : Informasjon.

- **Søk etter informasjon:** I et kontekstavhengig system er brukeren prisgitt de valg som systemet tar. Det er en vanskelig oppgave å velge ut hva som skal presenteres og hva som er irrelevant, og det kan skje at systemet tar feil. Erfaringene gjort i GUIDE[8] har blitt tatt hensyn til, og funksjonaliteten som lar brukeren søke i tilgjengelig informasjon uavhengig av lokasjon, er blitt implementert.

  - **Kategori:** Informasjon er organisert i kategorier, se avsnitt 2.2.4. Som nevnt har Fumble kun et nivå som inneholder flere typer for å illustrere hvordan dette kan implementeres. Et nivå med typer gir et svært bredt søk, og kan fort bli svært uoversiktlig.
  - **Stikkordsøk:** Dette er et enkelt søk som viser hvordan en søkefunksjon kan fungere. Søket er begrenset til kun et ord, eller del av ord, da dette ikke er en oppgave om å lage en søkemotor, men et eksempel på mulig funksjonalitet. Funksjonen lar brukeren søke etter, og få tilgang til, informasjon som ellers ikke ville vært tilgjengelig basert på kontekst.
  - **Vis informasjon:** Når ønsket informasjon er funnet, kan brukeren velge å få denne presentert. Ved å trykke på "Get info", vil prototypen hente websiden og presentere denne i nettleseren, uavhengig av hvor brukeren befinner seg. Informasjonen blir presentert i nettleseren på samme måte som da brukeren velger å se informasjon som er tilgjengelig ut i fra gitt posisjon, se "Tilgjengelig informasjon" over. Er ingen webside tilgjengelig vil prototypen presentere en alternativ beskrivelse i form av tekst.

- **Søk etter venner:** Omtalt som "Buddy Search" i prototypen er dette funksjonalitet som ikke er implementert i den forstand at den ikke virker. Brukergrensesnittet eksisterer, men koden bak mangler, grunnet at denne

iterasjonen ikke ble ferdig. Derfor er dette et eksempel på tenkt funksjonalitet.

- **Legg til venn:** Her er tanken at brukere kan legge til hverandre, feks. basert på et brukernavn eller E-mail. Ved å legge inn en annen bruker (en buddy), vil denne få et spørsmål om han eller hun vil tillate at informasjon, om hvor han eller hun befinner, seg blir gjort tilgjengelig for brukeren som spurte. Se også punktet "Godkjenn venn".
- **Fjern venn:** Dette fjerner en venn fra tillatelseslista, og dermed er ikke informasjon om lokasjon lengre tilgjengelig for brukeren som ble fjernet.
- **Godkjenn venn:** Når en venn godkjenner en forespørsel om å bli en buddy er det mulig for begge parter å foreta spørringer som: "Hvor befinner 'buddy' seg?" Prototypen illustrerer også flere mulige funksjoner som kan knyttes til en buddy-tjeneste, se under.
- **Chat, File Transfer, Webcam osv.** Systemet kan tilby venner flere måter å kommunisere på. De mest aktuelle teknologiene i dag er lynmeldinger, taleoverføring og videokonferanse. I tillegg til dette kan systemet tilby filoverføring, mail, dele kalender, osv. Her er det kun fantasien som setter grenser. Se også 6.3.

Prototypen prøver altså å tilby tjenester som er knyttet til lokasjon. Sett som helhet, gjør alle disse funksjonene Fumble i stand til å presentere hva som finnes "her" uten at brukeren må foreta seg noe.

4.5 Relasjonen mellom lokasjon og informasjon

Avsnitt 4.2 og 4.3 beskriver hhv. informasjonsmodulen og informasjonsmodulen. Hver for seg gir ikke disse modulene økt verdi for brukeren. Disse modulene må derfor implementeres slik at de snakker sammen, og en relasjon mellom lokasjonsdata og informasjonsdata må utvikles. Dette avsnittet beskriver hvordan FUMBLE implementerer en slik relasjon.

Dette avsnittet antar at lokasjonsmodulen alltid foreslår en lokasjon og gjør denne tilgjengelig for informasjonsmodulen.

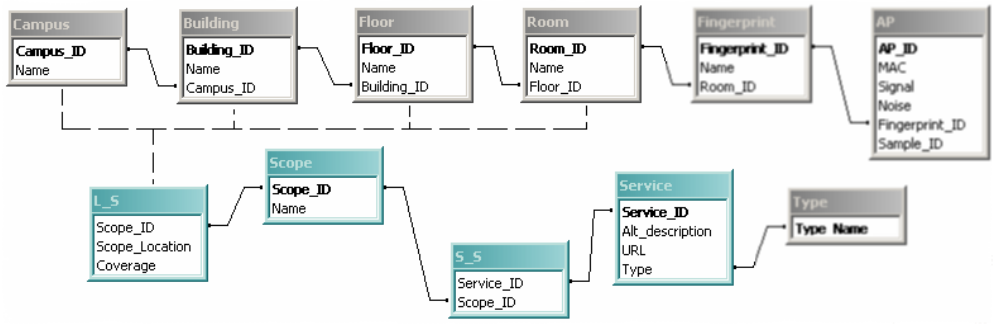
AROUND-arkitekturen foreslår å knytte en tjeneste (service) (se Figur 5) til forskjellige nivåer i lokasjonshierarkiet. Modellen er svært forenklet, da det må tas hensyn til at en tjeneste kan knyttes til flere lokasjoner og en lokasjon kan ha flere tjenester. I tillegg er det foreslått i denne oppgaven at systemet skal tilby informasjon (tekst) parallelt med tjenester.

Det var derfor nødvendig å utvide den foreslåtte AROUND-arkitekturen. Først ble modellen endret slik at en lokasjon kan ha flere tjenester. Dette ble løst vha. av et "scope". Et scope er en entitet som inneholder en eller flere tjenester. Et

scope, slik det er implementert i denne oppgaven, inneholder kun en unik ID og et navn, men det er også mulig å legge inn alternative attributter som en beskrivelse, dato for opprettelse, eier osv. Dette støtter opp under forslaget i AROUND ved at systemet skal slippe å søke igjennom hele det tilgjengelige informasjonsrommet, se avsnitt 3.5.

Oppgaven har tidligere sagt at et scope kan inneholde en eller flere tjenester. Dette skaper en en-til-mange relasjon. Men for at en tjeneste ikke kun kan eksistere i et scope, var det nødvendig å dele opp relasjonen mellom scope og tjeneste. Slik modellen er implementert ble det opprettet en ekstra entitet, S_S (Scope til Service), som løser opp denne mange-til-mange relasjonen i to en-til-mange relasjoner. Se Figur 16. Tjeneste er her beskrevet som "Service".

Et scope kan nå inneholde flere tjenester, og vice versa, men samme problem oppstår når et scope skal knyttes til en lokasjon: En lokasjon kan være knyttet til flere enn et scope, og et scope kan være knyttet til flere lokasjoner. Igjen har modellen en mange-til-mange relasjon. Denne ble delt opp i to en-til-mange relasjoner ved å innføre en entitet L_S (Lokasjon til Scope), på samme måte som for scope til tjenester. Forskjellen er her at lokasjon, slik definert i modellen, kan være: campus, bygning, etasje eller rom. Entiteten L_S inneholder derfor en attributt "Coverage" som sier hvilken av disse områdene som dekkes og hvilken lokasjon (for området "Room" hvilket rom, for området "Floor" hvilken etasje osv.) det er snakk om.



Figur 16: Relasjonen mellom lokasjon og informasjon i FUMBLE

Denne modellen skaper en relasjon mellom lokasjonsdata og informasjonsdata som utnytter strukturen i lokasjonshierarkiet og som samtidig støtter både informasjon og tjenester. For å utdype dette litt så er informasjonsdata knyttet til lokasjonsdata på en slik måte at systemet slipper å traversere lokasjonshierarkiet for å finne fram til korrekt lokasjon. Måten modellen støtter både informasjon og tjenester er vha. attributten URL i entiteten Service, ved å legge ut informasjonen på en webside, eller ved at informasjonen skrives inn i "Alt_description". Her bør det ta en avgjørelse som systemet skal kunne lagre informasjon internt (i Alt_description) og via eksterne websider, eller at informasjon kun kan knyttes til systemet via eksterne websider.

Se også avsnitt 6.3.

4.6 Datastruktur

Lokasjonsdataene er organisert i en klassisk tre-struktur. I et ferdig system ville rotnoden vært "Verden"[27], men siden dette er en prototype er rotnoden "Gløshaugen". Et nivå lengre ned finner vi foreløpig bygningene IT-Vest, IT-Øst og IT-Sør. Tredje nivå består av etasjer og fjerde nivå inneholder rom. Femte nivå inneholder fingeravtrykkene som er det minste området (altså et punkt) dette systemet kan behandle. Grunnen til at oppgaven har valgt denne strukturen er fordi dette er et innendørs lokasjonssystem, da utendørs posisjonering allerede er realisert gjennom GPS[28] og snart Galileo [29].

4.6.1 Datainnsamling

For å gjøre Fumble operativ må det samles inn data beskrevet som fingeravtrykk tidligere. Dette skjer i en offlinefase, eg. ingen lokalisering kan skje før denne fasen er over. (Det er derimot ingenting i veien for å legge til flere fingeravtrykk selv når offlinefasen kan sies å være avsluttet.) I offlinefasen lages og analyseres fingeravtrykkene med tanke på å øke presisjonen og lette gjenfinningsarbeidet i onlinefasen. I onlinefasen trekker prototypen slutninger ut i fra hva slags signaler den får inn på gjeldene lokasjon mot fingeravtrykkene som er lagret i offline fasen. Sammenlignet med [24] er offlinefasen den samme, men onlinefasen er noe endret. I RADAR[24] sender hver klient ut et signal som fanges opp av basestasjonene og signalstyrken lagres i disse. I vår løsning er det klienten som foretar en periodisk scan og lagrer signalstyrken lokalt. Uansett, dette har ingen betydning for presisjonen som konkludert med i RADAR.

Et trådløst nettverkskort samler inn masse informasjon om de trådløse nettverkene som er tilgjengelige, men bare et lite sett av dette er brukbart for lokalisering. Det er fire attributter som er aktuelle for lokalisering: MAC (Media Access Control), RSS (Recieved Signal Strenght), Noise og SNR (Signal-to-noise ratio). MAC brukes til å skille de forskjellige aksesspunktene fra hverandre. RSS, Noise og SNR kan alle benyttes for å lokalisere en trådløs klient, men som vi skal se er det bare RSS som er praktisk.

RSS sammen med Noise danner grunnlag for SNR, men som konkludert med i [24, 18] er RSS en sterkere funksjon enn SNR fordi SNR blir påvirket av tilfeldig støy (Noise). Dette er bekreftet gjennom arbeidet med Fumble, da presisjonen økte merkbart ved å basere lokaliseringen på RSS, i stedet for SNR. I tillegg viste seg at NetStumbler i noen tilfeller rapporterte -100dBm for noise for enkelte kort,

[27] Omtalt som "Campus" i kildekoden.
[28] `http://en.wikipedia.org/wiki/Gps`
[29] `http://en.wikipedia.org/wiki/Galileo_positioning_system`

muligens fordi enkelte kort ikke støtter å hente ut annet enn RSS. I disse tilfellene vil SNR ikke kunne brukes til posisjonering.

Datainnsamlingen foregikk ved at det ble lagt inn områder som kan være aktuelle for et kontekstavhengig system. Eksempler på dette er rommene 263 (studentarbeidsplass), 200 (korridor i 2.etasjen), Oasen (myldrerom i 2.etasjen), 254 (kopirom) osv. Alle testrommene er ved IT-Vest. Det er foreslått i [17, 18] å dele inn områdene i et forhåndsbestemt grid, med fast avstand mellom punktene, og det diskuteres om hvor stor denne avstanden skal være. Fumble har ikke valgt å bruke et grid, men heller legge inn punkter som er mest aktuelle. I spesialtilfeller derimot, for eksempel i auditorium, kan det være aktuelt å lage et slikt grid for å kunne posisjonere klienter innad i rommet.

4.6.2 Data i informasjonsmodellen

Dette avsnittet gir eksempler på innhold i databasen for både lokasjonsdelen og informasjonsdelen.

DATAMODELL:

Campus

Campus_ID (PK)	**Name**
1	Gløshaugen

Tabell 5: Tabell Campus

(Merk at for denne prototypen vil denne tabellen kun inneholde Gløshaugen.)

Building

Building_ID (PK)	**Name**	**Campus_ID** (FK)
1	ITV	1
2	ITØ	1
3	ITS	1

Tabell 6: Tabell Building

(Merk at for denne prototypen vil denne tabellen kun inneholde de tre bygningene ITV, ITØ og ITS.)

Floor

Floor_ID (PK)	**Name**	**Building_ID** (FK)
1	Kjeller	1
2	Sokkel	1
3	1.Etg.	1
4	2.Etg.	1
5	3.Etg.	1
6	4.Etg.	1
...	...	...

Tabell 7: Tabell Floor

Room

Room_ID (PK)	**Name**	**Floor_ID** (FK)
3034	K00	1
3035	K01	1
3036	K02	1
3037	K03	1
3038	K04	1
3039	K05	1
...	...	...

Tabell 8: Tabell Room

Fingerprint

Fingerprint_ID (PK)	**Name**	**Room_ID** (FK)
1	Geir	3543
2	Bjørn	3543
3	Test (d)	3543
4	Sør	3480
5	Oasen	3480
6	Nord	3480
...	...	...

Tabell 9: Tabell Fingerprint

(Merk at ID 3 “Test” er en tom lokasjon (dummy) symbolisert med “(d)”.)

AP

AP_ID (PK)	**MAC**	**Signal**	**Noise**	**Fingerprint _ID** (FK)	**Sample _ID**
1358	00146A171180	-66	-91	1	1
1359	00146A3D9D40	-81	-91	1	1
1360	00146A3D9DC0	-80	-89	1	1
1361	000FF7B69590	-85	-89	1	1
1362	00146A3D9CD0	-87	-90	1	1
1363	00146A171180	-71	-92	1	2
1364	00146A3D9D40	-75	-92	1	2
1365	00146A3D9CD0	-77	-92	1	2
...	...	...	...	...	...

Tabell 10: Tabell AP

(Merk at selv om systemet lagrer "Noise" er dette ikke lengre brukt i posisjoneringen.)

INFORMASJONSMODELL:

L_S

Scope_ID (FK)	**Scope_Location**	**Coverage**
1	1	Campus
2	1	Campus
3	3543	Room
4	1	Building
...	...	...

Tabell 11: Tabell L_S

Scope

Scope_ID (PK)	**Name**
1	Innsida
2	It's:Learning
3	Printer tilgjengelig
4	Bygning for IME og IDI.
...	...

Tabell 12: Tabell Scope

S_S

Service_ID (PK)	**Scope_ID** (FK)
1	1
2	2
3	3
4	4
...	...

Tabell 13: Tabell S_S

Service

Service _ID	**Alt_description**	**URL**	**Type**
1	Innsida	https://innsida. .ntnu.no/	Stud.Info.
2	It's:Learning	https://innsida.ntnu. .no/sso/?target= itslearning	Stud.Info.
3	HP LaserJet 4050	http://printhost /printers/itv263	Printer
4	Bygning for IME og IDI.	http://idi.ntnu.no	Lokasjons info.
...	...	...	...

Tabell 14: Tabell Service

Type

Type_Name
Adm.Info.
Forelesning
Lokasjons info.
Printer
Stud.Info.
Test info.

Tabell 15: Tabell Type

4.7 Brukergrensesnitt

Brukergrensesnittet er et resultat av tidligere prosjekter, der arkfaner (tabs) viste seg å være enkelt for mange brukere å bruke. Arkfaner gjør det enkelt å samle funksjonalitet, samtidig som det er enkelt å endre og legge til funksjonalitet.

Utover dette har ikke prototypen tatt særlig hensyn til grensesnittet, bortsett fra valget om trevisning (Tree view) i informasjonsdelen som nevnt i avsnitt 4.4.

Under beskrives kort brukergrensesnittet i lokasjonsdel og informasjonsdel.

4.7.1 Brukergrensesnitt lokasjonsdel

Brukergrensesnittet (se Figur 17) for lokasjonsdelen er likt det vi finner i Nibble, men med noen forbedringer. Grensesnittet lar en bruker velge hvilket rom klienten befinner seg i (prototypen forutsetter dermed at brukeren befinner seg innendørs, og rommet har et romnummer) og gir dette et beskrivende tekst ("Description"). Velger brukeren å lagre lokasjonen vil beskrivelsen dukke opp under "Locations". Dette er lokasjoner som prototypen skal være i stand til å kjenne igjen. Brukeren har muligheten til å forsterke (Reinforce) eller slette (Remove) en lokasjon.

Brukeren kan også velge å endre presisjonen ("Accuracy"), da forskjellige trådløse kort rapporterer ulike signaler og kan dermed føre til at prototypen ikke virker for alle typer kort. Ved å endre presisjonen støttes flere typer trådløse nettverkskort.

Posisjonering (figur 17 viser at posisjonering er av) startes ved Fil-menyen, som er tilgjengelig uavhengig av om brukeren befinner seg i lokasjons- eller informasjons-brukergrensesnittet.

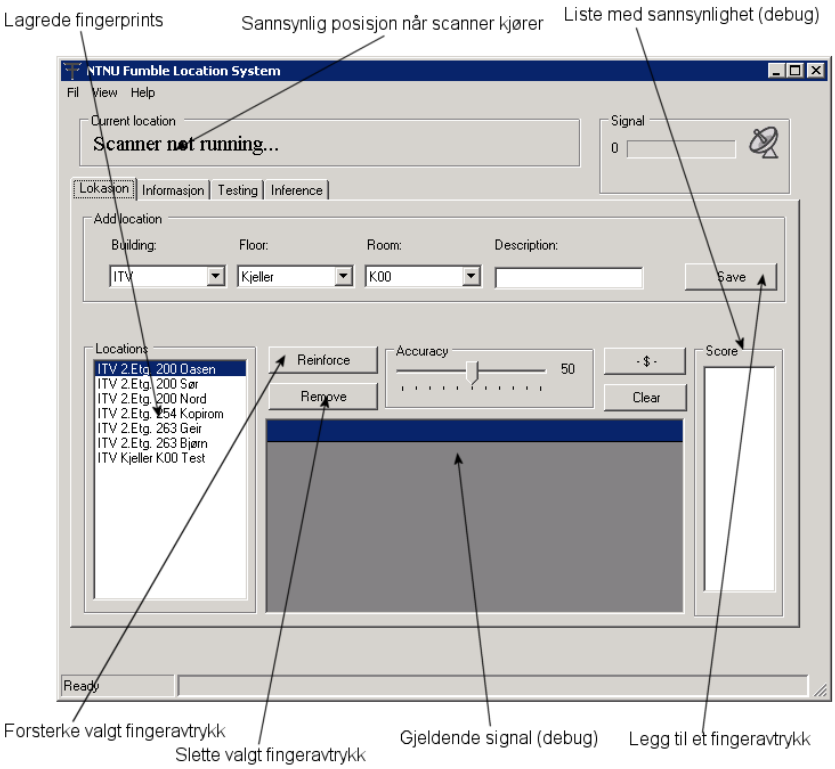
Signalet oppe til høyre gir brukeren viktig tilbakemelding om hva slags signal prototypen mottar, hvilken måling den er i ferd med å utføre og om målingen blir lagret eller forkastet.

4.7.2 Brukergrensesnitt informasjonsdel

Prototypen har et grensesnitt som kombinerer administrative oppgaver og bruker oppgaver. Dette er gjort for å spare tid og arbeid. Et ferdig utviklet program vil skille disse basert på rettigheter, og om funksjonaliteten hører til på klient eller server siden, gjør ikke prototypen et slikt skille. Selv om dette gjorde at det raskt kunne utvikles en prototype, er det lagt opp til at brukergrensesnittet skal være enkelt. Erfaringer gjort viser at programmet som er vanskelig å bruke eller forstyrrer brukeren skaper frustrasjon og kan føre til at brukeren velger vekk funksjonalitet eller enda verre, velger et annet program[30]. Brukergrensesnittet er også vektlagt i [8] som en viktig komponent, dette vil bli forklart nærmere i avsnittene under.

Nåværende implementasjon av Fumble gir brukeren tilgang til både lokasjonsdelen og informasjonsdelen. I et ferdig system vil det her måtte tas en avgjørelse om en bruker skal ha tilgang til lokasjonsinformasjon, og i så fall hva som skal kunne utføres. Dette tar ikke oppgaven stilling til, da det er nødvendig å ha tilgang til

[30]Som feks. "Clippit" i MS Word



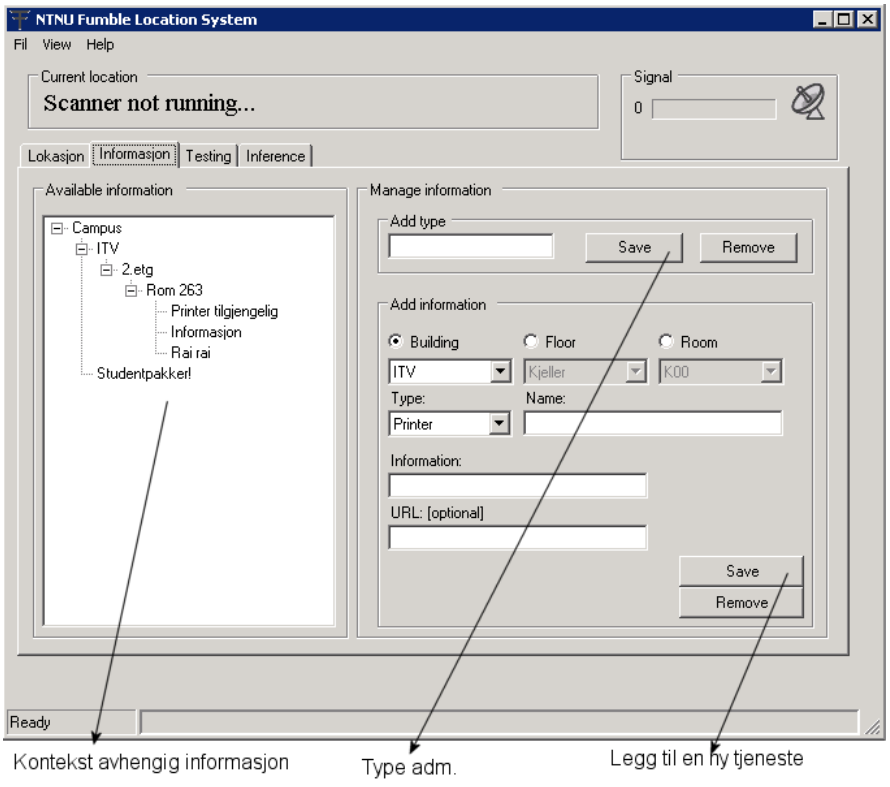
Figur 17: Lokaliseringsbrukergrensesnitt

begge under utvikling av prototypen. Derimot ser oppgaven nærmere på hvordan informasjonsdelen av programmet kommuniserer med brukeren.

Fumble, når ikke i bruk, kjører som et lite ikon i statusfeltet (system tray), se Figur 18. Ikonet er formet som en antenne som også kan tolkes som en F og speilbildet til denne.

Figur 18: Kjørende FUMBLE

Selve informasjonsdelen, se Figur 19, inneholder trevisningen på venstre side, og administrative oppgaver på høyre side. Trevisningen vil til enhver tid inneholde kontekstavhengig informasjon når systemet kjører, og vil være tom når det ikke kjører. Mens systemet kjører kan brukeren klikke på informasjon som kommer opp for å få opp mer data. Finnes det en webside knyttet til informasjonen vil kontrollen overføres til nettleseren. De administrative oppgavene er gjennomgått i avsnitt 4.4, og brukergrensesnittet her består kun av standard tekstbokser og knapper. Det er altså ikke lagt vekt på å et "fancy design", men heller å få frem funksjonaliteten.



Figur 19: Informasjonsbrukergrensesnitt

Informasjonsdelen kan sies å omfatte nettleseren og buddytjenesten. Oppgaven går ikke gjennom hvordan en nettleser fungerer, da det antas at de fleste er kjent med hvordan en slik ser ut. Buddytjenesten gåes heller ikke igjennom, da dette er kun et eksempel på brukergrensesnitt, ikke på funksjonalitet.

Varsling

Et tema som må vurderes nøye er hvordan programmet skal varsle brukeren om en endring i tilgjengelig informasjon. Skjer dette for ofte, vil brukeren oppleve varslingen som forstyrrende, og skjer det for sjeldent vil brukeren finne funksjonen unyttig. I tillegg til å avgjøre hvor ofte en slik varsling skal skje, er det viktig å bestemme hva som skal varsles.

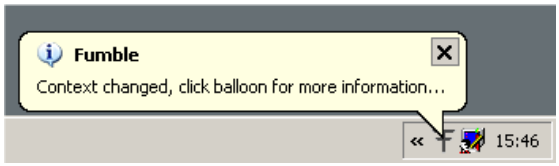
Fumble har implementert et varslingsikon (notification icon) som varsler brukeren ved endring av kontekst, se Figur 20. Denne varsler hver gang det har skjedd en endring i lokasjon og det finnes informasjon tilgjengelig. Dette er ikke ønskelig, som diskutert senere.

Fumble varsler altså når det har skjedd en endring i lokasjon og hvis det finnes informasjon knyttet til denne lokasjonen. Dette gjelder uansett informasjon. Et mål vil være å la være å varsle når det dreier seg om informasjon som kan antas å eksistere, feks. tilgjengelige printere, rom- eller bygnings-informasjon eller globale tjenester som Innsida og It's:Learning. Varsel er ønskelig hvis informasjonen er

begrenset i et visst tidspunkt, eller på annen måte kan være av interesse for brukeren. Eksempel på dette kan være studentpakker, bedriftspresentasjoner, osv. Med andre ord, informasjon som kan tenkes å være nyttig, og som er knyttet til en bestemt lokasjon, eller gjelder kun i et begrenset tidsrom, bør varsles.

Det er også viktig å vurdere hvor lenge det er siden siste varsling. Har det nettopp blitt varslet, vil det da være ønskelig å varsle igjen feks. mindre enn fem minutter etter sist varsling? I et miljø der det skjer ofte endringer vil dette kunne forstyrre en bruker så mye at brukeren velger bort denne funksjonaliteten. Det finnes flere løsninger på dette problemet: 1) Det første som bør gjøres er å la brukeren avgjøre selv om hvor ofte slike varslinger kan skje 2) eller å velge et antall varslinger som samles sammen, før et varsel gis. 3) Det bør kunne lagres profiler som lar brukeren enkelt velge mellom å ikke bli forstyrret eller å la alle varsel slippe igjennom. 4) Arbeidet med å legge inn informasjon må inneholde retningslinjer for hvilke meldinger som kan være av interesse for brukeren og dermed verdt å varsle om.

Det må ikke glemmes at brukeren selv kan se etter tilgjengelig informasjon, uten at programmet nødvendigvis varsler om denne. Hvis en student befinner seg på en forelesning, er det ikke nødvendig å opplyse om at informasjon om kurset er tilgjengelig. Det antas at studenten er klar over dette. Det systemet er ute etter å tilby er hvor denne informasjonen befinner seg, og å kunne tilby denne informasjonen uten at brukeren aktivt må søke etter denne. All informasjon som er tilgjengelig betyr ikke nødvendigvis at brukeren kommer til å benytte seg av denne, men ved behov skal det være aktuelt å få tak i denne.



Figur 20: Varsling

4.8 Oppsummering prototyping

En oppgave som denne krever at det raskt utvikles en fungerende prototype. Dette for å kunne teste flere sider ved designet, illustrere ideer og tidlig få samlet brukererfaringer. Det ble valgt en iterativ prosess av flere grunner, men hovedsaklig fordi det gjorde det mulig å stoppe utviklingen når tidsfristen var nådd. Framdriften var derfor avhengig av at forrige runde var avsluttet. Prototypen har dratt fordel av en slik framgangsmåte, da moduler som var ferdigstilt ikke ble endret hvis dette førte til store forsinkelser. Ulempen med dette er at prototypen inneholder valg som i ettertid har vist seg å være mindre gode, noe som gjenspeiles i resultatet (se kapittel 5).

Dette kapitlet har beskrevet implementasjonen av lokasjonsmodulen og informasjonsmodulen, og hvordan disse er relatert.

5 Testing og evaluering av Fumble

For å avgjøre om Fumble har oppfylt målene definert i avsnitt 1.1.1, er det blitt gjennomført tre tester. Den første testen er beskrevet som "Test av lokasjonsdel" (avsnitt 5.1) og tar for seg mål nummer en: Er posisjonering ved hjelp av trådløse signaler mulig ved NTNU Gløshaugen. Den andre testen er beskrevet som "Test av informasjonsdel" (avsnitt 5.2) og tar for seg mål nummer to: Er det mulig å skape en relasjon mellom romlig informasjon og lokasjon som utnytter sammenhengen mellom informasjonsrommene (information spaces) og de fysiske rommene, og i tillegg utnytter måten lokasjonsdata er organisert på? Den siste testen er beskrevet som "Brukertester" (avsnitt 5.3) og tar for seg mål nummer tre: Kan et kontekstavhengig system være til hjelp for studenter i studiehverdagen?

Til slutt i dette kapitlet oppsummeres og evalueres resultatene i avsnitt 5.4.

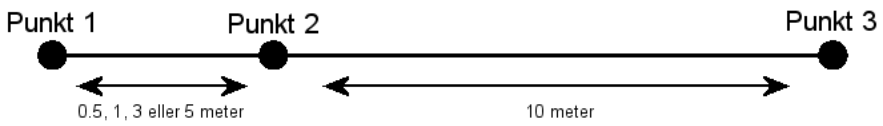
5.1 Lokasjonsdel

Det har tidligere blitt sagt at lokasjonsdelen ble utviklet for å kunne skille mellom rom, og at det i denne oppgaven var et tilfredsstillende resultat. Tidligere prosjekter [8, 21, 24, 19] oppgir nøyaktighet i sine resultater, men akkurat hva som menes med nøyaktighet, og hvordan dette er beregnet er ikke beskrevet. Feks. Nibble oppgir nøyaktighet som: (programmet) er i stand til å skille lokasjoner som ligger ca. 10 fot (ca. 3 meter) fra hverandre, men sier ikke noe om hvordan de har kommet fram til dette.

5.1.1 Test av lokasjonsdel

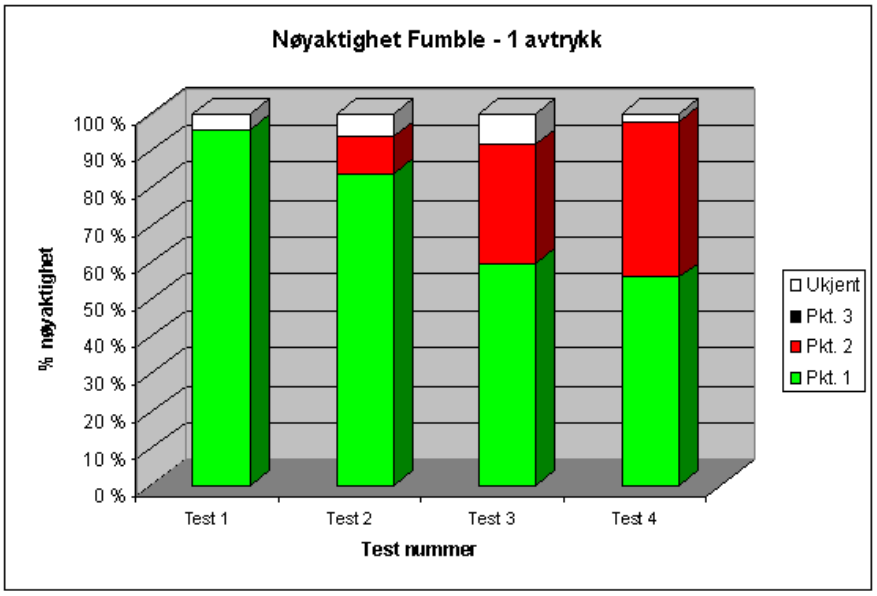
For å kunne teste hvor nøyaktig Fumble er, ble det laget et testoppsett som beskrevet under. Dette ble gjort for å kunne sammenligne Fumble med tilsvarende systemer, og for at andre skal kunne etterprøve resultatene. En slik test er også viktig for å avgjøre hvor stort område en lokasjon dekker i det fysiske rommet.

Testen ble gjennomført på en tilfeldig plass i IT-Vest, i arbeidstiden (kl 15:00), og det antas at nøyaktigheten mellom to punkt er lik for alle punkt som er lagret i systemet. Dette vil i virkeligheten variere pga. bygnings topologi, antall aksesspunkt tilgjengelig, støy osv., men for å teste dette ville det vært nødvendig med bedre tid og grundigere forberedelser.



Figur 21: Avstand

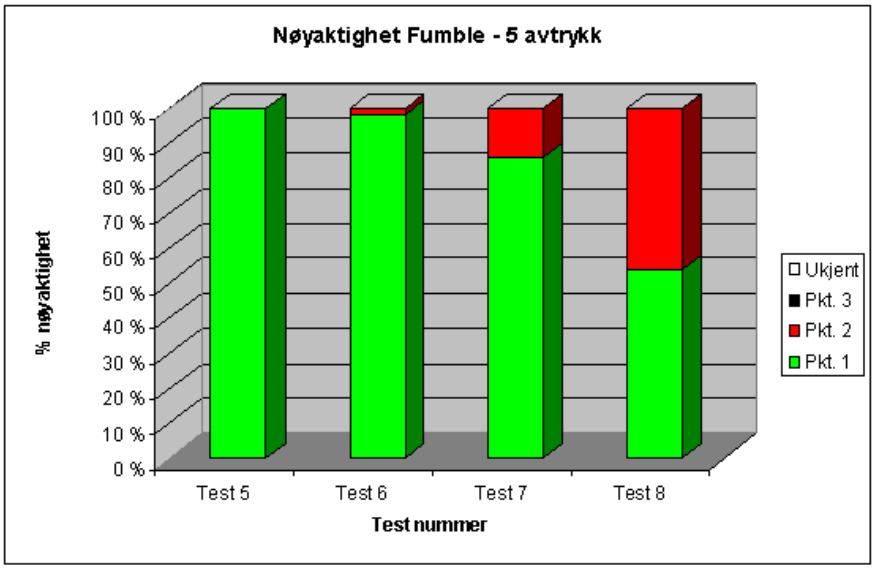
Testoppsettet består av åtte tester, med kun tre lokasjoner, hhv. Punkt 1, Punkt 2 og Punkt 3. Ved å plassere klienten på Punkt 1 registreres hvor mange ganger Fumble oppgir rett lokasjon når avstanden mellom Punkt 1 og Punkt 2 minker. Punkt 3 er kun et testpunkt, som illustrerer en lokasjon som klienten aldri skal befinne seg ved. I Test 1 er avstanden mellom Punkt 1 og Punkt 2 fem meter, i Test 2 tre meter, i Test 3 en meter og Test 4 en halv meter. Disse testene ble gjennomført med kun et avtrykk ved hvert punkt. I testene 5 til 8 er hhv. avstandene også fem, tre, en og en halv meter, men nå er antall avtrykk økt til fem. Se Figur 21. For hver test ble det kjørt 100 målinger, en måling har alternativene Punkt 1, Punkt 2 eller Punkt 3. Punkt 3 er som nevnt et testpunkt som befinner seg innenfor dekningsområdet til tilgjengelige aksesspunkt, og avstanden mellom Punkt 2 og Punkt 3 er hele tiden 10 meter. Punkt 3 skal dermed ikke foreslås som lokasjon i noen av testene, hvis dette skjer inneholder lokasjonsdelen feil. Resultatene fra testene er presentert i Figur 22 og Figur 23.



Figur 22: Nøyaktighet Fumble - et avtrykk

Forklaring til Figur 22 og 23:

- **Test 1:** Fem meter mellom Punkt 1 og Punkt 2, et avtrykk. Resultatene viser at Fumble foreslo Punkt 1 som rett lokasjon i 96 prosent av tilfellene, mens i de resterende 4 prosentene ikke kunne foreslå en nøyaktig posisjon.
- **Test 2:** Tre meter mellom Punkt 1 og Punkt 2, et avtrykk. Resultatene viser at Fumble foreslo Punkt 1 som rett lokasjon i 84 prosten av tilfellene, mens i 10 prosent foreslo Punkt 2, og i 6 prosent ikke kunne foreslå noen nøyaktig posisjon.
- **Test 3:** En meter mellom Punkt 1 og Punkt 2, et avtrykk. Resultatene viser at Fumble foreslo Punkt 1 som rett lokasjon i 60 prosten av tilfellene, mens



Figur 23: Nøyaktighet Fumble - fem avtrykk

i 32 prosent foreslo Punkt 2, og i 8 prosent ikke kunne foreslå noen nøyaktig posisjon.

- **Test 4:** En halv meter mellom Punkt 1 og Punkt 2, et avtrykk. Resultatene viser at Fumble foreslo Punkt 1 som rett lokasjon i 56 prosent av tilfellene, mens i 42 prosent foreslo Punkt 2, og i 2 prosent ikke kunne foreslå noen nøyaktig posisjon.
- **Test 5:** Fem meter mellom Punkt 1 og Punkt 2, fem avtrykk. Resultatene viser at Fumble foreslo Punkt 1 som rett lokasjon i 100 prosent av tilfellene.
- **Test 6:** Tre meter mellom Punkt 1 og Punkt 2, fem avtrykk. Resultatene viser at Fumble foreslo Punkt 1 som rett lokasjon i 98 prosent av tilfellene, mens i 2 prosent foreslo Punkt 2.
- **Test 7:** En meter mellom Punkt 1 og Punkt 2, fem avtrykk. Resultatene viser at Fumble foreslo Punkt 1 som rett lokasjon i 86 prosent av tilfellene, mens i 14 prosent foreslo Punkt 2.
- **Test 8:** En halv meter mellom Punkt 1 og Punkt 2, fem avtrykk. Resultatene viser at Fumble foreslo Punkt 1 som rett lokasjon i 54 prosent av tilfellene, mens 46 prosent foreslo Punkt 2.

5.1.2 Kommentar til lokasjonsdel

Det gode resultatet kan skyldes spesielle forhold som antall aksesspunkt tilgjengelig. Ved Punkt 1 og Punkt 2 er det sju aksesspunkt tilgjengelig, og ved Punkt 3 hele åtte. Dette gjør at det skal mye til for at Punkt 3 slår inn, da et ekstra

aksesspunkt utgjør en stor forskjell. En annen faktor som kan være med å påvirke resultatet er hva slags terskel som er valgt for å avgjøre om en måling stemmer med et avtrykk i databasen. Ved å øke denne terskelen kan man forbedre sjansene for å finne en lokasjon, men samtidig øker dette faren for å foreslå feil lokasjon. Legg merke til at Punkt 3 aldri ble foreslått som lokasjon. Denne terskelen er i Fumble satt til 60. Dette vil si at det kan være en forskjell på 60 poeng og likevel vil Fumble kunne foreslå en lokasjon, såfremt dette er den med minst score. Når Fumble ikke er i stand til å foreslå en posisjon, er det ingen lokasjoner som har mindre forskjell enn 60 beregnet ut i fra det lagrede signalet. Merk at det i Fumble er mulig å endre denne terskelen til en verdi mellom 0 og 100.

Resultatene viser at det er gunstig å ta flere avtrykk ved samme lokasjon, dette for å både øke nøyaktigheten og for å unngå at Fumble ikke er stand til å foreslå en lokasjon. Ser vi på Figur 22 går det fram av alle testene at Fumble i noen tilfeller ikke er stand til å foreslå en lokasjon. I Figur 23 derimot var Fumble alltid i stand til å foreslå en lokasjon, selv om denne ikke alltid var korrekt. Grunnen til dette kan være at forskjellige avtrykk på forskjellige lokasjoner kan være svært like da faktorer som orientering, personer i rommet, dører som åpnes osv., kan være med på å forstyrre signalene. Merk at når avstanden er mindre enn fem meter, er det ikke alltid kritisk at systemet tar feil, når disse punktene befinner seg i samme rom.

Resultatene viser også at nøyaktigheten til Fumble ligger på omtrent tre meter mellom to punkt, som er det samme som Nibble. Ved en meter viser Fumble tegn til unøyaktighet, og ved en halv meter er det tilnærmet 50\50 prosent sjanse for hvilket punkt som blir valgt.

5.2 Informasjonsdel

Et kontekstavhengig system reagerer som nevnt tidligere på en endring av kontekst, i dette tilfellet en endring i lokasjon. Når systemet oppdager en endring i lokasjon, kjøres funksjonen FindContext (se Appendix F.2) for å oppdatere tilgjengelig informasjon basert på den nye lokasjonen. Denne funksjonen er kritisk for systemet, og hvis denne feiler eller gir et galt resultat vil dette gi store konsekvenser. Derfor er denne delen av systemet testet for å se hvordan systemet fungerer ved NTNU Gløshaugen.

For at informasjondelen skal fungere er det to forutsetninger som må oppfylles. Den første er at posisjonering må være mulig, men et unntak her er hvis systemet ikke klarer å finne en posisjon, skal likevel global informasjon presenteres. For det andre må systemet ha en tilgjengelig databærer, men også her finnes det unntak feks. ved at informasjon lagres lokal når en databærer er tilgjengelig, og dermed gjør systemet i stand til å presentere informasjon selv uten en tilgjengelig databærer. Først er det valgt å se på når begge forutsetningene er oppfylt. I avsnitt 5.2.2 diskuteres systemet når disse forutsetningene ikke er møtt.

5.2.1 Test av informasjonsdel

Kjernen i systemet er informasjonsdelen, og det er derfor kritisk for systemet som helhet at denne virker tilfredsstillende. Derfor ble det laget en test som tester hvordan systemet klarer seg under normale forhold ved NTNU Gløshaugen. Normale forhold er når både posisjonering er mulig og det finnes en tilgjengelig databærer. Til slutt diskuteres også hvordan systemet vil kunne oppføre seg under spesielle forhold.

Først ble flere lokasjoner lagt inn i systemet. Så ble det ved noen utvalgte lokasjoner knyttet relevant informasjon som skal være tilgjengelig hvis brukeren befinner seg innenfor angitt område. De fleste lokasjonene inneholdt ingen relevant informasjon. De kan dermed kun gi brukeren informasjon om hvor denne befinner seg. Disse lokasjonene er ikke med i denne testen, da det ble bevist i avsnitt 5.1.1 at dette virker. I et ferdig system kan lokasjon brukes som informasjon til å presentere brukeren med veiforklaringer, men dette er ikke implementert.

Lokasjonene som inneholder informasjon er:

- Rommene:
  - ITV122 - Informasjon om rom
  - ITV200 - Informasjon om rom\gang
  - ITV254 - Informasjon om rom
  - ITV254 - Tilgjengelig printer
  - ITV263 - Informasjon om rom
- Etasjene:
  - 1.etg. ITV - Gir brukeren informasjon om at studentekspedisjonen befinner seg i denne etasjen
  - 2.etg. ITV - Tilgjengelig fargeprinter for hele 2.etg ITV
- Bygningene:
  - ITV - Informasjon om bygningen
  - ITØ - Informasjon om bygningen
- I tillegg finnes det global informasjon:
  - Studentpakker
  - Brann øvelse
  - Innsida (Se avsnitt 2.1.2.)

Når dette var på plass ble en bærbar pc brukt for å teste om systemet fant rett informasjon ved hver lokasjon. Ved å gå rundt i bygningene ITV, ITØ og ITS og registrere hva slags informasjon som ble presentert ved lokasjonene over, kan man analysere disse for å se om systemet virker. Hver lokasjon ble besøkt tre ganger, og det ble valgt forskjellige ruter gjennom byggene. Resultatene er oppsummert i tabell 16.

Informasjon	**Fast**	**Rett lok.**	**Feil lok.**	**Kommentar**
Rom ITV122	0	3	0	Tilgjengelig i rom 100 ITV Info. om rom.
Rom ITV263	0	3	0	Tilgjengelig i rom 263 ITV Info om rom.
Rom ITV263	0	3	0	Tilgjengelig i rom 263 ITV Tilgjengelig printer
Rom ITV254	0	3	0	Tilgjengelig i rom 254 ITV Info. om rom.
Rom ITV200	0	3	0	Tilgjengelig i rom 200 ITV Info. om rom.
Etasje Printer itv254farge	0	3	0	Tilgjengelig i 2.etg. ITV Info. om tilgj. printer.
Etasje Stud. ekspedisjon	0	3	0	Tilgjengelig i 1.etg. ITV Info. om stud. ekspedisjon.
Bygning ITV	0	3	0	Tilgjengelig i ITV Info. om bygningen.
Bygning ITØ	0	3	0	Tilgjengelig i ITØ Info. om bygningen.
Global Studentpakker	3	0	0	Tilgjengelig uansett lok. Info. om studentpakker
Global Innsida	3	0	0	Tilgjengelig uansett lok. Meld. fra Innsida.
Global Brann øvelse	3	0	0	Tilgjengelig uansett lok. Info. om brannøvelse

Tabell 16: Tabell Informasjonsdel

5.2.2 Kommentar til informasjonsdel

Resultatet tar ikke hensyn til når systemet deduserer feil posisjon. Dette er fordi dette ikke er en feil i informasjonsdelen, men i lokasjonsdelen. Et eksempel er: Hvis man befinner seg i ITV rom 263, mens systemet tror rett lokasjon er ITV rom 254, vil informasjonen som skulle vært tilgjengelig i ITV254 bli presentert i ITV263. Dette kan skje hvis en lokasjon er undersamplet (for dårlig forsterket), eller aksesspunkt faller ut. Lite har blitt gjort for å prøve å forbedre dette, da dette er utenfor oppgavens problemstilling.

Resultatene viser at systemet presenterer rett informasjon ved forskjellige lokasjoner. Det er derfor bevist at relasjonen mellom informasjon og lokasjon, slik den er foreslått i Fumble, virker. Legg også merke til at informasjonen kommer korrekt opp selv om denne er knyttet til etasje eller bygning i lokasjonshierarkiet. Dette gjør at systemet utnytter måten lokasjonsdata er lagret på, og slik finner den all tilgjengelig informasjon. Det er dermed opp til informasjonsdelen å traversere (lokasjons)treet opp til rotnoden for å undersøke om det finnes informasjon knyttet til andre nivåer.

Som nevnt innledningsvis skal systemet presentere global informasjon selv om det ikke er mulig å dedusere en posisjon. Å kunne dedusere posisjon er det første kriteriet for at systemet skal fungere, unntaket er altså global informasjon. Dette viste seg å være tilfelle, da Fumble er implementert slik at global informasjon ikke er relatert til lokasjon. Dette viser at Fumble ikke er avhengig av en posisjon for å kunne presentere global informasjon.

Det andre kriteriet er at en tilgjengelig databærer må være tilgjengelig. Her finnes det også unntak. Fumble kan ved oppstart laste ned all tilgjengelig informasjon, for så å bruke en lokal kopi av dataene for senere å presentere informasjon. Dette krever at en databærer er tilgjengelig ved oppstart, men kan være utilgjengelig ved et senere tidspunkt. Ulempen er at kritisk informasjon[31] da ikke kommer igjennom. Fumble er ikke implementert med denne funksjonaliteten.

Testen avslørte imidlertid at informasjon som er gjort tilgjengelig ved en posisjon, henger igjen når systemet ikke er i stand til å lokalisere seg. Et eksempel på dette er: Brukeren befinner seg i rom 263, og får presentert informasjon som er tilgjengelig ved denne lokasjonen. Beveger brukeren seg bort fra dette rommet, og systemet ikke er i stand til å finne en posisjon, vil informasjonen fra rom 263 henge igjen helt til en ny posisjon kan deduseres. Dette er noe som ikke ble tenkt igjennom under implementasjonen. Det kan diskuteres om dette er en feil i systemet, eller et valg som er tatt. Som nevnt tidligere skal global informasjon være tilgjengelig uansett, spørsmålet er om kontekstavhengig informasjon skal gjøres utilgjengelig når systemet ikke er i stand til å dedusere en posisjon.

5.3 Brukertester

For å besvare det siste spørsmålet i kravspesifikasjonen, "Kan et kontekstavhengig system være til hjelp for studenter i studiehverdagen?", ble systemet vist til noen studenter og så ble de bedt om å besvare et spørreskjema.

Det ble valgt ut studenter med forskjellig bakgrunn, men med generell dataerfaring. Dette fordi Fumble er en prototype der bruken ikke nødvendigvis er logisk. Det var også studenter som hadde tid og lyst til å gjennomføre en slik test, og som hadde bærbare pc-er med trådløse nettverkskort.

[31] Kritisk informasjon: Meldinger som brukere har nytte av å lese så fort de blir tilgjengelig.

Testen ble gjennomført ved at det ble foretatt en gjennomgang ("walkthrough") med brukeren. Hvordan testene ble gjennomført varierte veldig fra brukertest til brukertest, fordi den første gjennomgangen avdekket uforutsette sider ved testen, og fordi rom ikke alltid var tilgjengelig under testingen. Derfor ble det valgt forskjellige ruter ved hver test, selv om det her ville vært hensiktsmessig å velge samme rute for å oppnå like resultater. En oversikt over de punktene som i utgangspunktet var tenkt som testdata finnes i Figur 24. Hva slags informasjon som skulle være tilgjengelig finnes i Tabell 18.

For å gjøre det enklere å gjennomføre testen, ble informasjon lagt inn på forskjellige lokasjoner, i motsetning til at informasjon ble tilgjengelig ved et gitt tidspunkt. Eksempel: En "ny" melding fra Innsida vil dukke opp ved plass 6, og en "ny" E-post vil dukke opp ved plass 8.

Underveis ble det notert kommentarer fra brukeren, og hvis nødvendig ble brukeren veiledet om dette var nødvendig. Testen prøvde også i utgangspunktet å benytte seg av brukerens personlige bærbare pc. Fumble ble altså installert på flere bærbare pc-er, mens dataene brukt var lagt inn fra før vha. en bærbar pc som har blitt brukt til testing gjennom hele oppgaven. Dette for å teste om systemet fungerer med forskjellige trådløse nettverkskort og operativsystemer. Hvis dette ikke var mulig ble pc-en som var brukt til testing benyttet. Når dette ble gjort er det opplyst om i de testene det gjelder.

Gjennomgangen forsøker å vise testpersonene potensialet til et kontekstavhengig system. Det er spørsmålet: "Ville du brukt et slikt system?" som er avgjørende, sammen med eventuelt kommentarer. Dette vil være et subjektivt svar, men det er det testen er ute etter. Testen prøver også å kartlegge noen av vanene til studentene, ved å spørre om hvilke systemer de benytter, og hvor ofte.

Spørreskjemaet som ble brukt etter gjennomgangen er listet i Tabell 17.

Spørsmål	**Alternativer**		
Hadde du benyttet "buddy"-tjenesten?	Ja	Nei	Vet ikke
- Hvis nei, hva slags kriterier måtte ha vært oppfylt hvis du skulle?	Kommentar.		
Leter du etter informasjon du vet finnes, uten å finne det?	Ja	Nei	Aldri
Hvor mange systemer logger du deg inn i hver dag?			Antall.
Fungerte koblingen mellom sted og informasjon?	Kommentar.		
Logger du deg inn i It's:Learning via Innsida?	Ja	Nei	Vet ikke
Følte du at du kunne stole på systemet?	Ja	Nei	Vet ikke
- Hvis nei, hvorfor ikke?	Kommentar.		
Tror du et slikt system kan dekke informasjonsbehovet for studenter ved NTNU Gløshaugen	Ja	Nei	Vet ikke
- Hvis nei, hvorfor ikke?	Kommentar.		
Har du noengang lurt på hvor du skal ha forelesning?	Ja	Nei	Vet ikke
Hvor får du hovedsaklig tak i informasjon om kurs\gjesteforelesninger\begivenheter ol.?	Kommentar.		
Har du noengang gått glipp av et møte\kurs \pga. av manglende informasjon?	Ja	Nei	Vet ikke
Mener du at et ferdig system hadde vært nyttig ved NTNU Gløshaugen?	Ja	Nei	Vet ikke
Hvor mange ganger logger du deg inn i Innsida pr. dag?			Antall.
Ville du brukt et slikt system?	Ja	Nei	Vet ikke
- Hvis nei, hvorfor ikke?	Kommentar.		
Andre kommentarer?	Kommentar.		

Tabell 17: Tabell Spørreskjema

5.3.1 Innhold i testdatabasen

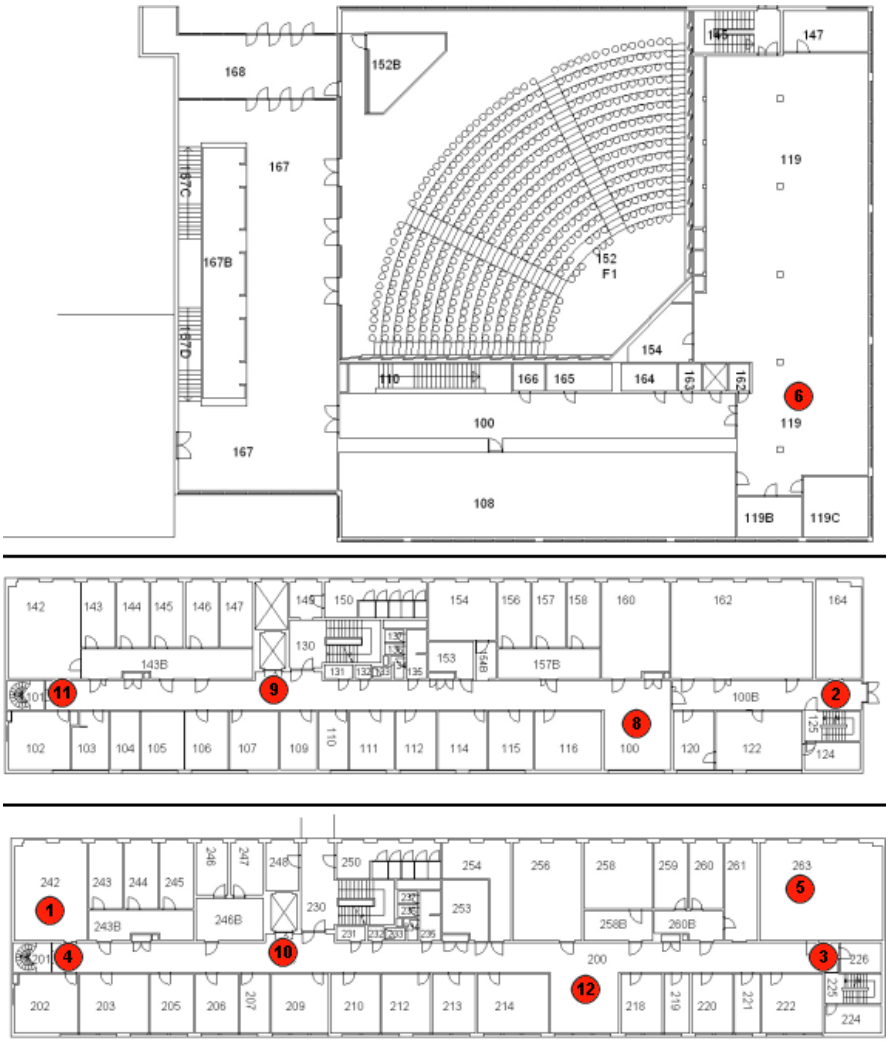
Det ble knyttet informasjon til tolv forskjellige lokasjoner. Merk at en lokasjon kan være en bygning, og dermed gjelde for hele bygningen, eller et rom, og dermed gjelde bare for det angitte rommet. Det er også lagt vekt på å variere typen informasjon tilgjengelig, som feks. fra bygningsinformasjon til tilgjengelige tjenester til tilgjengelige systemer.

ID	Navn	Beskrivelse	Dekning
1	Møterom ITV 242	Inneholder: Videokanon Overhead Whiteboard Surround sound Telefon 16 sitteplasser Kaffe-krok	Rom
2	Bygning	Gruppe for Informasjonssystemer Gruppe for systemarbeid og MMI Gruppe for systemutvikling Resepsjon/Studentekspedisjon	Etasje
3	Bygning	Gruppe for databaseteknikk Gruppe for informasjonsforvaltning	Etasje
4	Ola	Ola lurer på om du kommer til Sito.	Rom
5	Arb. rom	Arbeidsrom for masterstudenter.	Rom
6	Innsida	Du har 1 ulest melding.	Rom
7	Studentpakker	Hent din gratis studentpakke!	Global
8	E-mail	Du har 1 ny E-mail!	Etasje
9	Heis	Heis er tilgjengelig.	Etasje
10	Heis	Heis er tilgjengelig.	Etasje
11	Geir	Din venn Geir er i nærheten.	Etasje
12	Oasen	Myldrerom finnes i denne etasjen.	Room

Tabell 18: Tabell Informasjon

5.3.2 Kart over tilgjengelig informasjon

Informasjonen ble knyttet til faste lokasjoner, og disse er vist i figuren under.



Figur 24: Kart over tilgjengelig informasjon

5.3.3 Brukertest 1

Bruker 1 tar en bachelor i kjemi ved NTNU på Gløshaugen. Gjennomgangen ble foretatt på en søndag kl. 18.00, 16. April 2006 og tok ca. 2 timer.

Brukeren var kjent med systemet fra en tidligere versjon, og benyttet sin egen bærbare pc i testen. Pc-en er en Dell Latitude D505 med et Intel(R) PRO\Wireless LAN 2100 3A trådløst nettverkskort. Pc-en kjører Windows XP.

Gjennomgang

Det viste seg at det ikke var mulig for prototypen å benytte seg av lokasjonsdataene

lagt inn på forhånd. Tolkningen av signalene gjorde at forskjellen i signalet ble for stor og prototypen visste dermed ikke hvor den var uansett plassering. Grunnen til dette er at Fumble har mulighet for å stille grenseverdien for hva som gir en korrekt lokasjon mellom 0 og 100. Brukerens bærbare pc anga at gjeldende posisjon hadde en verdi på mellom 160-200. Hadde Fumble vært implementert slik at grenseverdien kunne stilles mellom 0 og 500 hadde prototypen fungert. Løsningen ble å forsterke hver plass en gang ved hver av plassene 1-12. Dette var nok for å gjennomføre testen. Det er verdt å nevne at selv med bare en forsterkning anga systemet rett lokasjon med en verdi på under 20, noe som er et godt resultat. Skalaen går fra 0, som er en eksakt match, til uendelig dårlig, som er en lite sannsynlig match.

Når systemet klarte å posisjonere seg selv, begynte testen i 1 etg. ITV, for så å besøke ITS og tilslutt 2.etg ITV. Ved hver plass kom informasjon korrekt opp, og systemet varslet om dette. Brukeren likte tanken på at systemet kunne varsle om nye meldinger på Innsida og ny E-post. Også det at det var mulig å søke opp "buddies" var av interesse. Brukeren påpekte at det var vanskelig å stole på systemet når man måtte forsterke plassene for å få det til å virke, og også at måten systemet varslet på var irriterende til tider.

Sitat:

> Jeg oppfattet "Rom 200" som en informasjonsdel som jeg ikke kunne forklare hvor kom fra under testen.

Dette gjorde at både bruker og tester ble usikker på om systemet inneholdt en feil. Grunnen til dette er gjennomgått i avsnitt 5.3.6.

Svarene på spørreskjemaet er listet i Tabell 19

Spørsmål	**Alternativer**		
Hadde du benyttet "buddy"-tjenesten?	Ja	~~Nei~~	~~Vet ikke~~
- Hvis nei, hva slags kriterier måtte ha vært oppfylt hvis du skulle?			
Leter du etter informasjon du vet finnes, uten å finne det?	Ja, ofte	~~Nei~~	~~Aldri~~
Hvor mange systemer logger du deg inn i hver dag?			2
Fungerte koblingen mellom sted og informasjon?	Kommentar 1. (Se under.)		
Logger du deg inn i It's:Learning via Innsida?	Ja	~~Nei~~	~~Vet ikke~~
Følte du at du kunne stole på systemet?	~~Ja~~	Nei	~~Vet ikke~~
- Hvis nei, hvorfor ikke?	Kommentar 2. Se under.		
Tror du et slikt system kan dekke informasjonsbehovet for studenter ved NTNU Gløshaugen	Ja	~~Nei~~	~~Vet ikke~~
- Hvis nei, hvorfor ikke?			
Har du noengang lurt på hvor du skal ha forelesning?	Ja	~~Nei~~	~~Vet ikke~~
Hvor får du hovedsaklig tak i informasjon om kurs\gjesteforelesninger\begivenheter ol.?	Innsida.		
Har du noengang gått glipp av et møte\kurs \pga. av manglende informasjon?	Ja	~~Nei~~	~~Vet ikke~~
Mener du at et ferdig system hadde vært nyttig ved NTNU Gløshaugen?	Ja	~~Nei~~	~~Vet ikke~~
Hvor mange ganger logger du deg inn i Innsida pr. dag?			4-5
Ville du brukt et slikt system?	Ja	~~Nei~~	~~Vet ikke~~
- Hvis nei, hvorfor ikke?			
Andre kommentarer?	Kommentar 3. (Se under.)		

Tabell 19: Tabell Svarskjema Bruker 1

Kommentar 1: Siden det skulle så små forandringer i posisjon til før systemet kom opp med annen info (kjente seg ikke igjen osv) kan det være vanskelig å få studentene til å bruke det. Da må du nesten opprette små soner, der alle posisjoner innenfor denne sonen er heis 2.etg osv.

Kommentar 2: Vanskelig å svare på, da det var noe rart med informasjonen ("Rom 200"), men den informasjonen som kom korrekt opp viste seg å være viktig.

Kommentar 3: Svaret på om jeg kunne stole på systemet ville blitt annerledes om det allerede lå korrekt data i systemet slik at det viste rett plass med en gang. Å ha en kartløsning der du ser hvor du skal eller hvor en "buddy" befinner seg hadde vært en ide.

5.3.4 Brukertest 2

Bruker 2 er utdannet drifter, og jobber som drifter ved Trondheim Folkebibliotek. Gjennomgangen ble foretatt på en lørdag kl. 13.00, 21. April 2006 og tok ca. 1 time.

Brukeren var ikke kjent med systemet, og skulle i utgangspunktet benytte sin egen bærbare pc i testen. Det viste seg imidlertid at selv om det trådløse nettverkskortet var støttet i NetStumbler, ville ikke NetStumbler kjøre scriptet som er nødvendig for å overføre signaldata til Fumble sin database. Det var ikke mulig å finne ut hvorfor dette skjedde før testen ble gjennomført. Dette siden vi under testen hadde liten tid tilgjengelig, ble test pc-en benyttet istedet. Pc-en er en Dell Latitude Cpi D300XT med et Avaya Wireless World Card Gold. Pc-en kjører Windows ME.

Gjennomgang

Merk! Det viste seg at lokasjon 1 ikke var tilgjengelig under testen, grunnet at dette rommet var låst. Derfor ble rom 464 lagt inn og tilsvarende informasjon knyttet til dette.

Ved å bruke samme pc som har lagret lokasjonsdata var ikke posisjonering lenger et problem, slik som i brukertest 1. Testen startet i 1.etg i ITV, for så å besøke hallen i ITS før vi fortsatt opp i 2.etg og avsluttet i 4.etg i ITV. Gjennomgangen foregikk uten nevneverdige problemer. Brukeren forsto raskt hvordan systemet fungerte, men påpekte at brukergrensesnittet kunne ha vært bedre.

Svarene på spørreskjemaet er listet i Tabell 20

Kommentar 1: Ja. Ser for meg at informasjonen blir presentert på samme måte som feks. `http://www.a9.com`, ved at brukeren selv kan legge til og fjerne informasjonsmoduler på ei og samme side.

Kommentar 2: Kunne det vært mulig å få en oversikt over ledige leseplasser\datasaler? Kanskje til og med reservere plasser? Programmet burde også støtte at brukeren kan legge inn egne private notater og linker. Brukergrensesnittet er

Spørsmål	**Alternativer**		
Hadde du benyttet "buddy"-tjenesten?	Ja	~~Nei~~	~~Vet ikke~~
- Hvis nei, hva slags kriterier måtte ha vært oppfylt hvis du skulle?			
Leter du etter informasjon du vet finnes, uten å finne det?	Ja, ofte	~~Nei, sjelden~~	~~Aldri~~
Hvor mange systemer logger du deg inn i hver dag?			3
Fungerte koblingen mellom sted og informasjon?	Kommentar 1		
Logger du deg inn i It's:Learning via Innsida?	Ja	~~Nei~~	~~Vet ikke~~
Følte du at du kunne stole på systemet?	Ja, men det bruker for lang tid.	~~Nei~~	~~Vet ikke~~
- Hvis nei, hvorfor ikke?	Kommentar.		
Tror du et slikt system kan dekke informasjonsbehovet for studenter ved NTNU Gløshaugen	Ja	~~Nei~~	~~Vet ikke~~
- Hvis nei, hvorfor ikke?			
Har du noengang lurt på hvor du skal ha forelesning?	Ja	~~Nei~~	~~Vet ikke~~
Hvor får du hovedsaklig tak i informasjon om kurs\gjesteforelesninger\begivenheter ol.?	Stripa, snakker med personer som står der. Innsida er uviktig.		
Har du noengang gått glipp av et møte\kurs \pga. av manglende informasjon?	~~Ja~~	Nei	~~Vet ikke~~
Mener du at et ferdig system hadde vært nyttig ved NTNU Gløshaugen?	Ja	~~Nei~~	~~Vet ikke~~
Hvor mange ganger logger du deg inn i Innsida pr. dag?			0
Ville du brukt et slikt system?	Ja	~~Nei~~	~~Vet ikke~~
- Hvis nei, hvorfor ikke?			
Andre kommentarer?	Kommentar 2		

Tabell 20: Tabell Svarskjema Bruker 2

elendig. Selv om dette er en prototype, burde det også være en prototype på hvordan brukergrensesnittet kunne se ut.

Det må både være lett å ta vekk informasjon som ikke er relevant, og også legge til informasjon. Det må legges opp til et multifunksjonelt brukergrensesnitt. Godt program, men noe "hjemmesnekret".

5.3.5 Brukertest 3

Bruker 3 tar en bachelor i mediavitenskap ved NTNU Dragvoll. Gjennomgangen ble foretatt på en fredag kl. 13.00, 20. April 2006 og tok ca. 1 time.

Brukeren var kjent med systemet fra en tidligere versjon, men det var ikke mulig å benytte brukerens pc da nettverkskortet i denne ikke var støttet av NetStumbler. Derfor ble test pc-en benyttet under testen. Pc-en er en Dell Latitude Cpi D300XT med et Avaya Wireless World Card Gold. Pc-en kjører Windows ME.

Gjennomgang

Merk! Det viste seg at lokasjon 1 ikke var tilgjengelig under testen, grunnet et møte. Derfor ble rom 464 lagt inn og tilsvarende informasjon knyttet til dette.

Som i Brukertest 2 var ikke posisjonering lenger noe problem når samme bærbare pc har registrert signaldataene brukes under testingen. Testen startet i 1.etg i ITV, så besøkte vi hallen i ITS før vi fortsatt opp i 2.etg og avsluttet i 4.etg i ITV. Gjennomgangen foregikk uten nevneverdige problemer. Brukeren virket imponert over systemet, og likte spesielt tanken på hvordan et slikt system kunne knyttet opp mot feks. Innsida og It's:Learning.

Svarene på spørreskjemaet er listet i Tabell 21

Kommentar 1: Til spørsmålet "Tror du et slikt system kan dekke informasjonsbehovet for studenter ved NTNU Gløshaugen?": Ordet "dekke" er et vidt begrep. Systemet vil utvilsomt hjelpe studenter da forskjellige studenter har svært forskjellig informasjonsbehov, men å dekke studenters informasjonsbehov er å ta i litt hardt.

Bruker Innsida fordi det er link til It's:Learning.

Er det mulig å foreslå hvor lang tid det vil ta å bevege seg fra en plass til en annen? Feks. kan systemet si at det vil ta deg ca. 10 minutter å gå til D5?

Innsida er ikke noe bra, lite relevant. Ser at dette systemet ikke vil endre faglig innhold som er lagt ut, bare presentere faglig innhold som kan være relevant for brukeren. Likevel vil dette være en stor forbedring.

Spørsmål	**Alternativer**		
Hadde du benyttet "buddy"-tjenesten?	Ja	~~Nei~~	~~Vet ikke~~
- Hvis nei, hva slags kriterier måtte ha vært oppfylt hvis du skulle?			
Leter du etter informasjon du vet finnes, uten å finne det?	Ja, av og til	~~Nei~~	~~Aldri~~
Hvor mange systemer logger du deg inn i hver dag?			4
Fungerte koblingen mellom sted og informasjon?	Ja! Imponert!		
Logger du deg inn i It's:Learning via Innsida?	Ja	~~Nei~~	~~Vet ikke~~
Følte du at du kunne stole på systemet?	Ja	~~Nei~~	~~Vet ikke~~
- Hvis nei, hvorfor ikke?			
Tror du et slikt system kan dekke informasjonsbehovet for studenter ved NTNU Gløshaugen	Ja	~~Nei~~	~~Vet ikke~~
- Hvis nei, hvorfor ikke?			
Har du noengang lurt på hvor du skal ha forelesning?	Ja	~~Nei~~	~~Vet ikke~~
Hvor får du hovedsaklig tak i informasjon om kurs\gjesteforelesninger\begivenheter ol.?	It's:Learning		
Har du noengang gått glipp av et møte\kurs \pga. av manglende informasjon?	Ja	~~Nei~~	~~Vet ikke~~
Mener du at et ferdig system hadde vært nyttig ved NTNU Gløshaugen?	Ja	~~Nei~~	~~Vet ikke~~
Hvor mange ganger logger du deg inn i It's:Learning pr. dag?			1-2
Ville du brukt et slikt system?	Ja	~~Nei~~	~~Vet ikke~~
- Hvis nei, hvorfor ikke?			
Andre kommentarer?	Kommentar 1		

Tabell 21: Tabell Svarskjema Bruker 3

5.3.6 Kommentar til brukertester

Brukertest 1: Brukertest 1 var den eneste testen som ble gjennomført med brukerens private PC, og dette forandret hvordan systemet ble mottatt. For å få gjennomført testen måtte hver lokasjon forsterkes, og dette ble oppfattet som forstyrrende for testen. Grunnen til at PC-en ikke klarte å nyttegjøre seg av eksisterende lokasjonsdata er at systemet har ikke blitt testet med flere PC-er under utviklingsfasen. Det er dermed ikke gjort noen forsøk på å få prototypen til å være kompatibel med flest mulige pc-er annet enn at prototypen skal i teorien være kompatibel med de trådløse nettverkskortene som NetStumbler støtter.

Testen bar også preg av at dette var den første gjennomgangen med en reell bruker. Feks. var det vanskelig å forstå hvorfor "Rom 200" kom opp som informasjon ved et tidspunkt. Det viste seg i ettertid at dette var lokasjonsinformasjon (vi befant oss i "Rom 200") og ikke informasjon knyttet til en lokasjon. Denne feiltolkningen av prototypen skyldes nok både en stresset situasjon, dårlig brukergrensesnitt og at det var den første gjennomføringen. "Rom 200" var altså ikke en feil i systemet, men en feiltolkning av informasjonen som ble presentert.

Slik systemet er implementert nå varsler den brukeren hver gang den oppdager en endring i kontekst vha. en informasjonsballong (se avsnitt 4.7.2). Dette oppfattet brukeren som irriterende. I et ferdig system vil det være mulig å endre hvordan systemet skal varsle, hvordan og hvor ofte. Men siden profiler ikke er implementert i prototypen er dette ikke mulig. Likevel, muligheten for å klikke på ballongen for å få opp hva slags informasjon som var gjort tilgjengelig var noe brukeren benyttet seg av.

Brukeren foreslo også en løsning der han så for seg en løsning med kart, der interesseområder var avmerket. Feks. kunne en se hvor "Buddies" kunne befinne seg, dette uten at dette var nevnt som en funksjonalitet. Dette er helt klart en løsning som må studeres nøyere i en videre utvikling av prototypen. Brukeren forsto også at systemet ville ha oppført seg annerledes hvis systemet hadde klart å utnytte eksisterende lokasjonsdata.

Brukertest 2: Personen i brukertest 2 var den personen med en bakgrunn innenfor informatikk. Dette gjorde seg utslag i at brukergrensesnittet ble kritisert, og hvordan systemet fungerte ble mindre vektlagt. Likevel kom personen med noen interessante spørsmål.

Forslaget med å presentere informasjonen er allerede gjennomgått i Avsnitt 4.7. Å kunne se hvor det er ledige leseplasser er mulig, men å reservere plasser kan bli vanskelig da systemet ikke nødvendigvis er nøyaktig nok til å skille to leseplasser fra hverandre. Det som kan være en mulig løsning er å telle hvor mange studenter som befinner seg i en lesesal, sammenligne dette med hvor mange plasser det er tilgjengelig og ut i fra dette presentere hvor mange plasser som er ledig. Dette forutsetter at alle studenter bruker systemet, og det kan gjøre at en slik funksjonalitet er vanskelig å oppnå.

Brukergrensesnittet ble kommentert som "elendig". Siden dette er en prototype som både har administrative og brukeroppgaver i samme skjermbilde, og i tillegg har valgt å vise hvordan sammenhengen mellom informasjon og lokasjon kan modelleres, kan brukergrensesnittet oppfattes som både rotete, ulogisk og uoversiktlig. Dette er noe som må vurderes nøye i et ferdig system, slik at grensesnittet blir intuitivt og enkelt å bruke.

Brukertest 3: I den siste testen var ikke det trådløse kortet i brukerens pc støttet. Pc-en er av eldre versjon og har et integrert trådløst nettverkskort. Det virker som om dette ikke hadde en negativ innvirkning på hvordan brukeren oppfattet systemet.

Testen gikk som forventet, da småfeilene fra de første testene var rettet opp. Igjen kom brukeren med nyttig tilbakemelding. Spesielt det at brukeren likte tanken på et system som kunne reagere ut i fra en timeplan og hvor brukeren befant seg. At systemet henter fram oppgaver, forelesningsnotater osv. i en forelesning var noe brukeren ønsket seg.

Det med avstand er diskutert tidligere i oppgaven, men det ble nevnt for brukeren at det ikke er mulig å bedømme avstand ut i fra WiFi signaler, men at det kanskje kan løses ved at avstandene mellom alle punkt til alle andre punkt legges inn i databasen. Dette er en oppgave som er svært ressurskrevende og systemet vil bli unøyaktig. Noe spesielt nøyaktig vil det heller ikke bli.

Innsida blir også av bruker 3 kun brukt for å få tilgang til It's:Learning. Ergo blir meldinger som er lagt ut på Innsida ikke lest. Brukeren så for seg at et system som kunne varsle om nye (og relevante basert på en profil) meldinger ville vært noe han benyttet seg av.

5.4 Oppsummering og evaluering av resultat

Resultatene fra de ulike testene beviser at Fumble oppfyller kravene slik de er spesifisert i kravspesifikasjonen. Lokasjonsdelen kan vise til gode resultater selv med en enkel lineær algoritme. Likevel vil ikke en lineær algoritme være god nok for et ferdig system, da det vil ta for lang tid å dedusere en posisjon. I tillegg vil det legge beslag på all tilgjengelig cpu-kraft mens algoritmen kjører (i gjeldene implementasjon er det hvert sjette sekund).Lokaliseringsmodulen kan vise til en nøyaktighet på ca. 3 meter, som er tilsvarende andre lokaliseringssystemer basert på WiFi.

Informasjonsmodulen er sterkt knyttet til lokaliseringsmodulen, men presenterer korrekt informasjon basert på oppgitt lokasjon. Dette beviser at det er mulig å utvikle en felles struktur for kontekstavhengige systemer. Det gjenstår likevel mye arbeid med å implementere feks. en gazetteer og integrere eksterne systemer.

Samtlige testpersoner ga uttrykk for at dette er et system de kunne tenke seg å benytte hvis det hadde eksistert. For å få et fullstendig svar på spørsmålet om

dette er et system som ville ha blitt benyttet av studenter, må prototypen først implementeres med funksjonaliteten nevnt i Kapittel 3, og så testes ut på flere studenter enn hva denne oppgaven hadde tid til.

Resultatene viser også at studentene bruker kun et fåtall av eksisterende informasjonssystemer.

6 Konklusjon og videre arbeid

Målet med oppgaven var å utvikle et kontekstavhengig system som benyttet seg av eksisterende teknologi og modeller. Dette innebar å designe og implementere en prototype som kunne teste om 1) det er mulig å benytte eksisterende trådløse signaler for posisjonering 2) det kan utvikles en informasjonsmodell som knytter informasjon til lokasjon og utnytte hvordan lokasjonsdata ble lagret, og 3) vil et slikt system være til hjelp for studenter i studiehverdagen?

For å kunne implementere en prototype var det viktig å identifisere hva slags krav som gjelder i et kontekstavhengig system. Et kontekstavhengig system er et komplekst system, der det stilles mange krav innenfor flere fagfelt. I oppstartsfasen ble det identifisert flere krav, noen ble implementert, andre ble bare beskrevet. Prototypen må derfor sees på som et verktøy som ble brukt til å evaluere kravene som er definert i oppgaveformuleringen.

Prototypen er implementert i to faser. Den første fasen gikk ut på å utvikle en lokaliseringsdel, mens den andre gikk ut på å bygge en informasjonsdel som bruker lokasjonen foreslått av lokasjonsdelen. Bak begge delene ligger det et omfattende litteraturstudium, og implementeringen resulterte i prototypen Fumble.

Fumble gjør en mobil enhet i stand til å lokalisere seg selv, og benytter denne informasjonen til å presentere kontekstavhengig informasjon. Dette kan bli et kraftig verktøy hvis tatt i bruk, for kontekstavhengig systemer åpner mange nye dører i et samfunn der informasjon spiller en viktig rolle. Fumble fungerer altså som en personlig agent, som henter og presenterer informasjon den tror kan være nyttig med tanke på hva slags kontekst brukeren befinner seg i.

Fumble ble testet kontinuerlig underveis, men gjennomgikk en grundig test etter at utviklingen var ferdig. Både resultatene fra egne tester og tilbakemeldingen fra medstudenter vitner om at Fumble oppnår gode resultater. Også sammenlignet med lignende systemer gir Fumble gode resultater. (Merk at her er lokasjonsdelen og informasjonsdelen testet hver for seg mot tilsvarende systemer, da det ikke finnes kontekstavhengige system basert på WiFi tilgjengelighet som det er mulig å teste mot.)

Hvorvidt målene fra problemstillingen (avsnitt 1.1) ble oppfylt, er oppsummert under:

- Mål 1 var å undersøke om det er mulig å utnytte eksisterende trådløse signaler i en lokaliseringsmodul. Fumble beviser at det er mulig å bruke eksisterende trådløs teknologi (WiFi) ved NTNU Gløshaugen til å lokalisere mobile enheter. Valget om å lage et eget system for lokalisering (i stedet for å bruke Nibble), viste seg også å være et riktig valg. Ved å ha full tilgang til koden, var det mulig å ta egne valg basert på egne beslutninger. Dette i motsetning til å måtte tilpasse prototypen til et eksisterende system. Valget om å lage et eget system gjorde det mulig å utvikle et system som skal fungere med de fleste nettverkskort. Lokaliseringsmodulen oppnår, selv med en enkel

inference algoritme, en nøyaktighet på ca. 3 meter. Dette er tilsvarende mer avanserte systemer som RADAR, Nibble og Wireless Campus LBS. Siden Fumble er implementert med en enkel inference algoritme i en egen modul, kan denne forbedres betraktelig hvis dette er ønskelig.

- Mål 2 var å utvikle en informasjonsmodell som knytter informasjon til lokasjon og utnytter hvordan lokasjonsdata er lagret. Informasjonsdelen bygger på AROUND-arkitekturen, men er tilpasset slik at den kan utnytte et trådløst lokaliseringssystem. Oppgaven beviser at det er mulig å skape en relasjon mellom lokasjon og informasjon, som ikke bare virker, men som også er konseptuelt enkelt å forstå for studenter. Modellen utnytter altså sammenhengen mellom de fysiske rommene og informasjonsrommene.

- Det siste målet om hvorvidt et slikt system vil bli benyttet av studenter hvis implementert er ikke fullstendig besvart. Dette er ikke fordi Fumble er en dårlig løsning, men fordi Fumble har begrenset funksjonalitet i forhold til et ferdig system. Spesielt integreringen med andre systemer, og en fungerende kartmodul, vil gjøre Fumble mer attraktiv for studenter, da det med slik funksjonalitet er lettere å se fordelene ved et kontekstavhengig system. Likevel ga de få studentene som ble spurt uttrykk for at et slikt system kunne ha hjulpet de i studiehverdagen.

Det er bevist at det er mulig å utvikle et innendørs kontekstavhengig system som benytter seg av eksisterende teknologi og modeller, og som kan være til gjelp for studenter ved NTNU Gløshaugen i studiehverdagen.

6.1 Avgrensning

Oppgaven spenner over flere fagfelt. For å komme i mål har det derfor vært nødvendig å avgrense oppgaven. Under følger endel punkter som denne oppgaven ikke prøver å løse:

- Denne oppgaven avgrenser seg til hvordan et kontekstavhengig system kan fungere i et innendørs miljø, da det allerede eksisterer kommersielle løsninger som fungerer utendørs. Systemet vil kunne fungere i et utendørs miljø, men gjeldene implementasjon av informasjonsmodellen støtter kun lokasjoner som befinner seg innendørs.

- Oppgaven prøver å utnytte allerede eksisterende teknologier for å utvikle et innendørs kontekstavhengig system. Denne begrensningen gjør at det ved NTNU Gløshaugen bare finnes en teknologi som kan utnyttes, det er WiFi. RFID, Bluetooth og RadioEye[32] krever alle at det installeres utstyr både der det skal brukes og hos klienten. Med eksisterende teknologi menes derfor

[32]RadioEye er installert i ITV, men krever som nevnt ekstra utstyr på klientsiden.

teknologi som allerede er installert ved NTNU Gløshaugen, og som kan brukes uten ekstra utstyr.

- Trådløs datakommunikasjon gjennom WiFi har blitt mer og mer vanlig, og de fleste bærbare pc'er selges idag med integrerte trådløse nettverkskort. Det finnes også løsninger for pc-er som ikke selges med integrerte nettverkskort, ved å kjøpe eksterne tilleggskort. Oppgaven valgte NetStumbler for i teorien å kunne støtte flere trådløse nettverkskort, men utover dette er det ikke gjort noen forsøk på å støtte flere trådløse nettverkskort. Prototypen er dermed begrenset til trådløse kort støttet av NetStumbler.

- Prototypen er begrenset til funksjoner som er nødvendige for å vise at et kontekstavhengig system er gjennomførbart ved NTNU Gløshaugen, og området som dekkes av systemet er begrenset. En integrering med Innsida og It's:Learning er ikke implementert. Videokonferanse, lynmeldinger og taleoverføring er heller ikke implementert. Prototypen dekker bare IT-Vest, IT-Øst og IT-Sør. Det forutsettes altså at IT-byggene er "verden"[33], altså at det største området som kan dekkes i prototypen er disse tre bygningene. Det skal likevel nevnes at utviklingen av prototypen er gjort på en slik måte at det skal være enkelt for andre å bygge videre på denne. Prototypen har heller ikke begrep om tid, da informasjon eksisterer til noen sletter denne.

- Selv om det finnes digitale kart som kan utnyttes i et GIS[34], og posisjon er kjent, støtter ikke denne oppgaven interaktive kart, da dette er et problemområde i seg selv. Å støtte interaktive kart er noe som egner seg for videre arbeid (se avsnitt 6.3).

6.2 Evaluering av prototypen

Prototypen ble utviklet for å svare på spørsmålet i oppgaveformuleringen (avsnitt 1.1.2). Det viste seg at det var mulig å utvikle et innendørs kontekstavhengig system som baserte seg på trådløse signaler, og at et slik system, hadde det eksistert, ville blitt godt mottatt av studenter. Dette selv om prototypen ble implementert med begrenset funksjonalitet med tanke på hva som var ønskelig. I tillegg er det oppdaget noen feil (avsnitt 6.4) etter at implementeringen ble avsluttet, men ingen så kritisk at det gikk utover evalueringen og testingen i nevneverdig grad.

Teknologien som ble valgt viste seg å fungere bra. NetStumbler gjorde det mulig å hente ut signaler fra flere typer trådløse nettverkskort. VB.NET, sammen med tre-lags modellen, gjorde at selve utviklingen tok relativt kort tid. .NET applikasjoner er godt egnet i prosjekter der det er liten tid tilgjengelig, og det var mulig i denne oppgaven å gjenbruke databaselaget fra et tidligere prosjekt gjorde at prototypen raskt kunne testes.

[33] Omtalt som "Campus" i kildekoden.
[34] Geografic Information System.

Måten prototypen ble implementert på egner seg godt i utviklingssammenheng, ved at all kode kjøres på klienten. Et ferdig system vil bestå av en klient-server løsning, der lokaliseringsmodulen kjører på klienten, og informasjonsmodulen kjører på serveren.

6.3 Videre arbeid

Denne oppgaven har prøvd å kombinere to komplekse tema og innenfor to ulike fagfelt. For å kunne komme i mål, er oppgaven begrenset til funksjonalitet som viser hvordan et kontekstavhengig system virker.

Selv om oppgaven beskriver et ideelt system, finnes det sider ved Fumble, som ikke er diskutert i denne oppgaven, som gir grunnlag for videre arbeid:

Personvern

Det ble tidlig bestemt at denne oppgaven ikke skulle se nærmere på personvern, fordi dette er et eget fagfelt. Likevel, et program som Fumble berører mange aspekter ved personvernlovgivningen som det evt. må tas hensyn til. Hvis dette ikke blir gjort vil Fumble aldri bli godkjent for kommersielt bruk.

Interaktive kart

Ved å implementere en kartmodul i Fumble, er det mulig for systemet å presentere et kart med feks. veibeskrivelse. Dette var en problemstilling som var av interesse i begynnelsen av oppgaven, men som viste seg å, foruten å ta lang tid å implementere, være et eget fagfelt. Interaktive kart og kartsystemer krever kunnskap innen GIS-systemer, noe denne oppgaven ikke går inn på.

Integrering

Dette har vært diskutert tidligere i oppgaven, men er et problemområde som er svært viktig for et slikt system. For å få integrert Fumble med eksisterende systemer kreves et samarbeid med de som er ansvarlige for systemene som skal integreres, og kjennskap til systemene. Igjen er det tidsfaktoren som avgjorde at dette ikke ble implementert i Fumble, men oppgaven beskriver hvordan en slik implementasjon er mulig.

Inference algoritmen

Inference algoritmen er et område ved prototypen som har et stort potensiale for forbedringer. Ved å implementere profiler og Kalman filter kan nøyaktigheten forbedres betraktelig. Gjeldende implementering med en lineær algoritme bør byttes ut med feks. en vektor basert algoritme slik at systemet slipper å søke igjennom alle tilgjengelige lokasjoner.

Gazetteer

Systemet er implementert med en generell Gazetteer, men det er ønskelig å implementere en standard Gazetteer, eller en egendefinert Gazetteer feks. den foreslått i [25]. Dette for å bruke en Gazetteer til å knytte informasjon til lokasjon, feks. slik det er beskrevet i [25].

6.4 Kjente problemer

Fumble er kun en prototype, og utviklingen hadde en kort tidsfrist for å sette av nok tid til testing og evaluering. Derfor inneholder prototypen feil, som er oppdaget i ettertid, men ikke korrigert. Til nå er det oppdaget følgende problemer:

- **Fjern lokasjon:** Denne funksjonen skaper problemer for Fumble hvis du fjerner en lokasjon som ikke er sist i lista. En mulig løsning er å fjerne lokasjonen og alle lokasjonene etter, for så å legge inn lokasjonene som måtte slettes på nytt.
- **Lokalisering:** Fingeravtrykk fra andre bygninger får en score som er bedre enn hva en skulle forvente. Dette har sammenheng med at Fumble "straffer" aksesspunkt som ikke finnes i gjeldene signal for lite. Dette ble oppdaget etter at utviklingen av Fumble ble avsluttet.
- **Ukjent lokasjon:** Selv om systemet ikke er i stand til å foreslå en lokasjon, skal global informasjon være tilgjengelig. Problemet er at kontekstavhengig informasjon "henger igjen" når systemet oppgir en ukjent lokasjon. Det kan diskuteres om dette er en feil i systemet, eller om dette er et bevisst valg. Oppgaven har ikke tatt stilling til dette.
- **Datagrid lokasjonsdel:** Denne komponenten ble tatt med for enkelt å kunne sjekke om prototypen mottar trådløse signaler. Men det har vist seg at denne kan få prototypen til å krasje hvis det blir brukt samtidig som skanneren kjører. Grunnen til dette er at skanneren kjører i en egen tråd, og kan derfor prøve å oppdatere gridet samtidig som hovedtråden, noe som fører til at systemer krasjer. Løsningen er å ikke bruke datagridet når skanneren kjører.
- **Relasjonen mellom Scope og Lokasjon:** Det ble oppdaget da informasjonsmodellen var implementert og testet, at det ikke er mulig å relatere et scope til flere lokasjoner slik som det i utgangspunktet var planlagt. Dette skyldes at det ikke ble implementert funksjonalitet for dette i brukergrensesnittet og dermed ble feilen oppdaget for seint (slik det er nå er det heller ikke mulig via grensesnittet å knytte et scope til flere lokasjoner) til at dette kunne rettes opp. Løsningen er å legge til en attributt ID som hovednøkkel i entiteten L_S og gjøre om Scope_ID til fremmednøkkel.

Referanser

[1] A9. A9, 2006. URL `http://www.a9.com`. Sist lest: 20. Mai. 2006.

[2] Robbie Allen and Richard Puckett. Managing Enterprise Active Directory Services. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002. ISBN 0672321254.

[3] Arne Asphjell. Startskudd for trådløse trondheim, 2005. URL `http://www.universitetsavisa.no/ua_lesmer.php?kategori=nyheter&dokid=43e061b77ce664.57772625`. Sist lest: 05. Okt. 2005.

[4] Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. Modern Information Retrieval. ACM Press / Addison-Wesley, 1999. ISBN 0-201-39829-X. URL `citeseer.ist.psu.edu/baeza-yates99modern.html`.

[5] Anne Marte Blindheim. Ntnu-professoren svarer deg, 2005. URL `http://www.dagbladet.no/dinside/2005/04/26/429867.html`. Sist lest: 22. Nov. 2005.

[6] Paul Castro, Patrick Chiu, Ted Kremenek, and Richard Munz. A probabilistic room location service for wireless network environments, 2001. URL `http://mmsl.cs.ucla.edu/publications.shtml#muse`.

[7] Yi-Chao Chen, Ji-Rung Chiang, Hao hua Chu, Polly Huang, and Arvin Wen Tsui. Sensor-assisted wi-fi indoor location system for adapting to environmental dynamics. In MSWiM '05: Proceedings of the 8th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems, pages 118–125, New York, NY, USA, 2005. ACM Press. ISBN 1-59593-188-0. doi: http://doi.acm.org/10.1145/1089444.1089466.

[8] Keith Cheverst, Nigel Davies, Keith Mitchell, and Adrian Friday. Experiences of developing and deploying a context-aware tourist guide: the guide project. In MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking, pages 20–31, New York, NY, USA, 2000. ACM Press. ISBN 1-58113-197-6. doi: http://doi.acm.org/10.1145/345910.345916.

[9] Wikipedia den frie encyklopedi. Lokasjonsbaserte tjenester, 2005. URL `http://en.wikipedia.org/wiki/Location_Based_Services`. Sist lest: 10. Okt. 2005.

[10] Radio Eye. Cordis radioeye, 2006. URL `http://www.radionor.com/Innhold/Cordis.htm`. Sist lest: 04. Apr. 2006.

[11] Free Software Foundation. Gnu general public license, 1991. URL `http://www.gnu.org/copyleft/gpl.html`. Sist lest: 22. Nov. 2005.

[12] Anne J. Gilliland-Swetland. Introduction to metadata: Setting the stage, 2000. URL `http://www.getty.edu/research/conducting_research/standards/intrometadata/pdf/swetland.pdf`. Sist lest: 22. Apr. 2006.

[13] gis.com. Gis, 2005. URL `http://www.gis.com/whatisgis/index.html`. Sist lest: 09. Nov. 2005.

[14] Filip Greve, Bart Lannoo, Liesbeth Peters, Tom Leeuwen, Frederic Quickenborne, Didier Colle, Filip Turck, Ingrid Moerman, Mario Pickavet, Bart Dhoedt, and Piet Demeester. Famous: A network architecture for delivering multimedia services to fast moving users. Wirel. Pers. Commun., 33(3-4):281–304, 2005. ISSN 0929-6212. doi: http://dx.doi.org/10.1007/s11277-005-0573-2.

[15] I. Guvenc, C.T. Abdallah, R. Jordan, and O. Dedeoglu. Enhancements to rss based indoor tracking systems using kalman filters, 2003. URL `http://www.eece.unm.edu/controls/papers/Guv_CTA_Jor_Ded.pdf`.

[16] Rui José, Adriano Moreira, Helena Rodrigues, and Nigel Davies. The around architecture for dynamic location-based services. Mob. Netw. Appl., 8(4):377–387, 2003. ISSN 1383-469X. doi: http://dx.doi.org/10.1023/A:1024531629631.

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

[18] Barend Köbben, Arthur van Bunningen, and Kavitha Muthukrishnan. Wireless campus lbs: Building campus-wide location based services based on wifi technology, 2004. URL `http://www.svgopen.org/2005/images/WirelessCampusLBS_KobbenV2.pdf`.

[19] Sue Long, Rob Kooper, Gregory D. Abowd, and Christopher G. Atkeson. Rapid prototyping of mobile context-aware applications: the cyberguide case study. In MobiCom '96: Proceedings of the 2nd annual international conference on Mobile computing and networking, pages 97–107, New York, NY, USA, 1996. ACM Press. ISBN 0-89791-872-X. doi: http://doi.acm.org/10.1145/236387.236412.

[20] Dragoş Niculescu and Badri Nath. Vor base stations for indoor 802.11 positioning. In MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking, pages 58–69, New York, NY, USA, 2004. ACM Press. ISBN 1-58113-868-7. doi: http://doi.acm.org/10.1145/1023720.1023727.

[21] Paul Castro og Patrick Chiu og Ted Kremenek og Richard Muntz. The nibble location system, 2001. URL `http://citeseer.ist.psu.edu/cache/papers/cs/30525/http:zSzzSzwww.fxpal.comzSzpeoplezSzchiuzSzcckm-UBICOMP01.pdf/castro01probabilistic.pdf`. Siste lest: 2 Nov. 2005.

[22] Ole Hestnes og Runar Mehl Teigen. Anvendelse av metodikken "contextual design" for utvikling av lokasjonsbaserte tjenester ved grimstad museum, 2003. URL `http://student.grm.hia.no/master/ikt03/ikt6400/g23/`.

[23] Marshall Brain og Tom Harris. How gps receivers work, 2006. URL `http://electronics.howstuffworks.com/gps.htm`. Sist lest: 02. Mai. 2006.

[24] Paramvir Bahl og Venkata N. Padmanabhan. Radar: An in-building rf-based user location and tracking system. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies, 2:775–84, 2000. URL `http://research.microsoft.com/~padmanab/papers/infocom2000.pdf`. Sist lest: 08. Nov. 2005.

[25] Marit Olsen. Integrasjon og bruk av gazetteers og tesauri i digitale bibliotek. Master's thesis, NTNU Norges Teknisk-Naturvitenskapelige Universitet, 2004.

[26] Rolf Olstad and Atle Sægrov. Cordis radioeye localisation technology, 2002. URL `http://www.radionor.com/Downloads/radionor_white_paper.pdf`.

[27] Einar Snekkenes. Concepts for personal location privacy policies. In EC '01: Proceedings of the 3rd ACM conference on Electronic Commerce, pages 48–57, New York, NY, USA, 2001. ACM Press. ISBN 1-58113-387-1. doi: http://doi.acm.org/10.1145/501158.501164.

[28] ADL Gazetteer Content Standard. Adl gazetteer content standard, 2000. URL `http://www.alexandria.ucsb.edu/gazetteer/gaz_content_standard.html`. Sist lest: 29. Apr. 2006.

[29] Thuan Thai and Hoang Q. Lam. .Net Framework Essentials. O'Reilly & Associates, Inc., Sebastopol, CA, USA, 2003. ISBN 0596005059.

[30] Wikipedia. Bluetooth, 2006. URL `http://en.wikipedia.org/wiki/Bluetooth`. Sist lest: 04. Apr. 2006.

[31] Wikipedia. Empirical, 2006. URL `http://en.wikipedia.org/wiki/Empirical`. Sist lest: 17. Apr. 2006.

[32] Wikipedia. Galileo positioning system, 2006. URL `http://en.wikipedia.org/wiki/Galileo_positioning_system`. Sist lest: 03. Apr. 2006.

[33] Wikipedia. Global positioning system, 2006. URL `http://en.wikipedia.org/wiki/Gps`. Sist lest: 03. Apr. 2006.

[34] Wikipedia. Kalman filter, 2006. URL `http://en.wikipedia.org/wiki/Kalman_filter`. Sist lest: 13. Apr. 2006.

[35] Wikipedia. Object oriented, 2006. URL `http://en.wikipedia.org/wiki/Object_oriented`. Sist lest: 03. Mai. 2006.

[36] Wikipedia. Rfid, 2006. URL `http://en.wikipedia.org/wiki/Rfid`. Sist lest: 04. Apr. 2006.

[37] Wikipedia. Screen scraping, 2006. URL `http://en.wikipedia.org/wiki/Screen_scraping`. Sist lest: 23. Apr. 2006.

[38] Wikipedia. Wi-fi, 2006. URL `http://en.wikipedia.org/wiki/WiFi`. Sist lest: 04. Apr. 2006.

Appendix

A Terminologi

- Aksesspunkt
  Aksesspunkt (Access point) eller AP er en node som tilbyr trådløs tilkobling.

- Fingeravtrykk
  Et fingeravtrykk slik det er definert i denne oppgaven inneholder flere sampler. Ved å legge inn en ny lokasjon lages en fingeravtrykk id og et navn i tillegg til en sample (se over). Senere kan man velge å forsterke (reinforce) denne lokasjonen, og da legges en ny sample til fingeravtrykket. Dette gjør programmet i stand til å gjenkjenne et sted under vekslende forhold.

- Geographic Information Systems
  Geographic Information Systems (GIS) er definert som:

  > GIS is a technology that manages, analyzes, and disseminates geographic knowledge.[13]

- Kontekstavhengig informasjon
  Kontekstavhengig informasjon er informasjon som kan relateres til områder eller tilstand.

- Lokasjonsbasert tjeneste
  Wikipedia[9] definerer en lokasjonsbasert tjeneste som

  > "... is a service provided to the subscriber based on their current geographical location."

- Prototype
  En prototype er et system som er under utvikling, altså før det kan sies å være sluttført.

- RSS
  Recieved Signal strength.

- Sample
  En sample er den beste samlingen av aksesspunkt (access points) tatt over en tidsperiode. Gjeldende implementering tar tre sampler over seks sekunder og lagrer den beste.

- SNR
  Signal-to-noise.

B Kravspesifikasjon

B.1 Bruker krav 1 - Hvor er

1. **Case id, name**
   UC1, Use Case 1 - "Hvor er"-spørsmål

2. **Case id, name**
   UC1, Use Case 1 - "Hvor er"-spørsmål

3. **Risk Factors**
   Frequency of occurrence:
   Flere ganger i timen (>10)
   Impact of failure:
   Likely: Spørsmål skrevet feil, lav risiko, bruker må stille spørsmålet på nytt
   * Skrivefeil, ikke et "sted" som systemet ikke kjenner til
   Rare: Systemet klarer ikke å finne noe svar, medium risiko, kan miste brukere, logg
   * kan minimeres ved å få med seg hele NTNU
   Worst: Systemet foreslår feil "sted", høy risiko, bruker må kunne gi tilbakemelding

4. **Case conditions**
   Invariants:
   Forskjellige spørsmål, samme svar. Eks: "stripa\Stripa\Stripa\stripa?", "Kor e Stripa?", "I hvilken bygning er Stripa?", "Hvor finner jeg Stripa?", "Where is Stripa?"
   "Hvor er forelesningene i IT3807?", "Hvor er dagens forelesning i IT3807?", "Hvor er forelesningen i IT3807 på mandag?".
   Pre-conditions:
   - Bruker trenger ikke å være logget inn.
   - Bruker trenger ikke å posisjoneres.
   - Bruker må vite at det er en plass som kalles Stripa, og ikke\eller S100\den lange gangen.
   - Stripa må være et oppslag i systemets database.

5. **Course of action**

   Student Lokasjonsbasert infosystem
   1.1. Spør etter "sted".
   1.2. Sjekker om "sted" finnes i systemet.

   Presenterer kart over Gløshaugen, med et merke over "sted".
   Steg 1.1 og 1.2 kan gjentas til bruker er fornøyd.

Alternative course
2.1. Får ikke treff og ber brukeren om å sjekke\reformulere spørsmålet.
2.2. Sjekker og evt. reformulerer spørsmålet.
2.2.1. Velger å loggføre spørsmålet. Se 1.2.
3.1. Systemet finner flere alternativ.

Presenterer en liste m\forklaring til bruker.
3.2. Bruker velger fra listen, eller søker på nytt. Se 1.2.
4.1. Sjekker om "sted" finnes i systemet og presenterer kart over Gløshaugen, med et merke over "sted", men er ikke korrekt.
4.2. Bruker finner ut at dette ikke er korrekt, og skal da kunne gi tilbakemelding til systemet.
4.3. Systemet loggfører episoden og sender varsel til systemansvarlig.

5.1. Hvor er "forelesning"?
5.3. Hvor er dagens "forelesning"?
5.2. Sjekker om "forelesning" finnes. Er ikke annet spesifisert, vis kart over Gløshaugen. Har kurset flere forelesningssaler, vis alle.
5.4. Skal kun gi et treff, evt. ingen hvis forelesning ikke finnes\avlyst.

6. **Comments**
Bruker er kjent ved Gløshaugen, dvs. klarer å orientere seg vha. kart, men ikke navn på bygninger eller romnummerering.

Kunne valgt å skille ut pkt 5 som egen UC, men kom til den konklusjonen at dette er en variasjon av spørsmålet.

B.2 Bruker krav 2 - Hva inneholder

1. **Case id, name**
UC2, Use Case 2 - "Hva inneholder'-spørsmål?"

2. **Risk Factors**
Frequency of occurrence:
Daglig (>5)
Impact of failure:
Likely: Spørsmål skrevet feil, lav risiko, bruker må stille spørsmålet på nytt
Rare: Systemet klarer ikke å finne noe svar, medium risiko, kan miste brukere
Worst: Systemet gir feilaktige opplysninger, høy risiko, bruker må kunne gi tilbakemelding

3. **Case conditions**
Invariants:
Forskjellige spørsmål, samme svar. Eks: antall seter\hvor mange seter
Pre-conditions:

- Bruker trenger ikke å være logget inn.
- Bruker trenger ikke å posisjoneres.
- Bruker må (muligens) kjenne til\presenteres med mulige attributter.
- Attributtene må være et oppslag i systemets database.

4. **Course of action**

   Student Lokasjonsbasert infosystem
   1.1. Spør etter "innhold".
   1.2.1. Sjekker om "innhold" finnes i systemet.
   presenterer kart over Gløshaugen, men et merke over "sted" og presenterer "innhold".

   eller

   1.2.2. Gir kun informasjon om "innhold".
   Steg 1.1, 1.2.1. og 1.2.2. gjentas til bruker er fornøyd

   Alternative course 2.1. Får ikke treff og ber brukeren om å sjekke\reformulere spørsmålet
   2.2.. Sjekker og evt. reformulerer spørsmålet.
   2.2.1. Velger å loggføre spørsmålet. Se 1.2.1 og 1.2.2.
   3.1. Systemet gir feilaktige opplysninger:
   - Feil i data
   - Kommer frem til feil svar
   3.2. Bruker finner ut at dette ikke er korrekt, og skal da kunne gi tilbakemelding til systemet.

5. **Comments** Ingen.

B.3 Bruker krav 3 - Hvor går jeg

1. **Case id, name (mulig avgrensning)** UC3, Use Case 3 - "Hvor går jeg"-spørsmål?

2. **Risk Factors** Frequency of occurrence:
   Flere ganger i timen (>5)
   Impact of failure:
   Likely: Spørsmål skrevet feil, lav risiko, bruker må stille spørsmålet på nytt
   Likely: Egne navn på steder, medium risiko, kan miste brukere, logg, nytt oppslag
   Rare: Systemet klarer ikke å finne noe svar, medium risiko, kan miste brukere
   Worst: Systemet foreslår feil veivalg, høy risiko, bruker må kunne gi tilbakemelding

3. **Case conditions** Invariants:
   Forskjellige spørsmål, samme svar. Eks: kantina\hangaren, og kantina for realfag er forskjellig fra kantina til elektro.
   Egne navn på steder. Eks: "lille Sito"
   Pre-conditions:
   - Bruker må være logget inn. (Lovlig posisjonering?)
   - Bruker må kunne posisjoneres.
   - Attributtene må være et oppslag i systemets database.

4. **Course of action** Student Lokasjonsbasert infosystem
   1.1. Spør etter "sted".
   1.2. Sjekker om det er mulig å posisjonere bruker. Sjekker om "sted" finnes i systemet.

   Presenterer kart over Gløshaugen (sentrert), med forklaring til "sted".
   Steg 1.1. og 1.2. gjentas til bruker er fornøyd.

   Alternative course
   2.1. Får ikke treff og ber brukeren om å sjekke\reformulere spørsmålet.
   Logger spørsmålet for vurdering.
   2.2. Sjekker og evt. reformulerer spørsmålet. Se 1.2.
   3.1. Bruker er ikke logget inn \kan ikke posisjoneres.
   3.2. Ber brukeren om å logge inn, eller opplyser om at posisjonering ikke er mulig.
   3.3. Bruker logger inn eller beveger seg til et sted som gjør posisjonering mulig. Se 1.2.
   4.1. Bruker får feil sted, gir tilbakemelding.
   4.2. Systemet sender en melding til systemansvarlig.

5. **Comments** Ved lille Sito. Merk at det her finnes to kafeer ved navn Sito.

B.4 Bruker krav 4 - Hvor befinner

1. **Case id, name (mulig avgrensning)** UC4, Use Case 4 - "Hvor befinner"-spørsmål?

2. **Risk Factors** Frequency of occurrence:
   Flere ganger i timen (>50)
   Impact of failure:
   Likely: Posisjonering ikke mulig, lav risiko, tilbakemelding
   Likely: Posisjonering upresis, lav risiko, foreslå område "person" kan befinne seg i
   Worst: Systemet klarer ikke å finne noe svar, høy risiko, kan miste brukere (her kan mye gå galt, vanskelig å gi presis tilbakemelding)

3. **Case conditions** Invariants:
   Ingen
   Pre-conditions:
   - Bruker må være logget inn.
   - "person" må ha gitt samtykke for å kunne posisjoneres
   - "person" må kunne posisjoneres
   - "person" må være logget inn

4. **Course of action**

   Student Lokasjonsbasert infosystem
   1.1. spør etter "person"
   1.2. sjekker om "person" kan posisjoneres og presenterer kart over Gløshaugen, men et merke over "person"
   Steg 1.1. og 1.2. gjentas til bruker er fornøyd

   Alternative course
   2.1. "person" kan ikke posisjoneres og systemet gir tilbakemelding
   3.1. "person" kan posisjoneres, men er "opptatt". Avhengig av status gir systemet tilbakemelding.
   - Opptatt: "person" er på "sted", men opptatt
   - Usynlig: "person" er ikke tilgjengelig
   - Ledig: "person" er på "sted"
   3.2. Avhengig av tilbakemelding tar bruker kontakt eller ikke.
   5. Bruker ikke pålogget\tilgjengelig. Ingen treff. (Søk ikke tilgjengelig når utlogget.)

5. **Comments** Ingen.

B.5 Bruker krav 5 - Hva er

1. **Case id, name** UC5, Use Case 5 - "Hva er"-spørsmål?

2. **Risk Factors** Frequency of occurrence:
   Ukentlig (>10)
   Impact of failure:
   Likely: Spørsmål skrevet feil, lav risiko, bruker må stille spørsmålet på nytt
   Rare: Systemet kommer med unøyaktig informasjon, medium risiko, tilbakemelding
   Worst: Ingen funnet enda.

3. **Case conditions** Invariants:
   Her kan spørsmålene variere stort.
   Pre-conditions:

- Attributtene må være et oppslag i systemets database.

4. **Course of action** Student Lokasjonsbasert infosystem
   1.1. spør etter "informasjon"
   1.2. sjekker om "informasjon" finnes i systemet. Presenterer først tekst, men mulighet til å velge å vise lokasjon (kart).

   Steg 1.1. og 1.2. gjentas til bruker er fornøyd

   Alternative course
   2.1. får ikke treff og ber brukeren om å sjekke\reformulere spørsmålet
   2.2. sjekker og evt. reformulerer spørsmålet. Se 1.2.
   3.1. får ikke treff og ber brukeren om å skrive inn aktuell informasjon.
   3.2. Bruker kan velge: skrive inn ny informasjon, prøve igjen eller gå tilbake.
   3.3. Skriver brukeren inn ny informasjon, skal denne godkjennes, evt. redigeres av en administrator.
   3.4. Bruker får tilbakemelding om at informasjonen er lagt til for godkjenning.

5. **Comments** Dette er et brukerkrav som kan inngå i brukerkrav 1 og 2.

B.6 Bruker krav 6 - Endring av kontekst

1. **Case id, name** UC5, Use Case 6 - Endring av kontekst

2. **Risk Factors** Frequency of occurrence:
   Flere ganger i timen (>5)
   Impact of failure:
   Likely: Lokasjonsdel deduserer feil posisjon, medium til høy risiko, tilbakemelding
   Rare: Lokasjonsdel klarer ikke å finne posisjon, lav risiko, ingen
   Rare: Informasjonsdel mangler informasjon, eller klarer ikke å finne riktig informasjon, medium risiko, kan miste brukere
   Worst: Kritiske meldinger når ikke fram, høy risiko, feilhåndtering.

3. **Case conditions** Invariants:
   Bruker vil ha informasjon som ikke er knyttet til gjeldende kontekst
   Pre-conditions:
   - Bruker må være logget inn
   - Posisjonering må være mulig

4. **Course of action** Student Lokasjonsbasert infosystem
   1.1. Klienten beveger seg fra et område til et annet
   1.2. Systemet oppdaterer kontekst for gjeldene område

   Steg 1.1. og 1.2. gjentas så lenge brukeren beveger seg

   Alternative course
   2.1. Bruker vil ha tilgang til informasjon som ikke er knyttet til gjeldene posisjon
   3.1. Systemet bør tilby mulighet for å søke etter all tilgjengelig informasjon

5. **Comments** Ingen.

C Installasjonsmanual Fumble

Fumble skal i teorien kjøre på de fleste Windows plattformer (98, ME, NT og XP)[35] og støtter de fleste trådløse nettverkskort[36]. I tillegg til dette må følgende kriterier være oppfylt:

- .NET Framework 1.1 må være installert på klienten (kan lastes gratis ned fra Windows Update[37]).
- NetStumbler versjon 0.4.0 må være installert på klienten (kan lastes ned gratis[38]).
- Følgende filer må legges i samme katalog: "C:\Fumble"[39]:
  - Fumble.exe (Selve prototypen).
  - Fumble.mdb (Databasen som inneholder lokasjons og informasjonsdata).
  - fumble_to_db.vbs (VB Script nødvendig for kommunikasjon mellom NetStumbler og Fumble).
  - AxInterop.SHDocVw.dll og Interop.SHDocVw.dll (Bibliotek som Fumble er avhengig av).

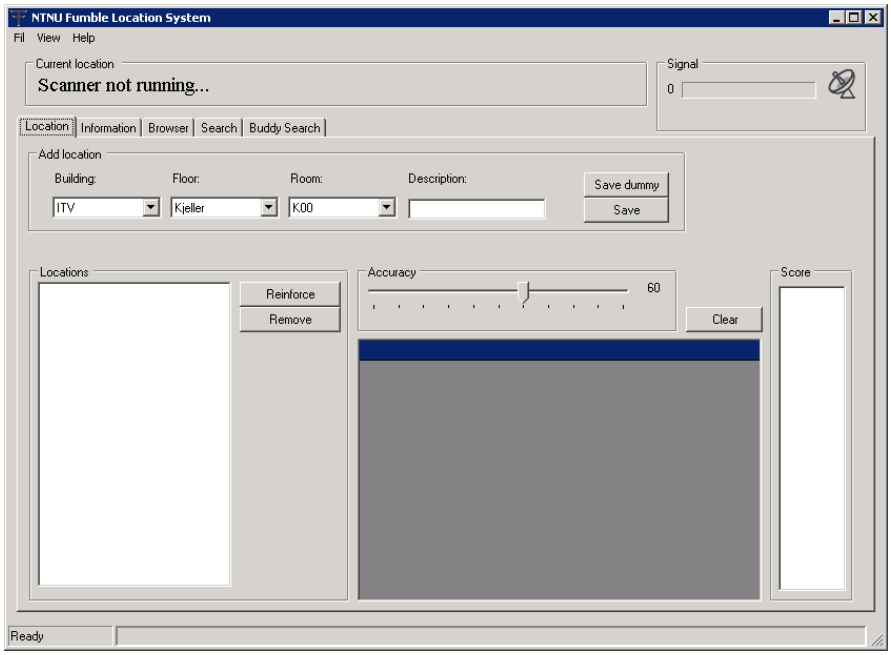
Filene som er nødvendig finnes på vedlagt CD (Vedlegg 1), bortsett fra .NET Framework og NetStumbler.

C.1 Installasjonen steg for steg

1. Opprett en katalog på C:, med navn Fumble ("C:\Fumble")
2. Legg Fumble.exe, Fumble.mdb, fumble_to_db.vbs, AxInterop.SHDocVw.dll og Interop.SHDocVw.dll i katalogen som ble opprettet i pkt. 1.
3. Installer NetStumbler, følg instruksjonene og start opp programmet.
4. Under View, Options på menyen, klikk på fanen Scripting og velg External Script som Type. Skriv inn C:\Fumble\fumble_to_db.vbs, eller klikk deg fra til fumble_to_db.vbs som nå skal ligge i "C:\Fumble" ved å klikke på Browse. Velg Ok og minimer NetStumbler.
5. I NetStumbler slå av "Auto reconfigure", er et icon med to tannhjul under menyen.

[35] Testet på ME og XP.
[36] For en liste over kort som er støttet se: `http://www.netstumbler.org/`
[37] `http://windowsupdate.microsoft.com`
[38] `http://www.netstumbler.com/downloads`
[39] Programmet vil ikke virke om filene legges i en annen katalog.

6. Start Fumble ved å dobbeltklikke på Fumble.exe i C:\Fumble.
7. Hvis alt har gått bra vil du se et skjermbilde som vist i Figur 25.



Figur 25: Fumble

Nå er alt klart for å ta i bruk Fumble.

D Brukermanual Fumble

Etter at Fumble er installert mangler systemet både lokasjons- og informasjonsdata. Dette er fordi Fumble distribueres med en tom database. Når Fumble startes for første gang, må minimum lokasjonsdata lagres. Dette gjør Fumble istand til å gjenkjenne lokasjoner. Hvordan dette gjøres er beskrevet i avsnitt D.1. For å fullt utnytte Fumble må det også lagres informasjonsdata. Det er verdt å merke seg at det spiller ingen rolle i hvilken rekkefølge dette utføres i. Om lokasjons- eller informasjonsdata legges inn først, har ingen betydning: Fumble tar seg av at relasjonen blir korrekt opprettet. Hvordan lagre informasjonsdata er beskrevet i avsnitt D.2.

Dette kapitlet forutsetter at Fumble er installert korrekt.

D.1 Lokasjonsdel

Når Fumble starter for første gang, må det legges inn lokasjoner. For å legge inn nye lokasjoner velg "Location"-fanen. Velg bygning, etasje og rom tilsvarende det stedet prototypen befinner seg, og velg en beskrivelse for plassen. Trykk deretter "Save". Det tar ca. seks sekunder for Fumble å lagre en ny lokasjon, og statusbar indikerer at Fumble jobber. Det er ikke mulig å bruke Fumble mens en ny lokasjon legges til, da de fleste knapper er deaktivert. Normal operasjon gjenopprettes når Fumble er ferdig. Gjenta disse stegene for andre lokasjoner.

Nå er Fumble i stand til å gjenkjenne lokasjonener som ble lagt inn. For å teste dette velg "Fil" og "Scan on". Det vil gå noen sekunder før Fumble kommer med et forslag, dette er helt normalt da det det tar tid å dedusere posisjonen.

Det viser seg at Fumble trenger flere målinger for å nøyaktig kunne dedusere posisjon. For å forsterke en lokasjon, velges denne "Locations" og trykk "Reinforce". Igjen vil det gå noen sekunder for Fumble gjenopptar normal operasjon.

Det som er verdt å merke seg når en legger inn eller forsterker lokasjoner er at orienteringen spiller en rolle. Derfor kan det være lurt å forsterke en lokasjon i alle himmelretninger for å gi Fumble best mulig grunnlag for å dedusere posisjon.

D.2 Informasjonsdel

For at Fumble skal tilby informasjon til brukere, må det defineres ved hvilke lokasjoner forskjellig informasjon skal tilbys. Dette gjøres ved å klikke på "Information"-fanen. Her er det mulig å knytte informasjon til en bestemt lokasjon. Framgangsmåten er ganske lik lokasjonsdelen, med unntak av at her har brukeren mulighet til å legge inn en alternativ beskrivelse og en url knyttet til informasjonen.

Oasen i 2.etasje i ITV-bygget kan legges inn med

Scope name	Alt.desc.	Building	Floor	Room	URL	Type
Oasen	Lunsjkrok for stud. og ansatte	ITV	2.etg.	200		Lok. info.

Tabell 22: Tabell Informasjon

D.3 FUMBLE er klar til bruk

Når både lokasjons og informasjonsdata er på plass, er FUMBLE klar til bruk. Slå på Fumble ved å trykke på "Scan on". FUMBLE vil da foreslå en lokasjon basert på trådløse signaler, og informasjon som er knyttet til gjeldene lokasjon.

D.4 Feilsøking

1. NetStumbler gir en feilmelding når jeg prøver å laste scriptet.

   - Sjekk at ditt trådløse nettverkskort er støttet av NetStumbler. Hvis det står helt nederst i NetStumbler: "No wireless adapter found" er ikke ditt kort støttet. Det er da ikke mulig å bruke Fumble med dette kortet.

2. Trådløskortet er støttet av NetStumbler, men Fumble rapporterer samme lokasjon uansett.

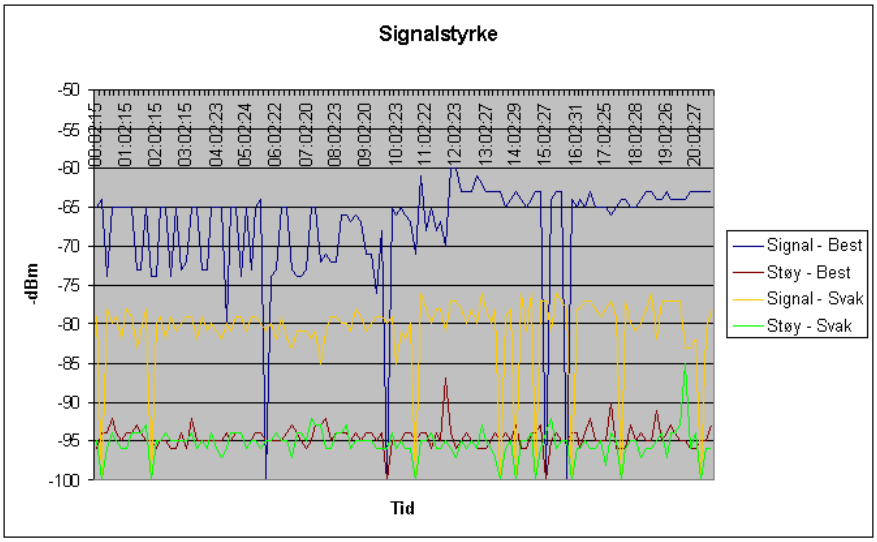
   - Det kan skje at NetStumbler av og til opplyste at scriptet kjører, selv om dette ikke var tilfellet. Løsningen på dette er i NetStumbler, under View, Options på menyen, klikk på fanen scripting. Trykk "Browse" og velg filen fumble_to_db.vbs. Trykk "Ok" to ganger.
   - Problemet oppstår også hvis filene ikke ligger i rett katalog, altså C:\Fumble.

3. Fumble rapporterer fortsatt samme lokasjon.

   - Selv om det finnes trådløs dekning i ditt område, krever Fumble minst tre unike aksesspunkt for å fungere. Dette er lett å sjekke ved å se i NetStumbler hvor mange aktive aksesspunkt som finnes.

E Trådløse signaler

For bedre å forstå hvordan trådløse signaler varierer over tid, ble det utviklet en liten applikasjon som lagret signalstyrke og støy med 10 minutters mellomrom. Det ble valgt ut to aksesspunkt som referanser, det ene var det med sterkest signal (Signal - Best), og det andre var et aksesspunkt med middels signal (Signal - Svak). Resultatene fra applikasjonen presenteres i Figur 26 under.



Figur 26: Signalstyrke

Resultatene viser at det beste signalet varierer mellom -60dBm og -80dBm i signalstyrke og -87dBm og -96dBm for støy, og det svake signalet varierer mellom -76dBm og -85dBm i signalstyrke og -85dBm og -98dBm for støy. Dette bekrefter at støy ikke er egnet for lokalisering, da støy er ganske konstant sett over tid, men har tilfeldige utslag som ikke kan forklares. Hadde støy økt betraktelig i arbeidstiden, ville dette kunne ha blitt brukt for å øke presisjonen. Derimot ser vi av signalstyrken at denne ikke er like konstant. Målingen er foretatt mellom 00:00 og 20:00, og vi kan se at mellom 11:00 og 12:00 skjer det en endring i både det sterke og det svake signalet. Det sterke signalet blir sterkere og mer stabilt, mens det svake blir også sterkere, forholder seg omtrent like stabilt, med unntak av at aksesspunktet ramler ut oftere. Dette trenger ikke å være pga. av dårlig signal, men at aksesspunktet ikke ble med i en skanning og derfor rapporteres styrke med -100dBm. Det er flere grunner til at en måling ikke får med seg et aksesspunkt og derfor er dette målinger man må se bort fra.

Vi kan konkludere med at det må eksistere et, eller helst flere, fingeravtrykk både før og etter klokka 11:00 - 12:00, da signalet varierer såpass mye, for å få høyest mulig nøyaktighet ved den lokasjonen der målingene ble utført.

F Funksjoner

Under følger en teknisk beskrivelse av de viktigste funksjonene i Fumble:

F.1 Viktige funksjoner i lokasjonsdel

Prototypen er utviklet etter den generelle trelags strukturen som består av lagene: Presentasjons-lag, logikk-lag og database-lag. Dette er gjort for enkelt å kunne bytte ut databasen med hvilken som helst JET kompatibel database hvis ønskelig og for å skille koden som oppdaterer brukergrensesnittet fra beregningene som blir gjort kontinuerlig. Logikklaget er det laget der alle beregninger utføres, og her er det tre funksjoner som er viktige: AddLocation, ReInforceLocation og Inference.

Et annet valg som ble gjort er å dele inn programmet i tråder (threads). Grunnen er at inference algoritmen er avhengig av å kjøre kontinuerlig. Uten tråder ville dette ha "låst" brukergrensesnittet da algoritmen ville lagt beslag på prosessoren. Det viste seg at bruken av tråder førte til et responsivt program, selv om det foregår til tider tunge beregninger i bakgrunnen.

Public Sub AddLocation
(ByVal Name As String, ByVal Room As String)
Funksjonen tar inn argumentene "Name" og "Room". Name er en beskrivende tekst for dette punktet, feks. "Oasen", og Room er hvilket romnummer klienten befinner seg i, feks. "200". For å kunne legge inn en ny lokasjon henter funksjonen inn høyeste Fingerprint_ID og legger til 1, og fyller inn fingerprint tabellen med Fingerprint_ID, Name, og Room. Siste steg er å fylle inn AP tabellen med den sample som på dette tidspunkt ligger klar, da StartSampling() har blitt kjørt som en egen tråd imellomtiden. Tabellen AP har kolonnene MAC, Signal, Noise, SNR, Fingerprint_ID og Sample_ID. MAC, Signal og Noise kommer fra StartSampling() som er en funksjon som leser signalet fra databasen som oppdateres kontinuerlig. SNR (Signal to Noise ratio) er beregnet fra Signal og Noise, og det er denne verdien som brukes i inference algoritmen.[40] Det brukes samme Fingerprint_ID som for Fingerprint tabellen og Sample_ID er alltid 1 da dette vil være den første sample'en for gjeldene lokasjon.

Public Sub ReInforceLocation
(ByVal ID As String)
Denne funksjonen er svært lik AddLocation, med unntak av to deler. Funksjonen oppdaterer kun AP tabellen med nye samples, og den finner høyeste Sample_ID for gjeldene Fingerprint og legger til 1. På denne måten sikrer programmet at det kan eksistere flere samples for et fingerprint.

Public Function Inference() As Integer() Dette er funksjonen som gjør prototypen i stand til å lokalisere seg selv. Den kjøres, på samme måte som StartSampling(), i en egen tråd og kan derfor slås av eller på hvis brukeren ønsker

[40] I gjeldene implementasjon er det RSS som benyttes for inference

dette. I tillegg er det to grunner til at vi valgte å kjøre denne funksjonen som en egen tråd, den første er at det lar oss endre prioritet på oppgaven. og den andre er at brukergrensesnittet er responsivt selv om programmet kontinuerlig sjekker hvor det befinner seg.

Funksjonen har to tilstander, den ene er "beregn hvor jeg er" og den andre er "finn beste signal", og den kan bare befinne seg i en tilstand av gangen. Dette for å utnytte ventetiden det tar å beregne lokasjonen bedre. For å ta den siste tilstanden først, leser den signalet (eg. en sample) hvert andre sekund og sammenligner dette med forrige avlesning. Er siste signal bedre, blir dette satt til gjeldene og det gamle fjernet, hvis ikke er det gamle gjeldende. Den første tilstanden tar det beste signalet og sammenligner dette med alle som er lagret i databasen og returnerer en tabell som inneholder alle aktive lokasjoner (lokasjoner kan slettes og er dermed ikke aktive) og en tilhørende poengsum. Den lokasjonen som har minst poengsum, 0 betyr et eksakt treff, er den lokasjonen programmet tror den befinner seg på.

F.2 Viktige funksjoner i informasjonsdel

I tillegg til funksjonene i lokasjonsdelen, er det (så langt) en viktig funksjon i informasjonsdelen. Den er avhengig av andre funksjoner som forklart i F.1 og noen mindre viktige funksjoner i informasjonsdelen. Disse er ikke tatt med for å forenkle forståelsen. Funksjonen er forklart i avsnittet under, sammen med noen administrative funksjoner som er nødvendig for å legge inn ny kontekstavhengig informasjon.

Public Sub FindContext(ByVal ID As String) Som beskrevet i F.1 er det opp til lokasjonsdelen å finne ut hvor klienten befinner seg. Når det skjer en endring i kontekst (dvs. at klienten har beveget seg inn i et nytt område), eller etter et gitt tidsintervall kaller programmet FindContext med identifikatoren til gjeldende fingeravtrykk. Dette er nok til at FindContext kan finne hva slags informasjon som er aktuell, og presentere denne til brukeren.

Public Sub addInfo(ByVal ScopeLevel As String, ByVal ScopeID As String, ByVal Name As String, ByVal Type As String, ByVal Info As String, ByVal URL As String)

Denne funksjonen lar brukeren lagre informasjon, og knytte denne til en, eller flere, lokasjoner. Prototypen lar brukeren legge inn hvilken lokasjon som skal relateres til informasjonen, samt informasjon som: Navn, alternativ beskrivelse og URL.

G Kildekode

Kildekoden er vedlagt som "Vedlegg 1". "Vedlegg 1" inneholder en CD og finnes bakerst i oppgaven.

CD-en inneholder følgende kataloger og filer:

- Katalogen "Database" inneholder databasefilen (fumble.mdb).
- Katalogen "DDLs" (dynamic link library) inneholder nettleseren (AxInterop.SHDocVw.dll og Interop.SHDocVw.dll).
- Katalogen "Fumble" inneholde både kildekode (fumble.sln), og en kjørbar fil i \bin (fumble.exe).
- Katalogen "VB Script" (visual basic) inneholder både skriv til fil (fumble_to_file.vbs) og skriv til database (fumble_to_db.vbs) scriptene.

Se Appendix C for hvordan Fumble skal installeres.