BACHELOROPPGAVE:

Hovdetoppen

Et pilotprosjekt for opptak av forelesninger

FORFATTERE:

Simen Rune Bragen

Marius Slåtsveen

Dato: 27.05.11

SAMMENDRAG

Tittel:	Hovdetoppen – Et pilotprosjekt for opptak av forelesninger	Nr. :
		Dato : 27.05.11
Deltaker(e):	Simen Rune Bragen	
	Marius Slåtsveen	
Veileder(e):	Kjell Are Refsvik	
Oppdragsgiver:	IT-avdelingen Høgskolen i Gjøvik	
Kontaktperson:	Stian Husemoen	
Stikkord (4 stk)	Matterhorn, forelesningsopptak, OpenSource, Linux	

Antall sider: 74	Antall bilag: 7	Tilgjengelighet (åpen/konfidensiell): Åpen

Kort beskrivelse av bacheloroppgaven:

Høgskolen i Gjøvik tilbyr blant annet fleksibel utdanning. Dette sammen med endrede studievaner generelt, setter nye krav til undervisning.

Mange høgskoler og universiteter har i de siste årene jobbet med separate prosjekter rundt opptak av forelesninger. Dette har Uninett nå tatt tak i, og har nå et samarbeidsprosjekt med høgskole- og universitetssektoren. Prosjektet tar blant annet sikte på å finne en felles løsning for opptak og distribusjon av forelesninger. De startet arbeidet i april 2010, men de har forstatt ingen løsning i sikte.

Matterhorn er et system for opptak av forelesninger som utvikles i samarbeid mellom universiteter og høgskoler i hele verden. Dette systemet gjør det mulig og ta opp og distribuere forelesninger. Men det krever at en administrator manuelt legger alle opptak som skal gjøres inn i systemet.

Vårt overordnede effektmål har vært å bane vei for automatiske opptak av forelesninger.
Vi har utviklet en prototype av et slikt automatisk system og en manuell løsning, som sammen med Matterhorn gjør det enklere å ta opptak av forelesninger. Det automatiske systemet tar opptak basert på data som finnes i skolens eksisterende timeplan – TimeEdit. Mens det manuelle systemet gjør det mulig å ta opptak når som helst i de rommene der det er montert opptaksutstyr.

Det automatiske systemet er utviklet i PHP og overskriver opptakstimeplanen til Matterhorn. Mens det manuelle systemet er utviklet i C, Bash, PHP og JavaScript og tar opptak som det senere overfører til Matterhorn Server.

SUMMARY

Title:	Hovdetoppen – A proof of concept for lecture recording	No. :
		Date : 27.05.11
Participants:	Simen Rune Bragen	
	Marius Slåtsveen	
Supervisor:	Kjell Are Refsvik	
Employer:	IT-department Høgskolen i Gjøvik	
Contact person:	Stian Husemoen	
Index (4)	Matterhorn, lecture-recording, OpenSource, Linux	

Number of pages: 74	Attachments: 7	Availability : Public

Short description of our bachelor thesis:

Gjøvik University College offers flexible education among other things. Along with changes in how students study, this changes the criteria of how teaching is done.

Many university colleges and universities have been developing their own solutions to record lectures. Together with these institutions, Uninett has started a project to create a joint solution for recording lectures. They started this project in April of 2010, but still there is no solution in sight.

Matterhorn is a system developed between universities all over the world. This system enables recording and distribution of lectures, but requires an administrator to manually enter all recordings in the systems schedule.

Our main goal has been to pave the way for an automatic lecture recording system.
We have developed two prototype systems, one for automatic and one for manual recording of lectures, which together with Matterhorn makes recording of lectures easier. The automatic recording system makes recordings based on data from the schools already existing schedule – TimeEdit. While the manual system makes recording possible at any time in any of the rooms where recording-equipment is available.

The automatic system is developed with PHP and overwrites Matterhorns recording-schedule. The manual system is developed with C, Bash, PHP and JavaScript and makes recordings that it later transfers to Matterhorn Server.

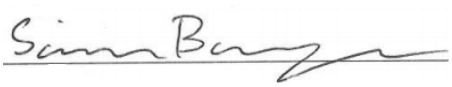
Forord

Prosjektet er et pilotprosjekt for opptak av forelesninger og er utført av to studenter ved studieretningen "Ingeniørfag – Data" og er utformet sammen med IT-tjenesten ved Høgskolen i Gjøvik som er oppdragsgiver.

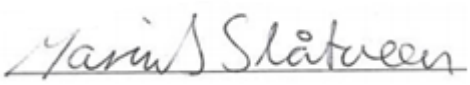
Utviklingen av systemet innebar variert arbeid og har vært svært interessant og utfordrende, både faglig og tidsmessig.

Vi vil takke veileder Kjell Are Refsvik for oppfølging og oppdragsgiver Stian Husemoen for hjelp gjennom dette spennende prosjektet. Vi vil også rette en spesiell takk til Nina Tvenge, Jon Langseth og Einar Jørgen Haraldseid.

27.05.11



Simen Rune Bragen



Marius Slåtsveen

Innholdsfortegnelse

1. Innledning 1
1.1 Problembeskrivelse 1
1.2 Målbeskrivelse 2
1.2.1 Resultatmål 3
1.2.2 Effektmål 3
1.3 Avgrensning 4
1.4 Målgruppe 4
1.6 Gruppens bakgrunn og kompetanse 5
1.7 Arbeidsmetoder 6
1.7.1 Utviklingsmodell 6
1.7.2 Organisering av arbeid 7
1.7.3 Ansvarsforhold og roller 8
1.8 Organisering av rapporten 8
2. Kravspesifikasjon 10
2.1 Introduksjon 10
2.1.1 Krav til systemet 10
2.1.2 Systemets omgivelser 10
2.2 Brukerbeskrivelse 11
2.2.1 Systemets brukere 11
2.2.2 Funksjon 12
2.2.3 Operasjon 13
2.3 Funksjonell spesifikasjon 15
2.3.1 Operasjonelle krav 15
2.3.2 Funksjonelle krav 18
2.4 Begrensninger 25
2.5 Aspekter omkring livssyklus 26
2.5.1 Dokumentasjon 26
2.5.2 Utvidelser 26
2.6 Aspekter omkring installasjon 27

2.6.1 Installasjon 27
2.6.1 Implementering 27
2.6.2 Opplæring 28
3 Design 29
3.1 Overordnet systemarkitektur 29
3.1.1 Arkitekturbeskrivelse 29
3.2 Systemets moduler 31
3.2.1 Automatisk opptak 31
3.2.2 Manuellt opptak 33
3.2.3 E-postsending 34
4 Implementering 36
4.1 Utviklingsmiljø 36
4.1.1 Ubuntu Linux 36
4.1.2 Gedit 36
4.1.3 GCC 37
4.1.4 GStreamer 37
4.1.5 Eclipse 37
4.1.6 Apache 38
4.1.7 SSH 38
4.1.8 Avidemux 38
4.2 Bruk av andres arbeid 39
4.2.1 Matterhorn 39
4.3 Implementering av modulene 40
4.3.1 Versjoner 40
4.3.2 Kommunikasjonen 41
4.3.3 Selve opptaket 44
4.3.4 Generering av opptakstimeplan 45
5 Testing og kvalitetssikring 47
5.1 Introduksjon 47
5.2 Utførte tester 47
5.2.1 Generere timeplan 47
5.2.2 Bilde fra kamera 48

5.2.3 Bilde fra projektor 50
5.2.4 Automatisk e-post 50
5.2.5 Informasjon om filene 50
5.2.6 Streaming 51
5.2.7 Test i auditorium 52
5.2.8 Ressursbruk 52
5.2.9 Stresstester 53
5.2.10 Framtidige tester 53
6 Avslutning 55
6.1 Evaluering 55
6.2 Evaluering av gruppearbeid 56
6.2.1 Innledning 56
6.2.2 Organisering 56
6.2.3 Fordeling av arbeid 58
6.2.4 Prosjekt som arbeidsform 58
6.3 Videre arbeid 59
6.3.1 Våre forslag til endringer av / tillegg til nåværende system 59
6.3.2 Naturlige utvidelser av vårt arbeid 62
6.3.3 Hva trengs for å ta et komplett system for opptak og streaming av forelesninger i bruk? 63
6.3.4 Matterhorn i fremtiden 67
6.4 Konklusjon 68
7 Litteraturliste 70
7.1 Figurliste 73
8 Vedlegg 74

1. Innledning

Høgskolen i Gjøvik tilbyr blant annet fleksible studier innen ingeniør-[1], økonomi-[2], informatikk-[3], medieteknikk-[4] og sykepleie-utdanning[5]. Slike studier tilbys hovedsakelig personer som allerede er i jobb, men mangler papirer på kompetansen sin. Denne utdanningen gjør det mulig for folk å ta utdanning mens de er i full jobb. Disse følger da studiene sine i stor grad over internett.

UNINETT leverer nettilknytning og støttetjenester til utdannings- og forskningsinstitusjoner i Norge[6]. I Uninetts 5-årsplan, fra 2011 til 2015, får opptak av forelesninger stor oppmerksomhet, men dette prosjektet er fortsatt i startgropen selv om det ble startet i april i 2010[7] [8].

Matterhorn er et opensource-prosjekt som utvikles i samarbeid mellom universiteter og høgskoler over hele verden[9]. Utviklingen har til nå resultert i et system som gjør det mulig for institusjoner å ta opp og distribuere forelesninger[10].

Matterhorn har mange funksjoner, men grunnlaget for et opptak er at en administrator manuelt registrerer alle opptak som skal gjøres en viss tid i forkant av opptaket. Hvis man ønsker automatiserte opptak, eller å gjøre opptak som ikke er timeplanlagt kan man ikke bruke Matterhorn helt uten videre[10].

1.1 Problembeskrivelse

Hvordan studenter studerer, endrer seg kontinuerlig. Det skyldes både hvordan studiene legges opp fra skole/universitet og hvordan samfunnet generelt utvikler seg[11].

For at fjernstudenter som følger studier som for eksempel fleksible utdanninger, skal få like god undervisning og oppfølging som studenter på campus, trenger man gode tekniske hjelpemidler. Blant annet sanntids lyd/video-streaming og lyd/video-opptak, blir brukt sammen med

forskjellige Learning Management System'er[l] (LMS)[12].

Også campus-studenter benytter seg av hjelpemidlene som i første rekke er tiltenkt fjernstudenter[13]. For noen kan dette dreie seg om å se/høre undervisningen en gang til og få mer oppfølging. For andre kan det være at de jobber deler av tiden da de har ordinær undervisning. Altså blir denne type undervisning viktigere og viktigere.

Til tross for at fleksible undervisningsmetoder blir stadig mer etterspurt, finnes det flere utfordringer med denne typen undervisning:

Dersom man holder samme undervisning og samme oppfølging for både campus- og fjernstudenter, fører det til at man må både holde sin ordinære forelesning og følge opp campusstudentene. I tillegg til at man enten holder en sanntidsforelesning over internett, eller lager ett lyd-/video-opptak for fjernstudentene for så å følge opp dem.

Mangel på tekniske ferdigheter i forhold til programvare og utstyr kan også gjøre det vanskelig å gjennomføre en slik undervisning.

Større arbeidsmengde og teknisk kompleksitet kan sammen skape motvilje mot å legge til rette for et slikt undervisningstilbud.

1.2 Målbeskrivelse

Målet med denne oppgaven er å utvikle et system for opptak av forelesninger. Systemet skal ikke settes i produksjon, men brukes for å se hvor de store utfordringene ligger i utvikling og bruk av et slikt system.

[l] LMS – eksempelvis Fronter og itslearning

1.2.1 Resultatmål

Systemet skal være et pilotprosjekt for opptak av forelesninger og vil bestå av automatiserte og manuelle opptak.

Automatisert opptak: Ved å knytte eksisterende systemer ved skolen sammen med Matterhorn skal vi gjøre det mulig med automatiserte opptak av forelesninger.

Manuelt opptak: Vi skal lage et system som lar foreleser starte og stoppe opptak av sine forelesninger uavhengig av tid og timeplanlegging. Dette skal gjøres ved å lage et system som tar opptak og overfører disse til Matterhorn.

Det skal utvikles en modul som etter hvert opptak sender en e-post til foreleseren som har gjort opptaket. Denne e-posten skal inneholde navnet på opptaket og en nettadresse som peker til opptaket.

1.2.2 Effektmål

Gjøre det langt enklere og billigere[14] for HiG som organisasjon, og alle foreleserne å legge til rette for fleksible studietilbud ved å automatisere forelesningsopptak. Dernest å gi muligheter til alle studenter å følge forelesninger på en mer fleksibel måte. Dette skal være en løsning som kan brukes parallelt med løsningene som enkelte forelesere benytter seg av allerede.

1.3 Avgrensning

Oppgaven vil være et pilotprosjekt, det vil si at vi vil fokusere på å få opp et system som utfører de spesifiserte oppgavene, men som ikke nødvendigvis er klart for produksjon/utrulling. Systemet vil i hovedsak bli testet i ett auditorium på skolen, men vi vil se på skalerbarheten etter endt testing. Vi kommer ikke til å legge stor vekt på kvaliteten på lyd og bilde under gjennomføringen av dette prosjektet, men heller bruke tid på slutten og reflektere og kommentere i rapporten over hva vi tror gir best resultat på området.

Systemet vårt vil gjøre opptak av forelesninger for deretter å gjøre de tilgjengelige i etterkant. Det vil ikke gi mulighet til å videreformidle forelesningen i sanntid, men det vil legges til rette for integrering av dette i ettertid.

I forbindelse med et slikt prosjekt er det en rekke komplekse personvern- og opphavsrettslige problemstillinger som må tas hensyn til. Vi vil ikke legge vekt på dette, men mener at det er et stort og vanskelig tema som bør håndteres separat som en oppfølging til det vi gjør.

Med andre ord skal vi kun se på de tekniske aspektene rundt opptaksløsningene.

1.4 Målgruppe

Oppgaven gjøres for oppdragsgiver.

Rapporten skrives for sensor, IT-avdelingen, veileder og andre som skulle være interesserte i temaet. Rapporten kan også være av interesse for de som arbeider med fleksible utdanningsspørsmål ved skolen, spesielt deltakerne i SAK-prosjektet[15].

Dersom noen skal videreutvikle systemet eller ta i bruk deler av det, vil rapporten være sentral for dem.

1.6 Gruppens bakgrunn og kompetanse

Begge gruppens medlemmer går Bachelor i ingeniørfag - data. I mellom gruppens medlemmer har vi tidligere jobbet med utviklingsmiljøene og programmeringsspråkene som kreves for denne oppgaven. Allikevel må vi sette oss inn i en del funksjonaliteter og temaer som ingen av oss har erfaring med.

Noen eksempler:

Begge har programmert i C/C++ før, men nå skal vi lage en prosess som skal kjøre i bakgrunnen (daemon), dette har ingen av oss jobbet med før.

Kommunikasjon mellom daemonen og resten av systemet skal foregå med PHP, denne kommunikasjonen er ukjent for oss.

Vi trenger å jobbe mye i bash, dette har vi jobbet litt med før, men aldri så omfattende. Dette kan bli utfordrende siden bash har en helt annen syntaks enn det vi er vant med.

I tillegg har gruppen begrenset kjennskap til Linux/Ubuntu fra før. Dette krever også en god del tilvenning.

For å utvikle de to løsningene, er vi og nødt til å bli kjent med hvordan Matterhorn fungerer. Dette er spesielt viktig for å få et system som fungerer godt.

1.7 Arbeidsmetoder

Her kommenterer vi valg av utviklingsmodell og hvordan vi organiserer arbeidet.

1.7.1 Utviklingsmodell

For å kunne gjennomføre prosjektet på en god og oversiktlig måte er det viktig å velge riktig utviklingsmodell. Vi diskuterte flere modeller og veide fordelene og ulempene mot hverandre før vi tok et valg.

En av modellene vi vurderte var "eXtreme Programming". Denne modellen fokuserer på at koden skal være best mulig. All koding skjer i lag på to programmerere[16]. Dette fører til bedre kode siden flere løsninger kan diskuteres fortløpende. All kode deles mellom alle deltagerne i prosjektet, og dersom en utvikler finner noe som kan forbedres skal den delen av koden skrives om med en gang[16]. Før utviklingen startes, designes det tester. For at en del av programvaren skal slippes må den bestå alle testene[16]. Vi har begrenset med tid, og det er relativt enkelt å teste det vi skal utvikle manuelt, så dette passer oss dårlig. Et annet minus med denne modellen i forhold til vår oppgave er at det, i XP ikke skal designes for endringer fram i tid[16]. Vi skal jobbe med noe som er ganske ukjent for oss, og vi må regne med flere endringer under veis. Denne modellen legger også opp til tett samarbeid med oppdragsgiver. Vi har kontor like ved oppdragsgiver, så dette passer oss bra. Hovedgrunnen til at vi ikke valgte XP som utviklingsmodell er at det ville medføre lite klare frister og at vi i verste fall kunne endt opp uten noe ferdig produkt i mai. Siden vi har en absolutt frist 27. mai ville dette vært meget uheldig.

En annen kjent modell er fossefallsmodellen. Her gjør man seg helt ferdig med en fase før man går videre til neste.[17] Fordelen er at man har en konkret plan på hva som skal gjøres, og når det skal gjøres. Det vil da være lett for oppdragsgiver å følge utviklingen og hvor langt vi har kommet. Siden planen er fastsatt tidlig vil det være vanskelig å tilpasse seg nye krav fra oppdragsgiver. Igjen er det et problem at vi har lite oversikt over dette fagområdet, og vi må

være forberedt på designendringer i løpet av perioden.

Vi endte til slutt på inkrementell utviklingsmodell. I denne modellen deles systemet inn i flere moduler som utvikles og testes hver for seg før man går videre til neste modul.[18] Vanligvis leveres en modul så fort den er ferdig utviklet og testet. Det kommer ikke vi til å gjøre. Derimot kommer vi til å sette den i drift på vårt testsystem. På den måten kan vi tilpasse oss nye krav som måtte komme.

1.7.2 Organisering av arbeid

Arbeidet blir gjort i hovedsak sammen på grupperommet på skolen, men kan også i spesielle tilfeller foregå hver for oss på hjemmekontor.

Systemet er delt opp i moduler og vi arbeider med en modul av gangen. Vi vil jobbe sammen om mye, mens noe av utviklingen vil skje parallelt. Dersom vi utvikler parallelt vil vi diskutere forskjellige løsninger på problemer i det de skulle dukke opp. Forskjellige overordnede oppgaver vil fordeles etter skjønn (blant annet under Ansvarsforhold og roller), mens arbeidet med selve utviklingen vil vi prøve å dele likt på gruppemedlemmene.

Vi har fast arbeidstid på skolen fra 0900 til 1500, der noen timer utgår grunnet annen undervisning. Unntak fra dette avtales gruppemedlemmene imellom.

For disponering av tiden i prosjektperioden, se vedlagt Gantt-skjema. Vedlegg A.

1.7.3 Ansvarsforhold og roller

Marius Slåtsveen er gruppeleder og har overordnet ansvar for fremdriften og at fremdriftsplanen følges.

Simen Bragen har ansvar for å opprette og oppdatere prosjekthjemmesiden jevnlig.

Begge har rett til å signere på vegne av gruppen.

IT-avdelingen på HIG, ved Stian Husemoen er oppdragsgiver.

Kjell Are Refsvik (IMT) er veileder.

1.8 Organisering av rapporten

Rapporten følger retningslinjene for større studentoppgaver gitt av Høgskolen i Gjøvik og IMT, og er delt inn i følgende kapitler:

Kapittel 1. Innledning

Gir oversikt over blant annet problemområdet, målbeskrivelsen og avgrensninger. I tillegg sier innledningen litt om gruppa og hvordan vi planla arbeidet.

Kapittel 2. Kravspesifikasjon

Dette kapittelet beskriver forskjellige krav til systemet og hvem dets brukere er. I tillegg inneholder det overordnede og detaljerte use case'er for de forskjellige hendelsene som oppstår under bruk.

Kapittel 3. Design

Her beskrives systemets arkitektur og modulene som til sammen utgjør systemet som er utviklet.

Kapittel 4. Implementering

I dette kapittelet beskrives utviklingsmiljøene vi har jobbet i og selve implementeringen av hver enkelt modul. Det nevnes også hva vi har brukt av eksisterende arbeid.

Kapittel 5. Testing og kvalitetssikring

Her forklares de forskjellige testene vi utførte underveis i utviklingsperioden og hvilke problemer vi støtte på.

Kapittel 6. Avslutning

Vi gjør en evaluering av arbeidet vi har gjort og kommer med våre tanker rundt videre utvikling av et slikt system. I tillegg kommenterer vi gruppearbeidet og hvordan det har vært å jobbe med prosjekt som arbeidsform.

Kapittel 7. Litteraturliste

Kapittel 8. Vedlegg

Vi har valgt å forklare visse ord og uttrykk direkte i teksten eller ved hjelp av fotnoter, mens andre ord og uttrykk kun er forklart i ordlisten. Denne er plassert som Vedlegg A i kapittel 8.

2. Kravspesifikasjon

Kravspesifikasjonen inneholder en beskrivelse av hvilke krav som stilles til systemet og hva som forventes av de ulike brukerne. Kravene er utarbeidet av oppdragsgiver, veileder og oss selv. Oppdragsgiver har stilt krav ut i fra en drifters synspunkt, mens veileder har sett det mer fra en sluttbrukers perspektiv.

2.1 Introduksjon

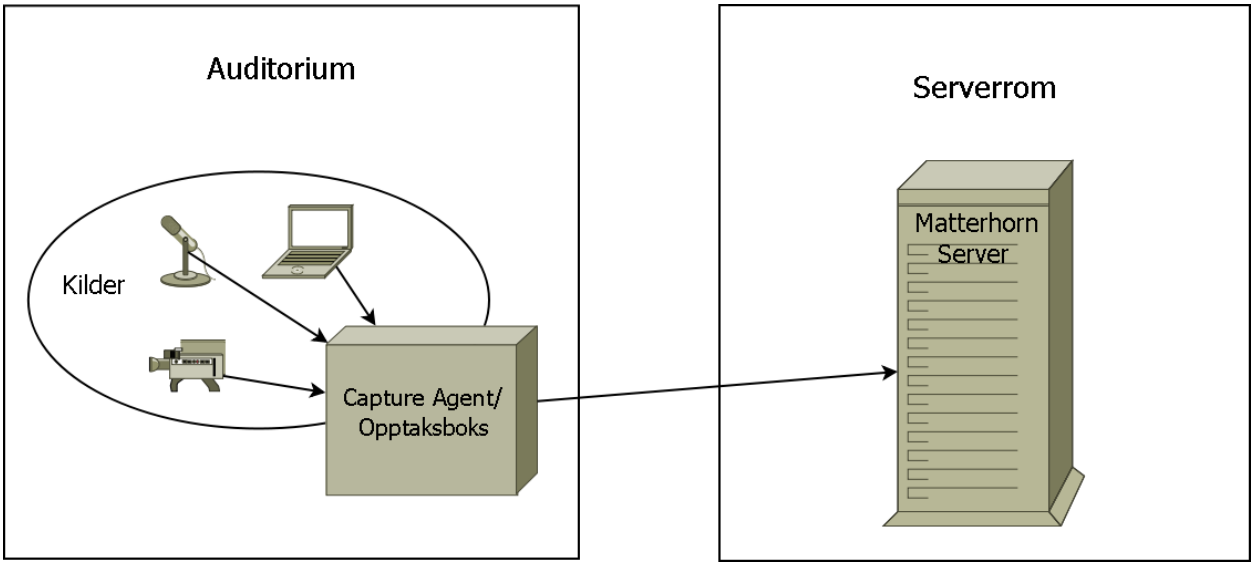
2.1.1 Krav til systemet

Vårt system skal fungere som en utvidelse til Matterhorn. Hovedpoenget med systemet er at det skal være enkelt og lite tidkrevende å gjøre opptak av forelesninger. Det skal genereres en opptaksplan automatisk etter skolens timeplan, i tillegg til at det skal være mulig å ta opptak utenom dette manuelt. Løsningen skal, i likhet med Matterhorn, støtte flere ulike typer kameraer.

2.1.2 Systemets omgivelser

Løsningen for manuelle opptak skal kjøre direkte på en datamaskin i det aktuelle rommet. Denne skal ha Ubuntu Linux 9.10, Apache og PHP5 installert. Grunnen til at vi har valgt Ubuntu Linux er at det er det som er anbefalt i installasjonsguiden til Matterhorn[19] og det gjør det enkelt med tanke på drifting at man kan logge seg på enheten via SSH utenfra.

Løsningen som genererer opptakstimeplan og løsningen som sender ut e-post til foreleser skal ligge på en server sammen med serverdelen til Matterhorn. Her må det være installert PHP5.



Figur 1. Systemets omgivelser

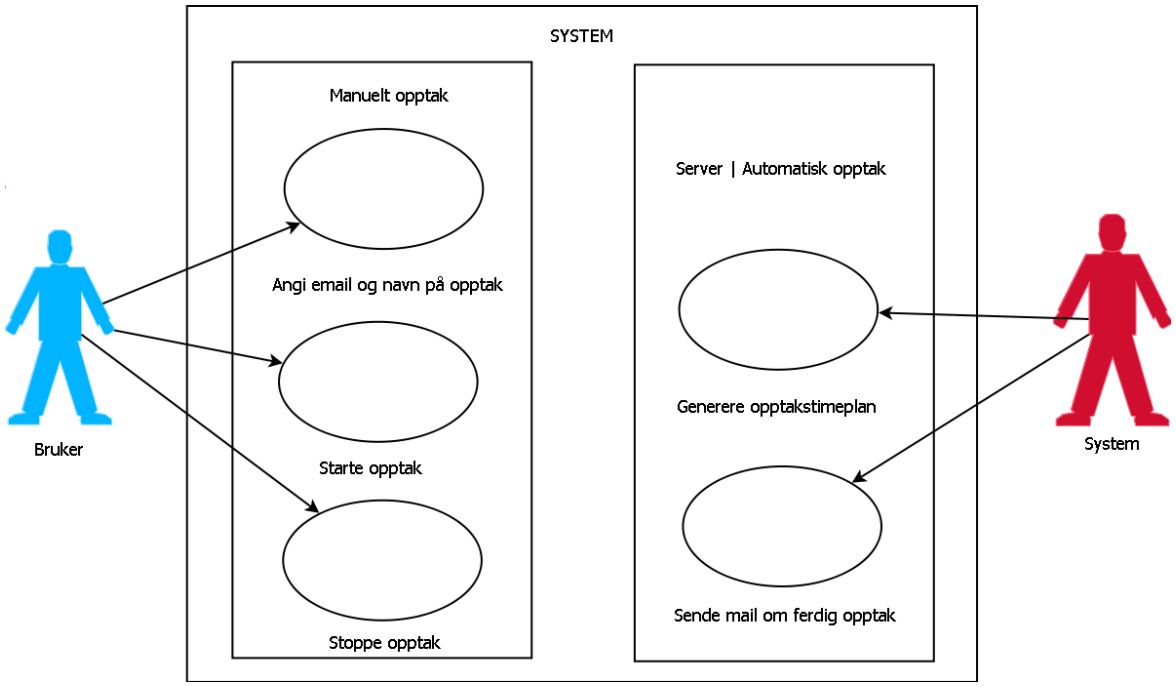
2.2 Brukerbeskrivelse

2.2.1 Systemets brukere

Forelesere vil bruke det manuelle systemet for å starte og stoppe opptak. De skal kunne logge seg på via en datamaskin som er tilgjengelig i rommet de foreleser i eller sin egen bærbare datamaskin/smarttelefon.

IT-tjenesten på skolen vil være ansvarlig for installering og konfigurasjon av systemet. Vi vil benytte oss av konfigurasjonsfiler for å enkelt kunne tilpasse systemet til ulike emner og utstyr.

2.2.2 Funksjon



Figur 2. UseCase - diagram

Use case'ene blir nærmere beskrevet i neste delkapittel.

2.2.3 Operasjon

Overordnet use case-beskrivelse:

Use case	Generere timeplan
Aktør	System/Server
Mål	Generere opptakstimeplan ut ifra data i skolens timeplan, TimeEdit.
Beskrivelse	Et shell kjører et PHP-script etter et gitt tidsintervall. Dette PHP-scriptet leser konfigurasjonsfiler som ligger på serveren og henter på bakgrunn av dette data fra TimeEdit. Scriptet genererer en opptakstimeplan fra disse dataene.

Use case	Angi navn på opptak og e-post
Aktør	Bruker/Opptaksboks
Mål	Angi hva opptaket skal hete og hvilken e-post URL/embedkode skal sendes til når opptak er ferdig behandlet.
Beskrivelse	Bruker skriver inn navn og e-post i de respektive feltene i brukergrensesnittet. JavaScript sjekker om navn og e-post er gyldige. Hvis de er gyldige, aktiveres knappene. Hvis ikke gis det beskjed og man kan ikke starte opptak.

Use case	Starte opptak
Aktør	Bruker/Opptaksboks
Mål	Starte opptak med kilder som er angitt i rommets konfigurasjonsfil
Beskrivelse	Bruker trykker på startknappen. Systemet begynner da å gjøre opptak fra de kildene som er koblet til opptaksboksen.

Use case	Stoppe opptak
Aktør	Bruker/Opptaksboks
Mål	Stoppe opptak og sende det til server for videre behandling
Beskrivelse	Bruker trykker på stoppknappen. Opptaket stoppes og sendes sammen med nødvendig metadata til server for videre behandling.

Use case	Sende e-post med URL til opptak
Aktør	System/Server
Mål	Gi foreleser direkte tilgang til opptak
Beskrivelse	Systemet sjekker med jevne mellomrom om det er kommet nye opptak inn til serveren. Dersom nye opptak blir oppdaget, henter systemet ut info om forelesers e-postadresse fra metadata-filene som ligger med opptaket og hvis nødvendig Uninetts LDAP-server. Systemet sender så ut e-post med URL og/eller embedkode til opptaket.

2.3 Funksjonell spesifikasjon

I dette kapittelet beskriver vi hvordan systemet skal virke i detalj.

2.3.1 Operasjonelle krav

2.3.1.1 Ytelse

På serversiden skal det benyttes Matterhorn, så her vil ikke vår egenutviklede programvare utgjøre noen særlig forskjell med tanke på ytelse. Det blir kun noen enkle script som sender ut e-post når nye opptak er registrert, og disse vil bruke minimalt med ressurser. Matterhorn i seg selv bruker ganske mye systemressurser når det bearbeider opptakene og genererer Flash-filer[20]. Dette skal ikke være noe problem, siden det skal kjøres på en server på skolen med kraftig maskinvare.

På opptaksboksen er systembruken viktigere. Vår testmaskin har en Intel P8700 tokjerneprosessor på 2,53GHz og 4GB minne, og det er lignende maskinvare som er tenkt brukt videre for å holde kostnadene nede. Dette må vi tenke på ved valg av oppløsning og formater ved opptak. Båndbredde derimot er ikke noe problem, og det er derfor ønskelig å ta opp med lite datatap og heller komprimere filene på serveren.

2.3.1.2 Sikkerhet

Under utviklingen av dette systemet tar vi ikke hensyn til sikkerhet. Men for et system som skal settes i drift er det visse sikkerhetsaspekter som må tas hensyn til:

Foreleser bør selv kunne velge om hans/hennes forelesninger skal tas opp, og om han eller hun vil offentliggjøre opptaket. Et opptak bør også kunne slettes i ettertid dersom dette er ønskelig. Grunner til dette kan være at det blir vist rettighetsbeskyttet materiale i løpet av forelesningen,

eller at det er timer med mye interaksjon mellom foreleser og studenter, der studentene ikke ønsker å bli gjort opptak av.

Det er opp til den enkelte foreleser om materialet skal være tilgjengelig for alle eller kun for studenter som følger emnet. Altså bør det kreves pålogging for å se opptakene, eller at videoene legges ut i emnets Fronter-rom.

Når det gjelder sjekk på om materiale som lastes opp til serveren er en ekte forelesning vil det ikke være det. Studenter har dermed en mulighet til å laste opp sine egne filer, men dette krever at de vet passordet for tilkobling mot serveren. I tillegg må de sette opp en egen installasjon av Matterhorn eller selv generere alle metadatafilene, noe som er mye jobb.

2.3.1.3 Oppetid og tilgjengelighet

Vi setter ingen krav til oppetid for vårt system, men et system som blir satt i produksjon bør ha minimum 99 % oppetid og gjøre materialet tilgjengelig både internt på skolenettet og via internett utenfra.

For at brukerne skal oppleve tjenesten som pålitelig og ønske å fortsette å bruke den, er det viktig med høy oppetid. Man burde legge planlagt nedetid til perioder der aktiviteten på tjenesten er lav.

2.3.1.4 Feilhåndtering

Feil kan oppstå ulike steder i prosessen:

Enkelte reservasjoner i timeplanen mangler informasjon om foreleser og/eller emne. Da vil PHP-scriptet sette disse feltene som “undefind” for å sørge for at opptakstimeplanen

inneholder alle nødvendige punkter.

Dersom foreleser taster inn en ugyldig e-post-adresse eller for kort emnenavn vil ikke knappen for å starte opptak være aktiv.

Alle scriptene som blir kalt fra daemon skal skrive ut sitt eget navn når de starter. PHP-scriptene skal returnere “error:true” dersom noe er feil. På den måten kan en administrator se hvor en feil har oppstått dersom dette skulle skje.

Under prosesseringen og enkodingen vil eventuelle feil være synlige i Matterhorn sitt administratorbrukergrensesnitt. Det samme gjelder for feil ved automatiske opptak basert på timeplanen.

2.3.2 Funksjonelle krav

Usecase	Generere timeplan
Aktør	System/Server
Mål	Generere timeplan for Capture Agent fra data som ligger i TimeEdit
Beskrivelse	Systemet henter data fra TimeEdit og genererer opptakstimeplaner ut fra dette. Denne skjer ut fra et gitt tidsintervall.
Hendelsesflyt	
Aktør	**X**
1. Prosessen kjører getTEData.php som henter info fra TimeEdit for hvert rom som har en konfigurasjonsfil i "room_config"-mappa på serveren.	X
2. getTEData.php kjører så generateCalenderCode.php en gang for hvert rom. Dette genererer en kalenderfil med opptaksinfo for hvert rom. Disse filene legges tilgjengelig slik at Capture Agent'ene kan lese sin respektive kalenderfil.	X

Usecase	Angi navn på opptak og e-postadresse
Aktør	Bruker
Mål	Angi hva opptaket skal hete og hvilken e-postadresse URL/embedkode som peker til klippet skal sendes til.
Beskrivelse	Skriver inn navn på opptak og e-postadresse til foreleseren
Hendelsesflyt	
Aktør	**System**
1. Skriver inn navn på opptak	
2. Skriver inn e-post	
	Kontinuerlig: sjekker gyldighet på navn og e-postadresse
	3. Navn og e-post gyldig: aktiverer "start"-knapp
Alternativ hendelsesflyt	
3. Ikke gyldig: gir beskjed om at verdiene ikke er gyldige.	

Usecase	Starte opptak
Aktør	Bruker/Opptaksboks
Mål	Starte opptak fra de kildene som er angitt i rommets konfigurasjonsfil
Beskrivelse	Bruker trykker på startknappen. Systemet begynner å gjøre opptak fra kildene
Hendelsesflyt	
Aktør	**System**
1. Bruker trykker 'start recording'	
	2. Startknappen deaktiveres
	3. Stoppknappen aktiveres
	4. En socket mot daemon åpnes av PHP.
	5. PHP sender "start", e-post og navn på opptaket.
	6. Daemon genererer episodetittel, episodenummer og serienummer basert på opptaksnavnet og dato.
	7. Daemon starter recording.sh med tilgjengelige opptakskilder som parametere.
	8. recording.sh starter opptak og streaming basert på parameterene det får inn.
	9. recording.sh lagrer opptakets prosessID i

	filen pid.
Alternativ hendelsesflyt 5. Dersom PHP-scriptet ikke får kontakt med daemon får man en feilmelding i brukergrensesnittet.	

Usecase	Stoppe opptak
Aktør	Bruker/Opptaksboks
Mål	Stoppe opptak og sende det til server for videre behandling
Beskrivelse	Bruker trykker på stoppknappen. Opptaket stoppes og sendes til serveren sammen med nødvendige metadata-filer.
Hendelsesflyt	
Aktør	**System**
1. Bruker trykker 'stop recording'	
	2. Stoppknappen deaktiveres
	3. Websiden lastes på nytt.
	4. En socket mot daemon åpnes av PHP.
	5. PHP sender "stopp"
	6. Daemon starter stop.sh
	7. stop.sh leser opptakets prosessid fra fila pid og dreper den.
	8. Daemon starter episodeserie.sh
	9. episodeserie.sh starter episodexmlGen.php og seriexmlGen.php med episodenummer, episodenavn og serienummer som parametere.

	10. episodexmlGen genererer episode.xml
	11. seriexmlGen genererer serie.xml
	12. Daemon starter manifesto.sh
	13. manifesto.sh henter ut størrelse på filene, lengde på opptaket og MD5checksum.
	14. manifesto.sh starter manifestxmlGen.php
	15. manifestxmlGen.php genererer manifest.xml
	16. Daemon starter properties.sh
	17. properties.sh starter propertiesGen.php
	18. propertiesGen.php genererer org.opencastproject.capture.agent.properties
	19. Daemon starter zipsend.sh
	20. zipsend.sh pakker filene sammen i en zip-fil.
	21. zipsend.sh kopierer over filene til serveren.

Alternativ hendelsesflyt

9. Dersom noen av PHP-scriptene ikke får med alle parameterne de trenger, skrives det feilmelding til skjerm.

13. Dersom det har blitt feil med opptak av en av kildene vil ikke manifesto.sh klare å hente ut riktig info om filene.

21. Dersom det er satt feil IP eller passord til serveren vil filene ikke bli kopiert over.

Usecase	Sende e-post med link til opptak til foreleser
Aktør	System/Server
Mål	Gi foreleser enkel/direkte tilgang til sine opptak
Beskrivelse	System sjekker med jevne mellomrom etter nye opptak. Når nye opptak oppdages, henter systemet ut forelesers e-postadresse eller HIG-brukernavn fra metadata-filene og sender e-post med URL til foreleseren
Hendelsesflyt	
Aktør	**X**
1. System sjekker med jevne mellomrom etter nye opptak.	X
2. Nytt opptak oppdages	X
3. Forelesers identitet hentes ut og opptakets URL genereres	
4. E-post sendes til foreleser med URL til opptak og beskjed om at det er klart.	
Alternativ hendelsesflyt	
3. Dersom forelesers identitet ikke er en e-postadresse skal den være det offisielle HIG-brukernavnet. I så tilfelle slås dette opp i LDAP-serveren og e-posten hentes der.	

2.4 Begrensninger

Vår løsning vil bestå av flere ulike deler som jobber sammen. Brukergrensesnittet som starter og stopper opptak blir skrevet i HTML/CSS og JavaScript. Fordelen med å lage en web-basert løsning framfor for eksempel en Java-applikasjon er hovedsakelig tilgjengelighet. Vårt brukergrensesnitt skal kunne brukes fra alle datamaskiner med nettilgang, og de aller fleste mobiltelefoner som er i bruk i Norge i dag har en nettleser som støtter JavaScript.

Dette brukergrensesnittet legges på hver enkelt opptaksboks. Vi vil forsøke å holde oss til standardfunksjoner i PHP, så disse skal kun kreve en vanlig Apacheserver med PHP5. Det samme gjelder de andre PHP-scriptene. De blir kun PHP5, men må kunne kjøres fra kommandolinja.

Daemon skrives i C og åpner blant annet en unixsocket. Dette medfører at operativsystemet på opptaksboksen må være Unix/Linux. Vår testboks kjører Ubuntu Linux 9.10.

Shellene som startes av daemon vil være i bash. For å hente ut lengden på videofilene skal vi benytte avidemux-cli-pakken, så denne må være installert på forhånd. For opptak benyttes GStreamer, noe som er med i de fleste linuxdistribusjoner som standard.

For å støtte flest mulig opptakskilder lager vi en konfigurasjonsfil der kildenavn og type settes. Dersom oppløsningen vi har satt som standard ikke støttes, kan dette endres i pipelinen i opptaks-shellet.

Streamingserveren til skolen er en Darwin streamingserver som forventer en RTSP-strøm inn. Vår pipeline vil sende ut en udp-stream som kan sendes videre til Darwin streamingserver via for eksempel VLC.

2.5 Aspekter omkring livssyklus

Modulene som blir beregnet å kjøre på serveren skal kunne kjøre kontinuerlig uten noe behov for vedlikehold. Daemon på den lokale opptaksboksen skal også kunne kjøre i bakgrunnen uten å behøve noe vedlikehold. Dersom noe går galt under opptak bør bruker få beskjed om dette i brukergrensesnittet.

2.5.1 Dokumentasjon

For brukerne vil en ferdig utviklet opptaksløsning være svært enkel å bruke. Brukergrensesnittet skal være selvforklarende, men det burde allikevel stå en kort forklaring til hvert av feltene ved siden av. I tillegg kan det være et informasjonsark på én a4-side som forklarer hva de må gjøre og hvordan de får tilgang til opptaket i ettertid.

For drifterne på IT-tjenesten vil vi lage en installasjonsguide. Her forklarer vi hvilke verktøy som må være installert og hvilke konfigurasjonsfiler som må endres for å ta i bruk vårt testsystem.

2.5.2 Utvidelser

Vårt system er ment å være et pilotprosjekt. Vi lager en løsning som er enkel og fungerer, og som skal være mulig å utvide senere. Det er derfor viktig å være nøye med kommentering av kode. Vi har bestemt oss for å kommentere alt på engelsk, så flest mulig kan ha mulighet til å jobbe med koden vår videre. Øverst i scriptene tar vi også med informasjon om hvilke program

eller script som vanligvis kaller det enkelte scriptet, og kort hva det gjør.

2.6 Aspekter omkring installasjon

2.6.1 Installasjon

Matterhorn i seg selv krever middels kjennskap til Linux under installasjon. I tillegg inneholder vår løsning enkelte variabler som må endres i ulike script, og at PHP5 og avidemux kan kjøres via kommandolinja. Dette vil stå forklart i installasjonsguiden og de fleste med kjennskap til Linux vil klare dette.

Å gjøre et opptak er derimot veldig enkelt. Man åpner websiden og taster inn e-post, emne og trykker på startknappen, så vil opptaket starte.

2.6.1 Implementering

Vår løsning vil bli tatt gradvis i bruk. Den avhenger av at det er satt opp kamera, mikrofon og en opptaksboks i rommet som skal brukes. De lærerne som allerede i dag legger ut sitt materiale på egen måte, vil fortsatt kunne gjøre det på sin måte, eller begynne å bruke vår løsning dersom rommet er klargjort for det.

Når Matterhorn først er satt opp på skolens server, er det enkelt å tilknytte flere rom etterhvert.

2.6.2 Opplæring

Systemet er tenkt å være så enkelt å bruke at det ikke skal være behov for opplæring. Vi ser for oss at det skal henge en a4-plakat i hvert rom som er klargjort for opptak, med enkle instrukser om hvordan det gjøres. Det kan allikevel tenkes at det holdes en demonstrasjon for foreleserne om hvordan systemet virker, og hva som skjer etter at et opptak er klart.

For de som skal sette opp og drifte systemet vil vi lage en installasjonsguide og en forklaring av alle konfigurasjonsfilene. Disse vil inneholde eksempler på gyldige parametere.

Elevene som skal se på videoene trenger ingen opplæring. De vil få beskjed av foreleser om hvor videoene ligger og har da tilgang til dem på samme måte som annet fagstoff som publiseres i emnet.

3 Design

3.1 Overordnet systemarkitektur

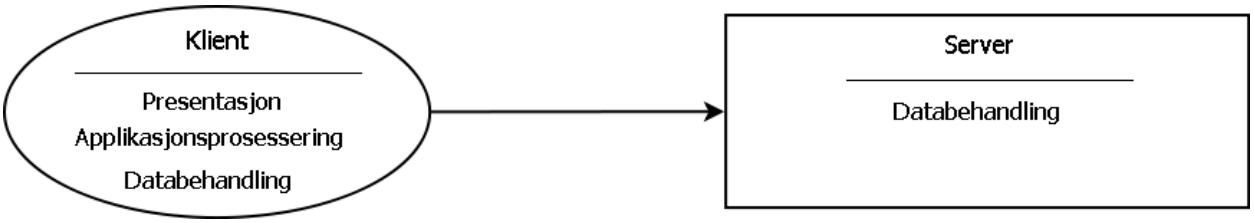
Her beskriver vi hvordan systemet er bygd opp som helhet. Det vil si systemets arkitektur og samhandlingen mellom de forskjellige modulene som utgjør systemet.

3.1.1 Arkitekturbeskrivelse

Matterhorn som system består av Capture Agent'er og Matterhorn server, der Capture Agent'ene er tykke klienter. Dette er altså en klient - server arkitektur. Vi har bygd vår automatiske opptaksløsning rundt det allerede eksisterende Matterhorn-systemet.

Det manuelle opptakssystemet kjører på den samme maskinvaren som Matterhorns Capture Agent-programvare, men er et helt separat system. På grunn av at oppgaven er omtrent den samme, systemene kjører på samme maskinvare og at serversiden skal være den samme er det naturlig å bruke den samme arkitekturløsningen. Altså er dette også en klient - server løsning med tykk klient.

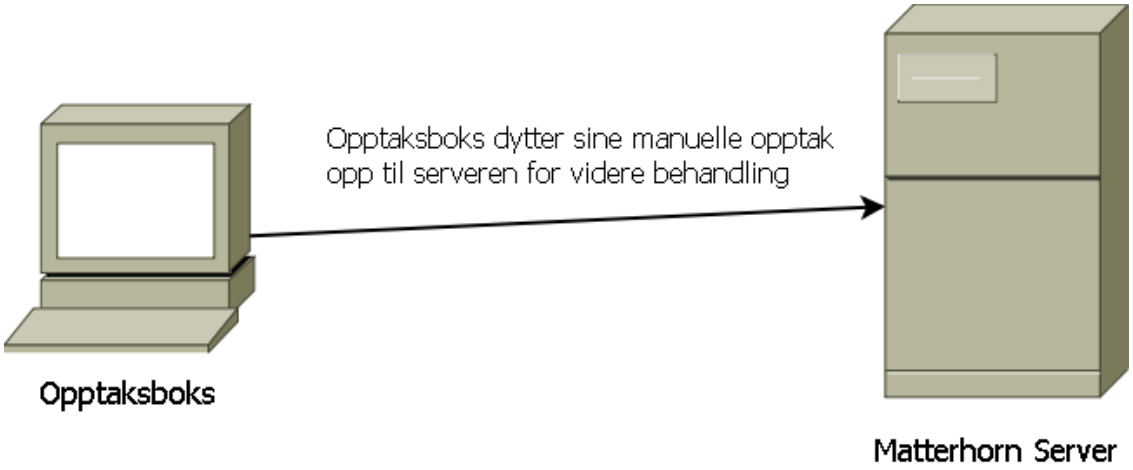
Alternativet til å bruke tykke klienter, er tynne klienter. Dette har noen fordeler, for eksempel kreves det mindre ressursbruk hos klienten da denne kun skal ta seg av presentasjon av data. I tillegg er det enklere å oppdatere et system som bruker tynne klienter da oppgraderingen kun skjer på server og ikke på alle klientene.[21]



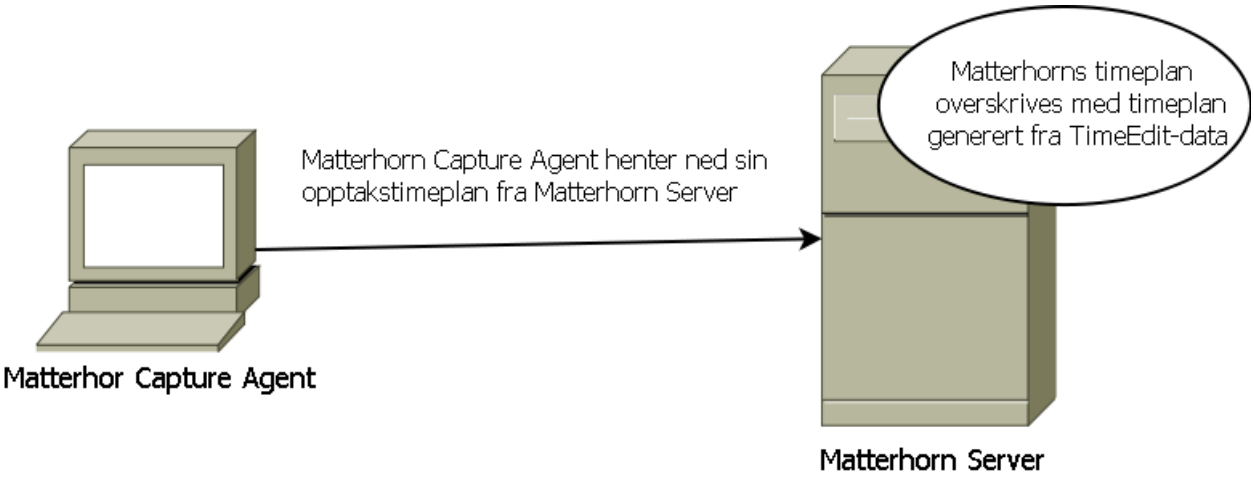
Figur 3. Tykk klient - tolags arkitektur

Klienten er tykk siden serveren krever en del tilleggsinformasjon sammen med video- og lydfilene når de blir lastet opp. For å legge ved denne informasjonen må klienten gjøre en del databehandling. Hva denne databehandlingen består i kommer vi nærmere tilbake til når vi beskriver hver enkelt modul i punkt 3.2.

Når det kommer til lagdeling er det naturlig for vår del å bruke to-lags arkitektur. Systemet vårt består av opptaksboksen med sin programvare, serveren med Matterhorn-programvaren og programvaren for sending av e-post. Lagring av forelesningsopptakene kan være på serveren med Matterhorn-programvaren eller på en annen server, men dette endrer ikke lagdelingsmodellen.



Figur 4. Opptaksboks og Matterhorn Server ved manuelle opptak



Figur 5. Capture Agent og Matterhorn Server ved automatisk opptak

3.2 Systemets moduler

Systemet er delt inn i tre hovedmoduler: Automatisk opptak, Manuelt opptak og E-post-sending. De blir nærmere beskrevet her.

3.2.1 Automatisk opptak

Beskrivelse av modulen:

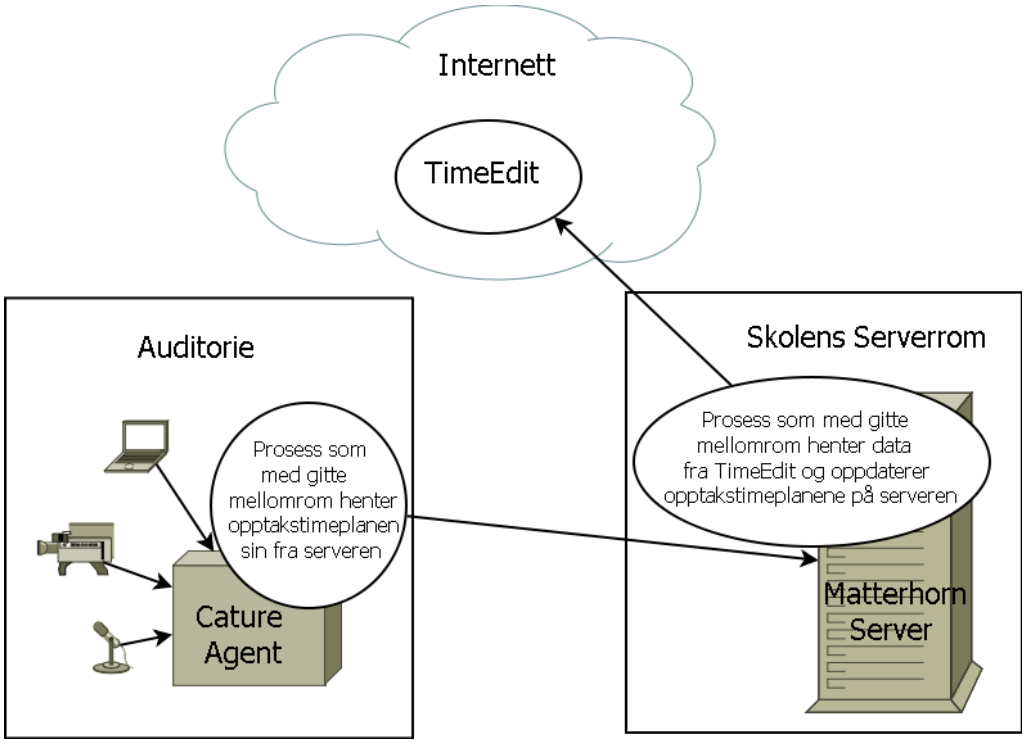
Denne modulen skal hente data som ligger i skolens eksisterende timeplan – TimeEdit, og ut ifra dette generere en opptakstimeplan som Matterhorns Capture Agent'er kan lese.

Prosessering/data:

Modulen leser først inn navnet på rom-konfigurasjonsfilene som ligger på serveren. Hvert rom som det er opptaksutstyr i, har en konfigurasjonsfil i denne mappen som er oppkalt etter navnet til rommet. I denne filen finner systemet URL'en til rommets TimeEdit-timeplan. Denne konfigurasjonsfilen inneholder også informasjon om rommets opptaksutstyr.

Deretter lastes innholdet i rommets TimeEdit-timeplan ned og opptakstimeplan for rommet genereres utfra dette og rom-konfigurasjonsfilen. Dette gjøres for alle rommene som har en konfigurasjonsfil på serveren. Opptakstimeplanen lagres i Matterhornserverens timeplanmappe der Capture Agent'ene henter sin opptakstimeplan.

På denne måten genereres timeplan for automatiske opptak som overskriver Matterhorns manuelle timeplan, dette gjentas med faste mellomrom.



Figur 6. Dataflyt i det automatiske opptakssystemet

3.2.2 Manuellt opptak

Beskrivelse av modulen:
Den manuelle opptaks-modulen er en egen prosess som skal gjøre det mulig og starte og stoppe opptak via et brukergrensesnitt som gir brukeren mulighet til å legge inn navn på opptaket i tillegg til å skrive inn en e-postadresse. Når opptaket er stoppet skal modulen hente ut nødvendig data fra lyd/videofilene og overføre dette sammen med lyd/videofilene over til Matterhornserveren.

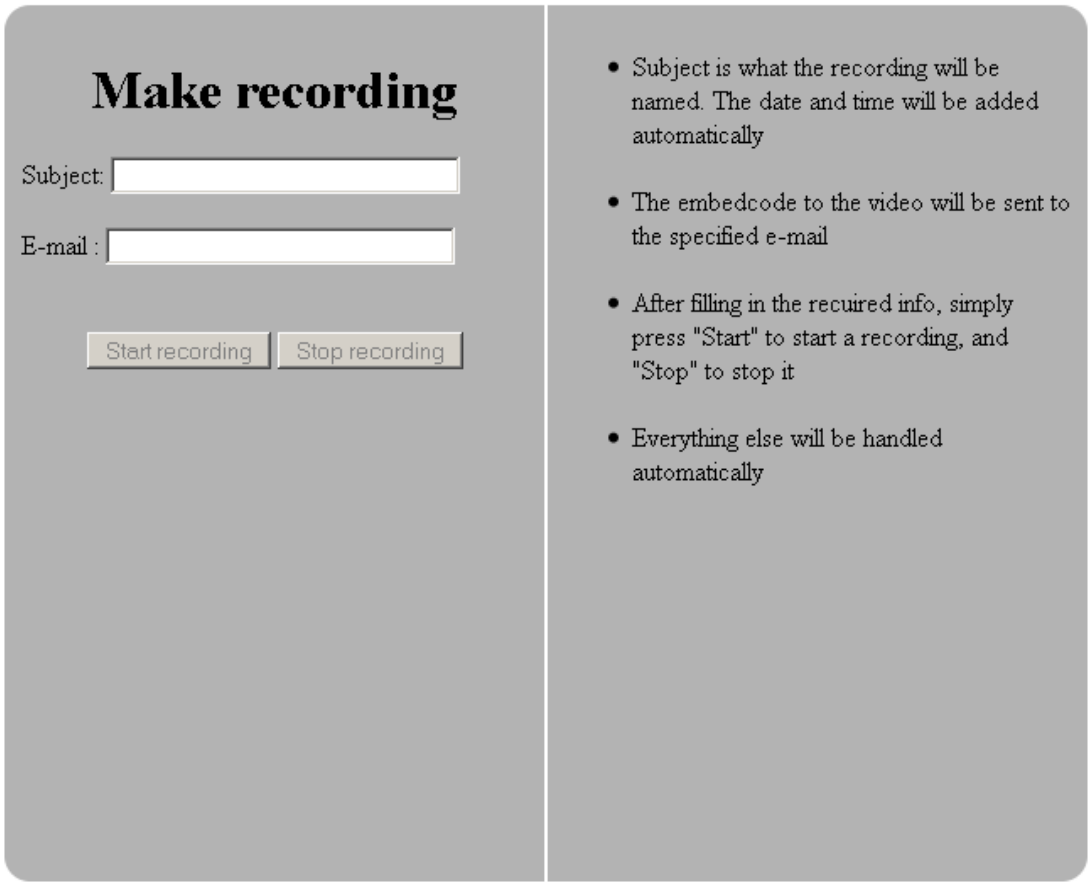
Prosessering/data:
Når opptaksprosessen startes, typisk ved maskinoppstart, henter den inn en del informasjon fra en konfigurasjonsfil som blant annet inneholder navnet på rommet maskinen er plassert i, og informasjon om opptakskildene som er koblet til maskinen.

Bruker skriver inn navn og e-postadresse og trykker start. Startsignal sendes til opptaksprosessen. Opptaksprosessen sørger for at opptaks-pipeline'en kjører.

Når brukeren trykker på stopp sendes stoppsignal sammen med navnet på opptaket og e-postadressen til opptaksprosessen. Denne sørger for å drepe prosessen som kjører opptaks-pipeline'en. Deretter hentes informasjon som lengde på opptaket, MD5 checksum også videre fra lyd/videofilene.

Metadatafiler genereres ut fra informasjonen om opptakskildene og lyd/videofilene. Når de er ferdig generert blir alle filene komprimert sammen i en zip-fil og overført til Matterhornserveren.

Brukergrensesnitt:



Figur 7. Web-brukergrensesnitt for testing av det manuelle systemet

3.2.3 E-postsending

Beskrivelse av modulen:

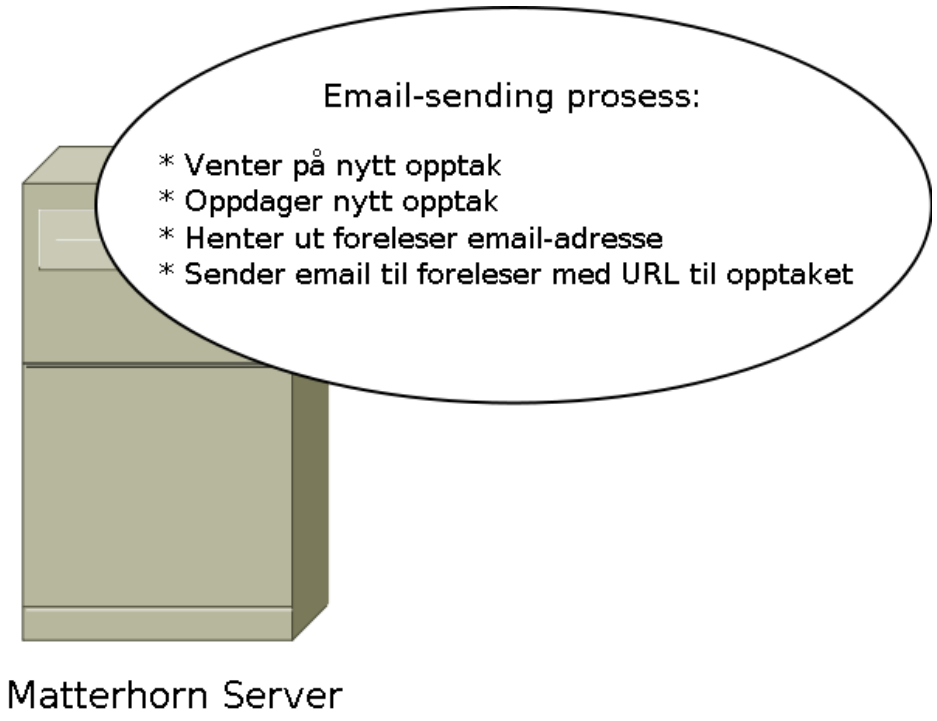
Når opptaksboksen har overført et nytt opptak til en bestemt mappe på serveren, sender modulen e-post til den foreleseren som har holdt forelesningen. Dette skjer uansett om opptaket er gjort manuelt eller automatisk. E-posten inneholder navnet til opptaket og en URL,

slik at foreleseren kan gjøre opptaket tilgjengelig for studentene.

Prosessering/data:

Modulen ser etter nye opptak. Når den oppdager ett nytt opptak, henter den ut informasjon fra metadatafilene om hvem som eier/har opprettet opptaket. Hvis opptaket kommer fra den automatiske løsningen, kontakter modulen LDAP-serveren og finner foreleserens e-postadresse der. Hvis opptaket kommer fra den manuelle løsningen, så ligger allerede e-postadressen i metadatafilene.

Når e-postadressen er funnet, genererer systemet URL'en til opptaket og sender dette i e-posten sammen med en beskrivelse av opptaket.



Figur 8. E-post sending skjer etter Matterhorn Server har oppdaget et nytt opptak er klart

4 Implementering

I denne delen av rapporten beskriver vi hvilke programmer og verktøy vi har brukt under utviklingen av systemet. Vi vil også komme inn på beskrivelse av modulene og kodeeksempler.

4.1 Utviklingsmiljø

4.1.1 Ubuntu Linux

På både server og opptaksboks har vi brukt Ubuntu Linux 9.10 som operativsystem. Dette er anbefalt av Opencast-gruppen. De offisielle installasjonsscriptene er skrevet for Ubuntu 9.10[22], så vi valgte dette for ikke å bruke for mye tid på å tilpasse script og for å kunne benytte oss av tilgjengelig dokumentasjon. Vi har brukt Ubuntu i faget "Operativsystemer" tidligere, så vi hadde litt kjennskap til det fra før. Utfordringen lå i å sette det opp riktig og forstå hvilke tilleggspakker vi trengte.

4.1.2 Gedit

Det meste av C og bashkode er skrevet i Gedit. Det er en teksteditor som er standard i Ubuntu. Det er en ganske enkel editor, men den har støtte for diverse programmeringsspråk, så syntaksen blir fargekodet. Dette hjelper veldig når man sitter og koder. Gedit hadde noen svakheter, blant annet gjenkjente det ikke regulære uttrykk i bash, og variabler ble vist feil på skjermen.

4.1.3 GCC

For å kompilere C-koden har vi brukt Gnu Compiler Collection. Det er en kompiler som kjøres fra oppgavelinjen. Det fungerer helt greit, da vi får både advarsler og feilmeldinger som i et fullt utviklingsmiljø. Det var litt tungvint å måtte kompilere manuelt etter alle endringer, men alt i alt gikk det helt fint.

4.1.4 GStreamer

Selve opptakene gjøres via GStreamer. Det er et multimedie-rammeverk som følger med i GNOME. For å ta opp med GStreamer skriver man pipelines i enten C eller direkte i terminalen/bash.

Ingen av oss hadde brukt dette tidligere, så vi var nødt til å sette oss inn i hvordan det fungerte før vi begynte å lage våre egne pipelines. Vi har funnet lite dokumentasjon på hvordan man bygger opp slike pipelines, så vi bestemte oss for å lage en enkel guide som vi har lagt ut på prosjekthjemmesiden vår så andre skal slippe å bruke like lang tid på å prøve å feile som vi gjorde.

4.1.5 Eclipse

For PHP har vi brukt Eclipse som rammeverk. Eclipse er gratis og har åpen kildekode. Å bruke et slikt rammeverk gjør det en del enklere å teste modulen underveis, da scriptet kan kjøres direkte, uten å gå via terminalen. En av oss hadde brukt Eclipse tidligere, og det var naturlig å

velge dette siden vi hadde kompetanse på det fra før i gruppa.

4.1.6 Apache

Som webserver har vi valgt å bruke Apache. Hovedsakelig fordi det er en av de mest utbredte webserverne[23], og at det i utgangspunktet er utviklet for Linux/Unix, men også fungerer på Windows.

4.1.7 SSH

For å kopiere filer mellom linuxmaskinene har vi brukt "scp" via SSH. Dette er omtrent som å kopiere en fil lokalt via terminalen i Linux, men før det kan brukes må det genereres nøkkelfiler på begge maskinene. Dette fungerte veldig bra, og vi sparte mye tid på å bruke det.

4.1.8 Avidemux

Lengden på opptakene henter vi ut ved hjelp av avidemux2cli. Det er et lite program for videobehandling og prosessering[24]. Fordelen med det er at det krever kun en enkel kommando for å hente ut informasjon om en mediefil.

Vi så også på andre løsninger, deriblant MediaInfo og oggInfo. Førstnevnte var for komplisert, og oggInfo var begrenset til ogg-filer. Vi endte med å ta opp video i mpeg2, så vi bestemte oss for avidemux sin kommandolinjeløsning.

4.2 Bruk av andres arbeid

På enkelte deler av løsningen vår har vi kunnet støtte oss til kode som finnes fra før.

Oppkobling til Uninetts LDAP-server og uthenting av e-postadresse var gjort av Kent Andersen på IT-tjenesten i PHP fra før, så vi har kunnet bruke deler av hans kode i en av funksjonene våre.

For å behandle iCal-eventene fra TimeEdit har vi brukt et PHP-bibliotek skrevet av Morten Fangel som er tilgjengelig her: https://github.com/fangel/SG-iCalendar

For å opprette en unixsocket har vi sett på dokumentasjon fra IBM (http://publib.boulder.ibm.com/infocenter/iseries/v5r4/index.jsp?topic=%2Frzab6%2Fuafunix.htm) og Linux Howtos (http://www.linuxhowtos.org/C_C++/socket.htm).

4.2.1 Matterhorn

Begge løsningene vi har lagd er ment for bruk sammen med Opencast Matterhorn. Den manuelle løsningen tar alle opptak selv, men bruker Matterhorn som serverløsning. Mens den automatiske løsningen i hovedsak skriver over opptakstimeplanen til Matterhorn, og lar Matterhornsystemet ta seg av opptakene.

4.3 Implementering av modulene

4.3.1 Versjoner

Automatisk opptak

Den første delen vi utviklet var automatisk opptak basert på data fra TimeEdit.
I første utkast hadde vi timeplan-generatoren direkte på opptaksboksen. Vi endret konfigurasjonen for hvor Matterhorn Capture Agent'en hentet timeplanen sin til vår lokale fil. Det fungerte ikke.

I versjon to flyttet vi generatoren over på serveren, og vi skrev et PHP-skript som tar vår kalender-fil og skriver den til /schedule/*romnavn*. Det fungerte.

Manuelt opptak

I første versjon av den manuelle opptaksløsningen startet vi ett skript for hver opptaksenhet. Vi hadde kun opptak, ingen løsning for streaming.
Vi inkluderte streaming via en icecastserver i versjon to.

I tredje versjon slo vi sammen alle opptakene til en stor pipeline. Vi bytta også ut icecastserveren med udp-streaming fra GStreamer. Denne streamen kan hentes opp av VLC og sendes videre til skolens streamingserver.

4.3.2 Kommunikasjonen

Her følger en beskrivelse av hvordan informasjonen blir sendt fra web-brukergrensesnittet, via daemon og til bash/PHP-script.

4.3.2.1 Starte opptak

Etter at e-postadresse og navn på opptak er sjekket OK, sendes informasjonen fra JavaScriptet til PHP-scriptet.

```
//ajax to PHP that communicates with the recording-daemon
  $.ajax({
     //the url to the PHP
     url: "PHP/startStopRecording.php",
     dataType: "json",
     //the data sent to PHP
     data: {start:"start", subject:sub, e-post:$("#e-post").val()},
     //if it works, outputs text
     success: function(){
        $("#info").text("Recording started!");
     },
     //if it doesn't work, outputs text
     error: function(){
        $("#info").text("Something went wrong on start.. -> ajax/PHP");
     }
```

I PHP-scriptet startStopRecording.php som startes gjøres det først en sjekk på at e-post og emne ikke er tomt. Deretter opprettes det en unix-socket:

```
$socket = socket_create( AF_UNIX , SOCK_STREAM , 0 );
```

og vi kobler til serveren som allerede er oppretta av daemon.

```
$connected = socket_connect($socket ,$address);
```

Etter dette sendes informasjonen via socketen:

```
//set string to be sent to socket
$buf = "start#" . $_GET['e-post'] . "#" . $_GET['subject'] . "#";
//send to socket
$sent = socket_send ($socket , $buf , strlen($buf) , MSG_DONTROUTE);
```

Daemon på sin side tar i mot med “recv”:

```
rc = recv(sd2, buffer, sizeof(buffer), 0);
```

Når daemon har gjort seg ferdig med å hente ut informasjon fra konfigurasjonsfilen til rommet og generert blant annet episodenummer, startes opptaks-shellet via “execl”.

```
// Starting the recording(s) in a new process
startpid1 = fork();
if (startpid1 == 0){
        // parameters: recording.sh, episode number, device1, dev1, device2, dev2, device3, dev3, NULL
        execl(argum[0], argum[0], argum[11], argum[5], argum[6], argum[7], argum[8], argum[9],
argum[10], NULL);
}
```

Før skriptet startes, kjøres det en “fork”, det vil si at programmet lager en kopi av seg selv og

skriptet kjøres som en egen prosess. Vi sjekker at startpid1=0 så det bare er den nye prosessen som utfører oppgaven.

4.3.2.2 Stoppe opptak

Når et opptak stoppes skjer mye av det samme som når det startes.
Hovedforskjellen ligger i hva som blir sendt og hva som startes av daemon.

På samme måte som i opptaksscriptet startes alle scriptene når opptaket er ferdig også som nye prosesser via “fork” og “execl”.

Først startes stop.sh som henter ut prosessid’en til opptaksscriptet og dreper det.

```
# Setting $1 from file pid
IFS=$' '
set $(cat pid)

## Stopping the processes that is started
kill $1

}
```

Daemon starter så en del script som tar seg av uthenting av informasjon og generering av metadatafiler. Her følger et eksempel på et av dem:

```
# Getting the size of the first file
size1=$(stat -c %s $3)

# Getting the MD5 checksum of the first file
MD51=$(MD5sum $3 | sed -e "s/$3//")

# Getting the duration at the form hour:minute:second.millisecond
dur=$(avidemux2_cli --autoindex --load $3 --info | grep Duration: | sed -e 's/Duration://' | sed '2 d')
```

Dette er manifesto.sh som henter ut filstørrelse, MD5 checksum og lengde på opptaket. På lengden må vi sile ut en del annen informasjon.
Etter dette startes manifestXMLGen.php som generer manifest.xml.

```
PHP -f manifestXMLGen.php $1 $2 $dur1 $pres1 $3 $video1 $size1 $MD51 ...
```

Til slutt pakkes filene til en zipfil og sendes over til serveren.

```
# Copies the media.zip file to the server
sshpass -p "$PASS" scp media.zip $USER@$SERVER_ADDRESS:/opt/matterhorn/felix/inbox
```

Etter at dette er gjort står daemon og lytter etter nye tilkoblinger.

4.3.3 Selve opptaket

I recording.sh lages pipelinen og opptaket startes.

```
# Setting the pipeline with camera, sound and vga
pipeline="gst-launch-0.10 v4l2src device=/dev/$dev1 ! videorate ! video/x-raw-
yuv,width=720,height=576,video-capture-framerate=25/1,videorate=25 ! queue ! ffenc_mpeg2video !
mpegtsmux ! tee name=videoout ! filesink location=rec/$1/$2 videoout. ! queue ! udpsink
host=128.39.82.249 port=1234 v4lsrc device=/dev/$dev3 ! videorate ! video/x-raw-yuv,width=1024,
height=768, video-capture-framerate=5/1, videorate=25 ! ffmpegcolorspace ! ffenc_mpeg2video !
mpegtsmux ! tee name=vgaout ! filesink location=rec/$1/$6 vgaout. ! queue ! udpsink
host=128.39.82.249 port=1235 alsasrc device=$dev2 ! audio/x-raw-int,rate=44100,channels=2,depth=16
! lamemp3enc ! tee name=audioout ! filesink location=rec/$1/$4 audioout. ! queue ! udpsink
host=host=128.39.82.249 port=1236"
```

Først startes GStreamer versjon 0.10 og vi oppgir at vi bruker en video4linux2 opptakskilde. Kilden er /dev/$dev1, der $dev1 er en parameter vi har fått med inn fra daemon. Så pipes dette gjennom “videorate” og oppløsning og antall bilder i sekundet settes. Videre enkoder vi

videoen som mpeg2 og splitter strømmen med “tee name=videoout”. En del av strømmen lagres som en fil på disk, mens en kopi streames via udp.

Etter dette gjøres tilsvarende for vga og lyd, men da med andre typer opptakskilder. Vga er video4linux1 og lyd er en alsakilde.

4.3.4 Generering av opptakstimeplan

Modulen for å lage timeplanen for automatiske opptak blir kjørt hvert tiende minutt. Dette tidsintervallet kan endres etter ønske.

getTEData.php finner .ini-filene for alle rom det er en opptaksboks i.

```
//globs the "room_config" directory and finds the .ini files
$fileNameArray = glob(dirname(__FILE__) . '/room_config/' . '*.ini');
```

For hver av disse hentes hendelsene fra TimeEdit. Dersom enkelte felter mangler fylles de med “undefined”. Brukernavnet følger ikke brukernavnstandarden fra LDAP-serveren, så det må oversettes ut i fra en egen fil.

Etter at dette er gjort kalles generateCalendarCode som genererer selve fila. Deler av denne består av base64 enkodede xml-filer.

```
//episode attachment:
$thisEvent .= "ATTACH;FMTTYPE=application/xml;VALUE=BINARY;ENKODING=BASE64;X-APPLE-
FILEN\r\n";
$thisEvent .= " AME=episode.xml:";
$thisEvent .= base64_encode ($episodeAttachment);
$thisEvent .= "\n\n";
```

Videre blir disse timeplanene hentet av opptaksboksen via scheduler.php.

Dette scriptet leser innholdet i timeplanen og skriver det til Capture Agent'en det får forespørsel fra.

```
//if agent is verified
if ( verify_agent($agent) )
{    //prints calendar for specific agent
  print get_calendar_for_agent( $agent );
}
```

5 Testing og kvalitetssikring

5.1 Introduksjon

I dette kapittelet tar vi for oss testene vi har utført under veis i prosjektet. Store deler av det vi har holdt på med har vært helt nytt for oss, så det har vært nødvendig å teste ofte. Å avdekke kodefeil har også vært viktige grunner til å utføre mange tester. Ut i fra resultatene vi har fått har vi gjort endringer fortløpende. Vi har naturligvis sjekket at all ny kode har fungert fortløpende, i tillegg til testene som er beskrevet her.
Testene er utført av oss selv dersom annet ikke er oppgitt.

5.2 Utførte tester

5.2.1 Generere timeplan

Først forsøkte vi å hente ut tilfeldig data fra TimeEdit-timeplanen til et rom. Dette virket bra med en gang. Når vi gikk videre til å oversette brukernavn fra TimeEdit-format til LDAP-format oppsto det problemer. Vi hadde feil med tegnsettet. Det skulle være UTF-8[II], men vårt var ASCII[III].

Neste steg vår å generere en egen .ics-fil[IV]. Å lage fila gikk fint, og alt så riktig ut. Problemet var at Matterhorn bare godtok noen av filene vi lagde. Dette virka helt tilfeldig, og vi brukte lang tid

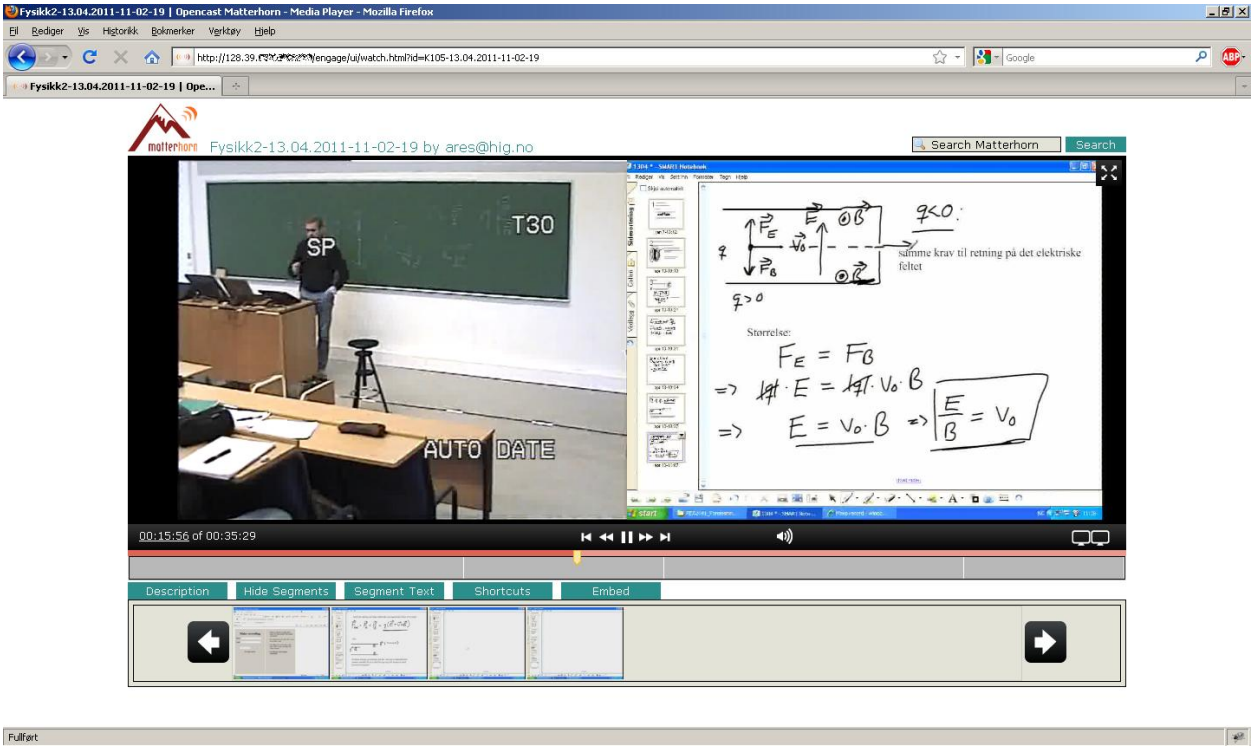
[II] 8-bit Unicode Transformation Format - Et tegnsett der hver bokstav er representert med en til fire byte. Det er bakoverkompatibelt med ANSII. - http://en.wikipedia.org/wiki/UTF-8
[III] American Standard Code for Information Interchange - Et tegnsett der hver bokstav er representert med 7 bit. Det ble forbigått av UTF-8 som det mest brukte tegnsettet på internett i 2007. - http://en.wikipedia.org/wiki/ASCII
[IV] .ics er endelsen på iCalendar-filer. Informasjonen fra skolens timeplan kan lastes ned som slike filer. Det er et standardisert format som er støttet av blant andre Google Calendar, Apple iCal og Microsoft Outlook. - http://en.wikipedia.org/wiki/ICalendar

på feilsøking av dette problemet. Etter å ha forsøkt med mange forskjellige datoer, med og uten "æøå", og diverse annet forsto vi til slutt at det hadde med linjeskift å gjøre. Vi brukte /n [mellomrom] i scriptet, mens Matterhorn krever /r/n [mellomrom]. Etter denne fiksen har det fungert bra.

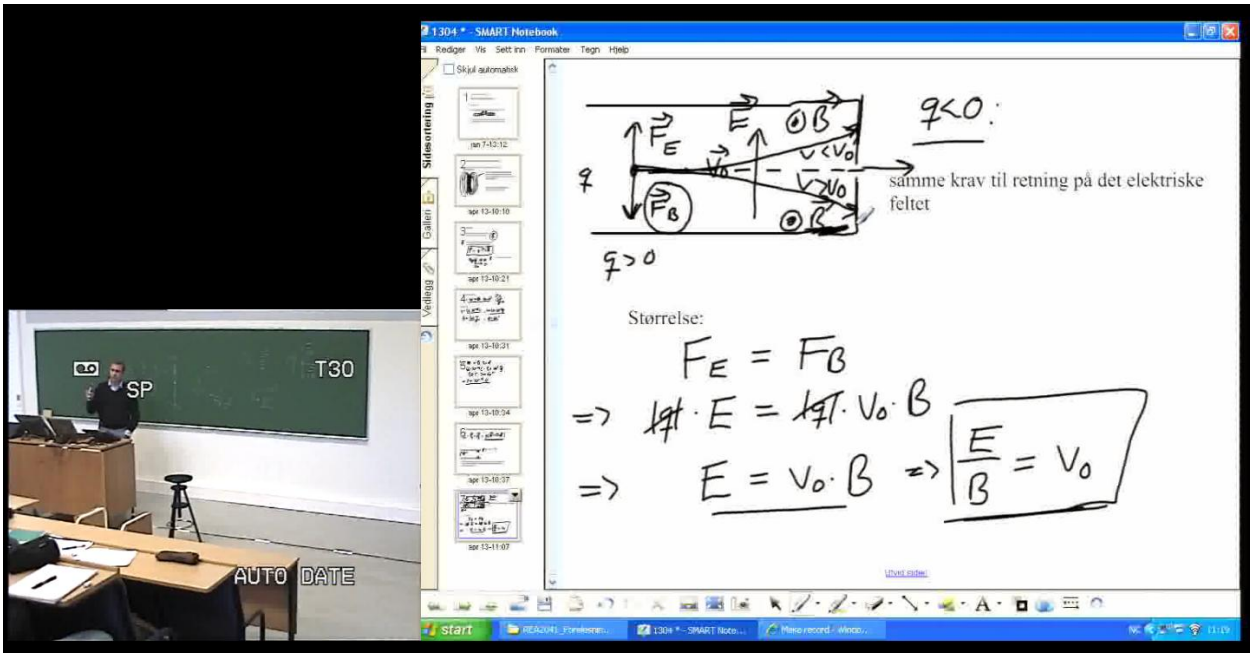
5.2.2 Bilde fra kamera

Under testopptak med Matterhorn fikk vi kun grønt bilde fra kameraet. Når vi så på kameraet direkte fikk vi riktig bilde. Vi tenkte først at det kunne være GStreamer som gjorde noe med videostrømmen. E-post ble sendt ut til Matterhornmiljøet, men svarene vi fikk hjalp lite. Det viste seg at andre også slet med grønt bilde, men av andre årsaker enn oss. Etter hvert forsto vi at det var oppløsningen på videokortet som var problemet. Når vi satt oppløsningen i pipelinen ble bildet som det skulle.

Vår manuelle løsning bruker en pipeline vi kan sette oppløsningen i. Ved automatisk opptak fra timeplanen fungerer dette foreløpig ikke. Fra og med versjon 1.1 skal det være mulig å bruke en "customProducer".[25] Det vil si at i stedet for å sette kilde som for eksempel "/dev/video1" kan man sette hele pipelinen som kilde. Vi antar at dette vil medføre at man også kan hente ut nettverksstrømmer som kilde, men har ikke fått dette bekreftet.



Figur 9. Slik presenterer Matterhorn 1.0 en forelesning gjort med vårt manuelle system



Figur 10. Slik presenteres ett opptak i fullskjerm

5.2.3 Bilde fra projektor

Ved opptak fra prosjektor via vga2usb-kilden har vi hatt en del problemer. Dersom oppløsningen er feil vil det automatiske opptaket ofte krasje. Vi har brukt 1024*768 siden det er dette de fleste prosjektorene bruker, og da har det gått bra.

På den manuelle delen fikk vi gjort opptak, men denne delen ble spilt av altfor fort. Etter litt regning forsto vi at matterhorn spilte av videoen med 25 bilder i sekundet, mens opptaket vårt var gjort med 5. Vi tar bare opp 5 bilder per sekund siden det er dette videokortet klarer.[26] Pipelinen ble endret så den dupliserer bildene og det blir 25 bilder per sekund (selv om det egentlig bare er 5). Etter dette spilles videofilen av i riktig hastighet.

5.2.4 Automatisk e-post

Automatisk utsendelse av e-post ble først utviklet på en Windows-maskin. Her var det bare å sette opp en smtp-server, så fungerte det uten problemer. Når vi gjorde det samme på Ubuntu-maskinen vår på skolen ble det ikke sendt noe e-post. Vi leste mye om at å sette smtp-server kun var for Windows, og at vi trengte å sette opp "sendmail" eller "postfix". Dette stemte ikke. Det holdt å bruke smtp.hig.no, men det måtte settes opp tilgang for oss hos IT-tjenesten. Etter at dette var i orden fungerte e-post-utsendelsen som den skulle.

5.2.5 Informasjon om filene

For at Matterhorn skal behandle opptaket vårt må det få med diverse informasjon om filene. Dette er størrelse, MD5 checksum og lengde på opptaket. De to første var greie å få tak i, men lengden på opptaket var vanskeligere. Etter at vi klarte å hente ut lengden ved å blant annet

bruke Avidemux, gjorde vi mange opptak. Det gikk stort sett bra, men en gang i blant feilet det. Enkelte ganger feilet det tre ganger på rad, andre ganger utførte vi over 40 opptak uten det skjedde igjen. Vi kom etter hvert fram til at det var på grunn av at vi ikke sjekket om tidsvariablene var mindre enn 10. Opptak på fem sekunder og 11 millisekunder fungerte, mens opptakt på fem sekunder og ni millisekunder fungerte ikke. Etter at vi la til at alle ledende nuller på tall under ti ble fjernet, har det virket som det skal.

5.2.6 Streaming

Som nevnt tidligere har vi vært innom ulike streamingløsninger. Den første vi kjørte ordentlige tester på var via en icecastserver. Dette gikk i utgangspunktet helt fint og streamen kunne hentes ut fra andre datamaskiner i nettverket. Et problem med denne løsningen var at vi ble bundet til ogg som filformat, da dette er det eneste videoformatet iceacast godtar.[27] Et annet var at skolens Darwin streamingserver forventer en RTSP-strøm inn, mens icecast kun produserer HTTP-stream.

Vi byttet til udp-streaming direkte fra GStreamer. Denne fikk vi hentet ut lokalt på samme datamaskin som den ble streamet fra, men ikke fra andre datamaskiner i nettverket. Løsningen ble å hente ut denne strømmen i VLC og streame den videre som RTSP herfra. Dette er gjort mange ganger tidligere og skal gå fint. Siste delen med å lage riktig sdp-fil har vi ikke rukket innenfor tiden vi har hatt til rådighet, men det er gjort før av andre, og vi er trygge på at dette skal fungere.

5.2.7 Test i auditorium

Vi har testet løsningen i to auditorier. Først i K109. Her er det videokonferanseutstyr fra før, så vi koblet oss på utgangen fra dette. Dermed fikk vi det samme på opptaket som det som ble vist på prosjektor 1 i rommet. Dette kunne være både forelesers bærbare maskin, datamaskinen som står i rommet eller dokumentviseren. Vi gjorde noen opptak av oss selv før vi tok ned boksen igjen for å jobbe mer på den.

K105 var det andre auditoriet vi tok opptak i. Her er det ingen konferanseløsning, så vi måtte velge å splitte videostrømmen enten fra datamaskinen som står der eller vga-inngangen som foreleser kan koble sin bærbare maskin til. Vi valgte det siste.

Vi spurte Are Strandlie om han kunne teste løsningen vår, noe han gjerne gjorde når han forsto at det ikke medførte noe ekstraarbeid. Han tok opp to trekvarters forelesninger som vi nå har liggende på Matterhornserveren. Bortsett fra at vi hadde stilt volumet på mikrofonen for lavt synes vi det ble to gode opptak. Vi fikk både bilde fra kameraet i salen og "prosjektorbilde" fra notatene hans.

På grunn av at vi har brukt et gammelt kamera med lav oppløsning var ikke kvaliteten på kameraopptaket spesielt bra, men det hadde vi regnet med på forhånd. Vi gjorde testen primært for å se hvordan løsningen vår virket, og for å se om vi fikk "klasseromsfølelsen".

5.2.8 Ressursbruk

I forbindelse med å skrive pipelines kjørte vi en del ressurstester. Både streaming og enkoding krever mye av systemet, og for å holde kostnadene nede vil det være begrenset hvor kraftig hardware det er på opptaksboksene.

Noen av de første løsningene våre krevde 95-100 % prosessorbruk på klienten. Dette vil ikke fungere i lengden, og det vil bli hakking i bildet, så vi bygde opp pipelinene helt fra bunnen av igjen. Pipelinen vi endte opp med å bruke ligger på cirka 36 % prosessorbruk pluss det VLC vil bruke på å videresende strømmen. Se mer i punkt 3.2.1.1 om oppsettet på vår testmaskin.

5.2.9 Stresstester

I den siste tiden av testperioden har vi kjørt gjentatte tester av manuelt og automatisk opptak om hverandre. Dette har gått bra så lenge opptaket før har rukket å avslutte. I timeplanen hender det at en time slutter et gitt tidspunkt og at neste begynner i samme øyeblikk. Etter disse testene la vi inn at scriptet som lager opptakstimeplanen legger til fem sekunder på starten og trekker fra fem sekunder på slutten. En time fra 12:00:00 til 13:00:00 vil dermed bli tatt opp fra 12:00:05 til 12:59:55. Da rekker opptaket å avsluttes riktig før neste starter, og vi slipper at det krasjer.

Med tanke på streaming av opptakene fra server har vi kun testet dette med tre maskiner på samme tid. Dette gikk fint, men siden vår sever har vært en vanlig datamaskin er dette av liten relevans for hvordan en endelig løsning på dette området vil fungere.

5.2.10 Framtidige tester

Vi vet lite om hvor lenge en må forvente å vente på at et opptak skal bli klart for nedlasting når det er flere opptaksbokser som sender opptak til serveren samtidig. Vår erfaring er at det tar cirka 1,3 ganger så lang tid som opptaket er. Om dette gjelder på en kraftigere server, og om det blir en kø ved mange opptak bør testes. Skalerbarheten med tanke på streaming av opptak og live-streaming bestemmes i stor grad av hvilken server-løsning som velges.

Versjon 1.1 av Matterhorn ble sluppet helt på slutten av vår oppgave. Dette kan testes med "customProducer" og IP-kamera som kilde.

6 Avslutning

6.1 Evaluering

Etter cirka tre måneder med svært mye jobb innen utvikling og implementering ser vi oss godt fornøyd med resultatet. Vi har laget mye mer funksjonalitet enn det vi først hadde tenkt, og enn det vi hadde trodd var mulig innenfor tidsrammene.

Når vi ser tilbake på arbeidet er det klart for oss at dersom vi skulle gjort det samme om igjen, hadde vi løst noen ting på andre måter. Dette skyldes at vi har vært nødt til å lære oss veldig mye om de forskjellige systemene og verktøyene vi har jobbet med underveis i prosessen. Allikevel er de endringene vi kunne tenkt oss å gjøre forholdsvis små, slik at essensen ville blitt den samme. Endringene blir beskrevet i punkt 6.2.1

Da vi begynte med implementeringen skulle vi egentlig bare utvikle den automatiske opptaksdelen. Etter vi hadde jobbet med denne i 2-3 dager spør oppdragsgiver om hva vi tror om å utvikle et system for å gjøre manuelle opptak, parallelt med den automatiske. På det tidspunktet følte vi at vi allerede hadde såpass god oversikt over den automatiske delen, at vi ble enige om å utvikle begge systemene.

Mot slutten av implementeringsperioden begynte vi også å diskutere implementering av live-streaming med oppdragsgiver. Vi fant ut at vi ikke hadde tid til dette, men selv om vi i forprosjektrapporten hadde skrevet at vi ikke skulle se på dette temaet, brukte vi et par dager på å undersøke mulighetene for å implementere dette i etterkant. Vi kom frem til at løsningen på dette var å skrive om pipelinen. Vi testet dette, og la inn mulighet for senere implementering av live-streaming.

Under utviklingen av den manuelle delen brukte vi lenger tid enn forventet blant annet på å velge kodeker og å få til en pipeline som tilfredsstilte krav først til videokvalitet, så til streaming også videre. Men denne stadige endringen av pipelinen førte til at vi fikk mer innsikt på området og dermed en pipeline som er mer effektiv og lettere å utvide i etterkant.

Vi føler at tross en del fordeler med helautomatisk opptak, er modulen med manuelt opptak, per nå, en bedre løsning. Dette skyldes delvis dataene som ligger i TimeEdit. Kommentarer angående dataene i TimeEdit og den helautomatiske løsningen kommer i punkt 6.2.3.1.

6.2 Evaluering av gruppearbeid

6.2.1 Innledning

Vi (gruppen) har jobbet sammen med et programmeringsprosjekt ved en tidligere anledning. Ettersom vi kjente hverandre fra før, har vi sluppet å bruke tid på å bli kjent med hverandre og hvordan den andre jobber. Dette har gjort det enklere å fordele arbeidsoppgaver og organisering av jobbingen generelt.

6.2.2 Organisering

Gruppen har jobbet sammen på skolen hver dag i prosjektperioden med unntak av to dager som er brukt til reising og to dager som vi har jobbet hjemme hver for oss. Vi har gjort det meste av jobbing på skolen, men de periodene det har vært størst arbeidsmengde har vi også jobbet utenfor “kontortiden”. Da har vi avtalt fra dag til dag hva vi skal jobbe med/bli ferdig med til neste dag.

Vi har jobbet med prosjektet 30 timer hver, i snitt hver uke. Det er for Marius sin del inkludert 3 timer forelesning hver uke, men uten timene vi har jobbet hjemme. Loggføringen ble gjort dag for dag med en kort beskrivelse av hva vi gjorde og hvor mange timer vi brukte. Tiden vi brukte og hva vi gjorde hjemme ble ikke spesifikt loggført, men gjerne tilføyd til hva vi gjorde dagen derpå. Vi arbeidet som regel fra cirka klokken 0900 til cirka 1500 og klokken 1000 til 1600.

Vi har ikke hatt faste møter med arbeidsgiver. Kontoret vårt ligger vegg i vegg med kontoret til arbeidsgiver, så vi har allikevel hatt jevnlig kommunikasjon, gjerne hver dag. I starten av prosjektet, under utformingen av forprosjektrapporten, hadde vi mye kontakt med veileder, mens under utviklingen fikk vi all hjelpen vi trengte hos arbeidsgiver. Under rapportskrivingen tok vi opp igjen kontakten med veileder, og han kommenterte rapporten ettersom den tok form i et Google Document. Dette fungerte bra.

Innad i gruppen ble vi enige om at vi ikke trengte faste møter, men at vi underveis gjorde opp status på oppgavene vi jobbet med der og da, i tillegg til prosjektets totale fremgang. Vi fulgte ikke utviklingsmodellen vi valgte til punkt og prikke, men brukte hovedmomentene til å styre prosjektet underveis.

For å holde styr på hva som skulle gjøres underveis hadde vi flere lister med arbeidsoppgaver. En liste med modulene som skulle utvikles og en liste per modul som beskrev hva som skulle gjøres med modulen i detalj. Detalj-listen hadde vi på en tavle på kontoret der vi kontinuerlig oppdaterte hva som var ferdig og skrev nye ting som dukket opp underveis. Denne listen bidro til at det hele tiden var klart hva som gjenstod på hver modul.

Generelt så har møter og organiseringen av arbeid vært forholdsvis uformelt siden gruppemedlemmene er såpass godt kjent fra før, og vi har sittet og jobbet så og si sammen med oppdragsgiver og andre medarbeidere på IT-avdelingen. Dette har vært en fordel med tanke på

at alle spørsmål som har dukket opp, både fra oss og arbeidsgiver, har blitt avklart med en gang.

6.2.3 Fordeling av arbeid

Arbeidet med de forskjellige modulene ble fordelt slik de ble av flere grunner. Når vi begynte på nye moduler og lignende, diskuterte vi hva vi ville jobbe med og fordelte ut fra ønsker. Ellers så fungerte det gjerne slik at den som var ferdig med en ting begynte på det som var naturlig å gjøre videre.

Selv om vi ofte jobbet med hver vår ting, diskuterte vi valg av løsningsmetoder der det var naturlig og jobbet litt sammen om det meste. Dette var en klar fordel under testing når det oppsto feil. Ettersom begge hadde jobbet med koden, hadde vi som regel to eller flere teorier om hva som kunne være årsaken til feilen.

6.2.4 Prosjekt som arbeidsform

Det har vært svært læringsrikt å jobbe med et prosjekt som er av denne størrelsen der noen venter på resultatet og har forventninger til det. Dette er det største prosjektet vi har jobbet med og vi føler det er en god forberedelse til enda større prosjekter som venter i arbeidslivet.

Ved å jobbe på denne måten har vi vært nødt til å ta styring over egen læring og holde styr på vår egen fremdrift. I tillegg har vi lært svært mye om fagfeltene vi har jobbet med siden vi har måtte oppsøke all informasjon selv der den er tilgjengelig. Dette fører til at man ikke bare finner den informasjonen man er ute etter, men man kommer over all mulig informasjon om

emnet som man deretter må sortere og filtrere.

I tillegg til det fag-tekniske har vi også lært en hel del om hvordan forskjellige organisasjoner jobber.

Vi har likt å jobbe med prosjekt som arbeidsform og skulle gjerne sett flere lignende oppgaver gjennom utdanningen vi har tatt.

6.3 Videre arbeid

6.3.1 Våre forslag til endringer av / tillegg til nåværende system

6.3.1.1 VGA - oppløsning

Slik systemet er nå krever det en oppløsning på 1024*768 på kilden som blir koblet til VGA-til-USB enheten. Her kan det være en mulighet å legge inn valg av forskjellige oppløsninger i brukergrensesnittet der opptak startes og stoppes, slik at den som skal gjøre opptak velger den oppløsningen som han/hun bruker. Eventuelt må det legges inn en sjekk på hvilken oppløsning som faktisk er i bruk, dette kan være teknisk utfordrende.

6.3.1.2 Valg av opptaksenheter

I de forskjellige rommene med opptaksutstyr kan det være koblet opp forskjellige typer og antall opptaksenheter. En foreleser kan ha ønske om å bruke bare noen av opptaksenhetene som finnes. Dette kan løses ved å integrere en liste over tilgjengelige opptaksenheter i brukergrensesnittet der brukeren kan velge hvilke enheter han/hun vil gjøre opptak fra.

Figur 11. Eksempel på kilder valgt av bruker

6.3.1.3 Signalsjekk ved opptak

Dersom det gjøres opptak fra en kilde som ikke har signal inn, kan pipeline'en som gjør opptak krasje. Dette løses delvis av punktet ovenfor, men dersom brukeren skulle ha valgt en enhet som ikke har signal inn, kan dette løses med en automatisk sjekk for å se om det er signal på alle valgte enheter rett før opptak startes.

6.3.1.4 Symlink

På grunn av at inngangene på videokortet kan bytte navn uten videre når opptaksboksen startes på nytt, må man lage en symlink som idenfiserer den samme inngangen på kortet selv om maskinen har startet på nytt. Dette er for å unngå å måtte bytte videokilde i opptaksprogramvaren hver gang man starter maskinen på nytt.

6.3.1.5 Lukke socket når daemon tvangsavsluttes

Når daemon tvangsavsluttes blir socketen værende åpen. Dette fører til at daemonen neste gang må startes to ganger for og virkelig starte. Dette kan løses med å fange opp avslutningssignal og avslutte socketen før daemon-prosessen dør.

6.3.1.6 Passord i klartekst

I zipsend.sh, skriptet som overfører nye opptak fra opptaksboks til server, ligger passordet for SSH-tilkoblingen i klartekst. Dette kan med fordel flyttes/krypteres.

6.3.1.7 Pipeline uten enkoding

For å få til å lagre og streame dataene som kommer fra opptakskildene måtte vi kjøre dataene gjennom en enkoder. Dette fører til datatap og høyere CPU-bruk. Ved å konstruere en pipeline som kan streame og lagre til fil, uten enkoding, vil man ha større valgmuligheter når det kommer til behandling av datastrømmen som skal live-streames og man vil bruke mindre CPU på opptaksboksen.

6.3.1.8 Sjekk om det allerede blir gjort opptak

Dette gjelder kun dersom det automatisk og det manuelle systemet blir brukt om hverandre.

Når opptak skal startes burde det være en sjekk på om opptak allerede er i gang. Hvis opptak er i gang allerede vil opptaksprosessen som prøver å starte nytt opptak krasje.

6.3.2 Naturlige utvidelser av vårt arbeid

Etter å ha jobbet med utvikling av denne programvaren og hatt samtaler med oppdragsgiver, veileder og andre interessenter, har vi fått endel ideer om hva som vil lønne seg å jobbe videre med.

6.3.2.1 Autentisering/Innlogging for å starte manuelle opptak

Brukergrensesnittet slik det er nå, er web-basert og åpent for alle på lokalnettverket på HIG. Dette blir gjort tilgjengelig via en lokal webserver på hver enkelt opptaksboks. Det vil si at hvem som helst, fra hvor som helst kan starte opptak i alle rom der det er en slik opptaksboks.

For å sikre at opptak kun er tilgjengelig for en viss gruppe burde det legges inn en autentisering før man kan gjøre opptak. Dette kan for eksempel løses ved å sentralisere brukergrensesnittet på en server og kreve autentisering for å få tilgang dit. I tillegg må man sikre at opptaksboksen kun kan styres fra disse sentraliserte brukergrensesnittene. Innlogging kan for eksempel skje via FEIDE.

6.3.2.2 Autentisering/Innlogging for å se forelesninger

Det må gjøres en forholdsvis stor jobb/avklaring rundt personvern og opphavsrett i forhold til publisering av forelesningsopptak. Vi ser for oss at i hvert fall en del av opptakene må ligge beskyttet, slik at bare en viss gruppe kan logge seg inn og se de. Dette kan være på grunn av opphavsrett til opptakene, opphavsrett på materiale som blir brukt i opptakene eller av personvernhensyn.

6.3.2.3 Filtrering av hva som skal gjøres opptak av (Automatisk løsning)

Det kan være emner som det av forskjellige grunner ikke skal gjøres opptak av. Dette kan for eksempel skyldes at foreleser ikke ønsker å bli filmet/publisert, eller at forelesningens innhold ikke egner seg for publisering.

Dette kan integreres på forskjellige måter, men vi ser for oss to løsninger som er enkle å integrere og bruke:

Konfigurasjonsfil med navn på emner som skal tas opp:
Det er mulig å lage en konfigurasjonsfil på serveren med en liste over emner som skal tas opptak av. Da sjekker systemet mot denne listen hver gang den skal generere opptakstimeplan. Dersom emnet ikke står i denne listen, blir det ikke generert opptaksinformasjon om dette tidspunktet i opptakstimeplanen. Da er det også mulig å ta bort emnet fra listen for en dag, eller en uke dersom det skjer noe spesielt akkurat i en forelesning eller lignende. Denne løsningen krever noe endring i TimeEdit, se punkt 6.3.3.1.

Informasjon om noe skal tas opp direkte i TimeEdit:
Det er også mulig å legge inn informasjon i TimeEdit om noe skal tas opp eller ikke. Når det da genereres opptakstimeplan fra TimeEdit, sjekker systemet rett og slett om timen skal gjøres opptak av eller ikke i TimeEdit dataene. Dette krever også som forklart endring i TimeEdit.

6.3.3 Hva trengs for å ta et komplett system for opptak og streaming av forelesninger i bruk?

6.3.3.1 TimeEdit

Under utviklingen av vår automatiske løsning opplevde vi stadig problemer med data fra TimeEdit. Hvis man ønsker å ta i bruk en automatisk opptaksløsning som henter

opptaksdataene sine fra TimeEdit, må informasjonen som ligger der standardiseres.

Beskrivelse av faget:
Slik det fungerer i dag består beskrivelsen som ligger i TimeEdit av deler av navnet til faget. Dette er ikke standardisert og kan endre seg for eksempel hvis en annen person tar over administreringen av TimeEdit.

Dersom det var standard at denne beskrivelsen skulle være **emnekoden** til faget ville det være mulig å utnytte flere muligheter angående filtrering av hva som skal tas opp og ikke. Det ville også gi goder på andre områder, for eksempel vil det være lettere å finne frem til emnebeskrivelsen på HIG sine hjemmesider.

Forelesers identitet:
Foreleser navn blir per dags dato lagt inn i TimeEdit på en proprietær form.

Eksempel:

Mitt navn er Marius Slåtsveen. Hvis jeg holder av et rom, ville jeg blitt lagt inn i TimeEdit som "MaSl". Altså brukes de to første bokstavene fra begge navn. Hvis man har et mellomnavn, ville "a"-en i "MaSl" blitt byttet ut med forbokstaven i mellomnavnet. Det samme gjelder ved to etternavn.

Siden alle ansatte ved HIG har et offisielt HIG-brukernavn hadde det vært en fordel om dette ble brukt i TimeEdit. Dette brukernavnet ville da være kompatibelt med alle skolens systemer. Dette gir da mulighet for å slå opp all informasjon om foreleserne i en database med informasjon over skolens ansatte og studenter (LDAP).

Dersom personen som har reservert rommet ikke har et HIG-brukernavn, burde det da brukes en standard for dette også.

Det er mulig å legge inn informasjon i TimeEdit som sier noe om en forelesning skal tas opp. Dette må i så fall også standardiseres.

Et opptakssystem vil være avhengig av å bruke informasjon fra en iCal-fil hentet fra TimeEdit. Her presenteres all informasjonen nevnt ovenfor i ett datafelt. Nærmere bestemt "summary"-feltet. Dette gjør det svært viktig at alle feltene er fylt ut og at de kommer i samme rekkefølge hver gang. Rekkefølgen og hvilke felter som alltid må være med må også standardiseres. Dersom rekkefølgen er feil, eller at det mangler felter vil ikke systemet fungere slik det er tenkt.

Kommentar angående rom/location i TimeEdit:
Ofte står det flere rom listet opp i "location"-feltet. Dette kan føre til at det blir tatt opptak i et rom der det ikke er forelesning.

6.3.3.2 Formater

I forbindelse med formater er det to viktige problemstillinger. Hva slags format materialet skal lagres i, og hvordan det skal distribueres. I Høgskolen i Gjøviks Strategiske plan for 2009 - 2012 står det at høgskolen skal "utnytte og bruke teknologi basert på åpne standarder"[28].

Når det gjelder lagring, kan video for eksempel være kodet med Theora og pakket som Ogg. Dette er løsningen Standardiseringsrådet har foreslått skal brukes på offentlige nettsider i Norge[29]. I tillegg til dette er også H.264 enkoding pakket i MP4 foreslått av den samme gruppen. Formater for lagring skaper ikke problemer siden man kan etterkomme krav satt i den strategiske planen til HIG samtidig som man bruker formater støttet av Matterhorn[30].

Til distribuering er det hovedsakelig tre alternativer. HTML5, Silverlight og Flash. Silverlight er ikke en åpen standard[31] og det støttes i veldig liten grad av mobile enheter[32]. HTML5 har stort potensiale, men også her er det flere utfordringer. Per i dag er H.264 det mest utbredte

formatet i forbindelse med HTML5. Problemet med dette er at H.264 er patentert[33]. Dermed vil høgskolen kunne ende opp med å måtte betale for alle forelesninger som tas opp, og det er lite sannsynlig at for eksempel Mozilla Firefox vil støtte det på grunn av lisensbetingelsene[34]. Google's Chrome og Opera vil heller ikke støtte det i de nærmeste fremtidige versjonene[35]. Et alternativ innen HTML5 er WebM/VP8. Dette er et åpent format som kan sammenlignes med H.264[36], men det er foreløpig liten støtte for akselerasjon av WebM i maskinvare[37]. Men derimot vil det være god støtte i de store nettleserne Firefox, Chrome, Opera, Android , Internet Explorer (med plugin) og Safari (med plugin til QuickTime) i tiden fremover[35].

Matterhorn bruker i dag Flash. Dette støttes på de tre store plattformene Windows, Mac og Linux, i tillegg til Android (versjon 2.3)[38].

For å gjøre opptakene tilgjengelig for flest mulig ser vi for oss at den beste løsningen i skrivende stund er å bruke Flash og i tillegg HTML5 med H.264 for iOS. Dette betyr sannsynligvis at filene må lagres i to forskjellige formater. Et åpent format for å tilfredsstille kravene i skolens strategiske plan, og H.264 for å nå ut til iOS-brukerne.

6.3.3.3 Bilde og lyd

Under vår testperiode har vi brukt et gammelt JVC håndholdt kamera og en enkel mikrofon for å få inntrykk av hvordan opptakene blir. Dersom et slikt system skal tas i bruk, må det naturligvis brukes bedre utstyr.

Vi har vært i kontakt med Rolf Ruediger ved universitetet i Osnabrück i Tyskland og fått tips om hva slags kameraer som brukes der. De bruker forskjellige Sony kameraer, der EVI D70 er det nyeste. Det samme kameraet bruker de også ved Universitetet i Wien. Andre kameraer som brukes i Matterhornmiljøet er Panasonic BB-HCM581A og Logitech C910 HD webcam.

For å få best mulig opptak av tavla kan det avtales at bare en del av tavla brukes. Kameraet står

da innstilt på å ta opp denne delen og foreleseren.

Vi ser for oss at en god løsning vil være IP-kameraer, da videostrømmen fra disse kan hentes opp på opptaksboksen samtidig som de er tilgjengelige på web. Et ønske fra oppdragsgiver, veileder og andre interessenter vi har snakket med er at en skal kunne gå inn på for eksempel K105.hig.no og få frem forelesningen som er aktiv i K105, dersom det er undervisning i rommet.

Når det gjelder lyd er det flere aspekter å tenke på. Er det ønskelig å ha mikrofoner i salen så man får med interaksjonen mellom foreleser og student? Eller vil dette generere unødvendig mye bakgrunnsstøy? Da må det i så fall være en mikser før lyden går inn til opptak.

6.3.3.4 Autentisering/innlogging

Det vil være nødvendig med autentisering både for å gjøre opptak og for å kunne se opptak. Dette er beskrevet i punkt 6.2.2.2.

6.3.4 Matterhorn i fremtiden

Matterhorn er fortsatt veldig nytt. Under vår oppgave har vi benyttet versjon 1.0 som har en del mangler.

I slutten av april ble versjon 1.1 sluppet, og denne versjonen skal være en markant forbedring. I tillegg til mulighetene i 1.0, skal det blant annet være mulig å klippe et opptak i ettertid og å bruke “customProducer” for å tilpasse forskjellig opptaksutstyr til Matterhorn. Selve designet skal også ha blitt mer brukervennlig.

Det tok omtrent syv måneder før 1.1 ble sluppet etter at 1.0 var ute. I mellomtiden har det kommet en liten oppgradering, 1.0.1 med noen små endringer. Det er flere store universitet involvert i utviklingen[39], og vi mener det har potensiale til å bli et meget godt produkt.

Sett ut i fra kostnad og muligheter kontra kommersielle alternativer bør skolen absolutt vurdere om de bør satse på å bruke dette produktet. Siden det utvikles av utdanningsinstitusjoner som skal bruke det selv vet de mye om hva slags funksjonalitet de ønsker. Vi vet ikke når de neste store oppgraderingene vil komme, men det er ukentlige møter i Matterhornmiljøet, og versjon 2.0 er allerede planlagt[40].

Under toårsplanen til Matterhornprosjektet fra desember 2009 er mobilapplikasjon og mobil-klient-support nevnt under "Strongly Consider"[41], noe som skulle tilsi at mobiltelefoner også vil støttes i nærmeste framtid.

6.4 Konklusjon

Vi har utviklet et system for opptak av forelesninger som skal gjøre det enklere for forelesere og HIG som institusjon å legge til rette for fleksible studietilbud.

Systemet tilbyr automatiske opptak via den allerede eksisterende timeplanen i TimeEdit og i tillegg manuelle opptak.

I løpet av prosjektet har vi lært mye om kommunikasjon mellom forskjellige datateknologier og hvordan man løser dette i praksis. Vi har også lært mye om hvordan det er å jobbe sammen med andre om et større prosjekt og hvor viktig planlegging av forskjellige aktiviteter er da. Det har vært lærerikt, spennende og motiverende å jobbe sammen om en slik oppgave.

Etter hvert som vi har jobbet utviklingen av systemet, har vi kommet frem til en del områder som det må jobbes videre med før et slikt system skal settes i produksjon og tas i bruk. Dette er blant annet:

- Personvern
- Autentisering for å gjøre opptak
- Autentisering for å se opptak

Disse punktene og resten av områdene det må arbeides med er nærmere beskrevet i punkt 6.2.

7 Litteraturliste

1. Høgskolen i Gjøvik. Ingeniørfag - fleksibel[Internetside] Gjøvik: Høgskolen i Gjøvik [sitert 12.05.11] Tilgjengelig fra: http://www.hig.no/studietilbud/teknologi/bachelor/fleksibel

2. Høgskolen i Gjøvik. Bachelor i økonomi og ledelse, fleksibel[Internetside] Gjøvik: Høgskolen i Gjøvik [sitert 12.05.11] Tilgjengelig fra: http://www.hig.no/studiehaandbok/studiehaandboeker/2011_2012/studiehaandbok_2011_2012/toel/bachelor_i_oekonomi_og_ledelse_fleksibel

3. Høgskolen i Gjøvik. Informasjonssikkerhet [Internetside] Gjøvik: Høgskolen i Gjøvik [sitert 18.05.11] Tilgjengelig fra: http://hig.no/studietilbud/it/master/mis

4. Høgskolen i Gjøvik. Medieteknikk [Internetside] Gjøvik: Høgskolen i Gjøvik [sitert 18.05.11] Tilgjengelig fra: http://hig.no/studietilbud/it/master/mmt

5. Høgskolen i Gjøvik. Bachelor i sykepleie [Internetside] Gjøvik: Høgskolen i Gjøvik [sitert 18.05.11] Tilgjengelig fra: http://hig.no/studiehaandbok/studiehaandboeker/2011_2012/studiehaandbok_2011_2012/hos/bachelor_i_sykepleie

6. Uninett. Uninetts produkter og tjenester[Internetside]. Trondheim: Uninett; [sitert 12.05.11] Tilhgjengelig fra: http://www.uninett.no/leveranser

7. Uninett. eCampus 2011-2015[Internetside]. Trondheim: Uninett; [oppdatert 16.02.10, sitert 12.05.11] Tilhgjengelig fra: http://www.uninett.no/sites/drupal.uninett.no.uninett/files/webfm/eCampus/eCampus%202011-2015.pdf

8. Uninett. eCampus arbeidsplan 2010[Internetside]. Trondheim: Uninett; [oppdatert 07.02.10, sitert 12.05.11] Tilhgjengelig fra: http://www.uninett.no/sites/drupal.uninett.no.uninett/files/webfm/eCampus/eCampus%20arbeidsplan%202010.pdf

9. Matterhorn. Directory of Opencast Organizations [Internetside]. Matterhorn; [sitert 12.05.11] Tilgjengelig fra: http://www.opencastproject.org/institutions

10. Matterhorn. Matterhorn Features & Functionality[Internetside]. Matterhorn; [sitert 12.05.11] Tilgjengelig fra: http://www.opencastproject.org/matterhorn_features

11. Uninett. Kunnskap trumfer alt [Internetside]. Trondheim: Uninett; [sitert 12.05.11]. Tilgjengelig fra: http://www.uninett.no/kunnskap-trumfer-alt

12. Høgskolen i Gjøvik. Ledningsnett [Internetside]. Gjøvik: Høgskolen i Gjøvik; [sitert 12.05.11]. Tilgjengelig fra: http://www.hig.no/studiehaandbok/studiehaandboeker/2010_2011/emner/avdeling_for_teknologi_oekonomi_og_ledelse/byg1311f_ledningsnett

13. Uninett. Løft for IKT-støtte til forskning, undervisning og formidling [Internetside]. Trondheim: Uninett; [sitert 12.05.11]. Tilgjengelig fra: http://www.uninett.no/ecampusmillioner

14. Priser opplyst av IT-avdelingen for kommersielle systemer satt opp mot et system av vår art.

15. Høgskolen i Gjøvik. Rapport og planer 2010 – 2011 [Internetside] Gjøvik: Høgskolen i Gjøvik [sitert 18.05.11] Tilgjengelig fra: http://www.hig.no/content/download/28372/325561/version/1/file/Rapport+og+planer

16. Sommerville I, Agile software development. I:Horton, M, redaktør. Software Engineering. 9. Boston, Massachusetts:Pearson Education Inc;2010.s.64-77

17. Sommerville I, Software processes. I:Horton, M, redaktør. Software Engineering. 9. Boston, Massachusetts: Pearson Education Inc;2010.s.30-32

18. Sommerville I, Software processes. I:Hornton, M, redaktør. Software Engineering. 9. Boston, Massachusetts: Pearson Education Inc;2010.s.32-35

19. Matterhorn. Capture Agent Installation V1.0 [Internetside]. Matterhorn; [sitert 12.05.11] Tilgjengelig fra: http://opencast.jira.com/wiki/display/MH/Capture+Agent+Installation+v1.0

20. Matterhorn. Scaling [Internetside]. Matterhorn; [sitert 12.05.11] Tilgjengelig fra: http://opencast.jira.com/wiki/pages/viewpage.action?pageId=18874477

21. Sommerville I, Distributed software engineering. I:Horton, M, redaktør. Software Engineering. 9. Boston, Massachusetts: Pearson Education Inc;2010.s.492

22. Matterhorn. Capture Agent Installation V1.0 [Internetside]. Matterhorn; [sitert 15.05.11] Tilgjengelig fra: http://opencast.jira.com/wiki/display/MH/Capture+Agent+Installation+v1.0

23. http-stats.com. Common Web Server software comparison report [Internetside]. http-stats.com; [sitert 24.05.11] Tilgjengelig fra: http://www.http-stats.com/

24. Mean. What is avidemux [Internetside]. Mean; [sitert 15.05.11] Tilgjengelig fra: http://avidemux.sourceforge.net/

25. Matterhorn. Capture Agent Configurration V1.1 [Internetside]. Matterhorn; [sitert 12.05.11] Tilgjengelig fra: http://opencast.jira.com/wiki/display/MH/Capture+Agent+Configuration+V1.1

26. Epiphan Systems Inc. Frame grabbers Overview [Internetside]. Epiphan Systems Inc; [sitert 12.05.11] Tilgjengelig fra: http://www.epiphan.com/products/frame-grabbers/

27. Stream24.com. Frame grabbers Overview [Internetside]. Stream24.com; [sitert 12.05.11] Tilgjengelig fra: http://www.stream24.com/?page=icecast-server

28. Høgskolen i Gjøviks Strategiske plan for 2009 – 2012

29. Standardiseringsrådet. Standardiseringsrådets forslag til 2. versjon av Referansekatalog for IT-standarder i offentlig sektor [Internetside]. Regjeringen [sitert 19.05.11] Tilgjengelig fra: http://www.regjeringen.no/upload/FAD/Vedlegg/IKT-politikk/Standardiseringsradet/Forslag_Referansekatalog_IT-standarder_v2.pdf

30. Matterhorn. Create a Custom Encoding Profile V 1.0 [Internetside]. Matterhorn; [sitert 19.05.11] Tilgjengelig fra: http://opencast.jira.com/wiki/display/MH/Create+a+Custom+Encoding+Profile+v1.0

31. David Meyer. Frame grabbers Overview [Internetside]. ZDnet.com; [sitert 12.05.11] Tilgjengelig fra: http://www.zdnet.com/news/mozilla-warns-of-flash-and-silverlight-agenda/199508

32. Wikipedia [Internetside]. Wikipedia; [sitert 12.05.11] Tilgjengelig fra: http://en.wikipedia.org/wiki/Microsoft_Silverlight

33. Nilay Patel [Internetside]. Engadget; [oppdatert 04.05.10 sitert 12.05.11] Tilgjengelig fra: http://www.engadget.com/2010/05/04/know-your-rights-h-264-patent-licensing-and-you/

34. Harald Brombach [Internetside]. Digi.no; [oppdatert 16.12.10 sitert 12.05.11] Tilgjengelig fra: http://www.digi.no/858468/microsoft-gir-firefox-h264-stotte

35. Mark Pilgrim [Internetside]. diveintohtml5.org; [sitert 20.05.11] Tilgjengelig fra: http://diveintohtml5.org/video.html

36. Jan Ozer [Internetside]. Streamingmedia.com; [sitert 12.05.11] Tilgjengelig fra: http://www.streamingmedia.com/conferences/west2010/presentations/SMWest-2010-H264-VP8.pdf

37. WebM Project Group [Internetside]. WebM Project Group; [sitert 12.05.11] Tilgjengelig fra: http://www.webmproject.org/hardware/

38. Adobe Systems Incorporated [Internetside]. Adobe Systems Incorporated; [sitert 15.05.11] Tilgjengelig fra: http://www.adobe.com/software/flash/about/
39. Matterhorn. Matterhorn Partners [Internetside]. Matterhorn; [sitert 15.05.11] Tilgjengelig fra: http://www.opencastproject.org/matterhorn_partners

40. Matterhorn. Vision for Opencast Matterhorn 2.0 [Internetside]. Matterhorn; [sitert 15.05.11] Tilgjengelig fra: http://www.opencastproject.org/vision_opencast_matterhorn_20

41. Matterhorn. Considered Year 2 Requirements [Internetside]. Matterhorn; [sitert 15.05.11] Tilgjengelig fra: http://opencast.jira.com/wiki/display/MH/Considered+Year+2+Requirements

7.1 Figurliste

Figur 1. Systemets omgivelser 11
Figur 2. UseCase - diagram 12
Figur 3. Tykk klient - tolags arkitektur 30
Figur 4. Opptaksboks og Matterhorn Server ved manuelle opptak 30
Figur 5. Capture Agent og Matterhorn Server ved automatisk opptak 31
Figur 6 Dataflyt i det automatiske opptakssystemet 32
Figur 7. Web-brukergrensesnitt for testing av det manuelle systemet 34
Figur 8. E-post sending skjer etter Matterhorn Server har oppdaget et nytt opptak er klart 35
Figur 9. Slik presenterer Matterhorn 1.0 en forelesning gjort med vårt manuelle system 49
Figur 10. Slik presenteres ett opptak i fullskjerm 49

8 Vedlegg

Vedlegg A – Ord og uttrykk

Vedlegg B – Gantt-skjema

Vedlegg C – Logg

Vedlegg D – Oversiktstegning over systemet, lagd for blogg

Vedlegg E – Forprosjekt uten vedlegg

Vedlegg F – Statusrapporter

Vedlegg G – Prosjektavtale

Vedlegg A – Ord og uttrykk

Bash – Bourne-again shell – Et unix shell som gjør at man kan skrive kommandoer og starte andre programmer via kommandolinjen.

Capture Agent – datamaskin med programvare for opptak av forelesninger, her brukt i sammenheng med den automatiske opptaksløsningen. Også brukt: Matterhorn Capture Agent

customProducer – En løsning i Matterhorn for å definere egne pipelines på en opptakskilde.

FEIDE – Felles Elektronisk IDEntitet – Felles løsning for innlogging/autentisering for høyere utdanning i Norge.

fork – En kommando i C som starter en kopi av programmet fra den kodelinjen det startes på.

Icecast – Icecast Streaming Server er et serverprogram for streaming av lyd og bilde. Opprinnelig programvare for nettradio, men støtter nå også video.

LDAP – Lightweight Directory Access Protocol – En protokoll for uthenting av informasjon fra en database. Her brukt om Uninetts LDAP-server med informasjon om skolens ansatte og elever.

LMS - Learning Management System - Fronter, itslearning og lignende.

MD5checksum – 128 bit MD5 hash for å verifisere at det er riktig fil som er overført, og at filen ikke skadet eller endret.

Metadata – Tilleggsinformasjon til hoveddataene. Her er XML-filene med informasjon om videofilene metadatafiler.

Opptaksboks – datamaskin med programvare for opptak av forelesninger, her brukt I forbindelse med manuelle opptak

Pipeline – En serie med prosesser som gir input til hverandre. Det den første prosessen returnerer blir sendt som input til neste.

Postfix – Et e-post-program for unix. Et alternativ til Sendmail.

RTSP – Real Time Streaming Protocol – En nettverksprotokoll ment for å opprette og styre mediaoverføringer.

Script – En samling av kommandoer som utføres etter hverandre. Et slags lite program, men koden blir ikke kompilert til maskinkode før det kjøres.

Sendmail – Et e-postprogram for unix som også inneholder sin egen smtp-server.

Smtp-server – Simple Mail Transfer Protocol - En server for utgående e-post.

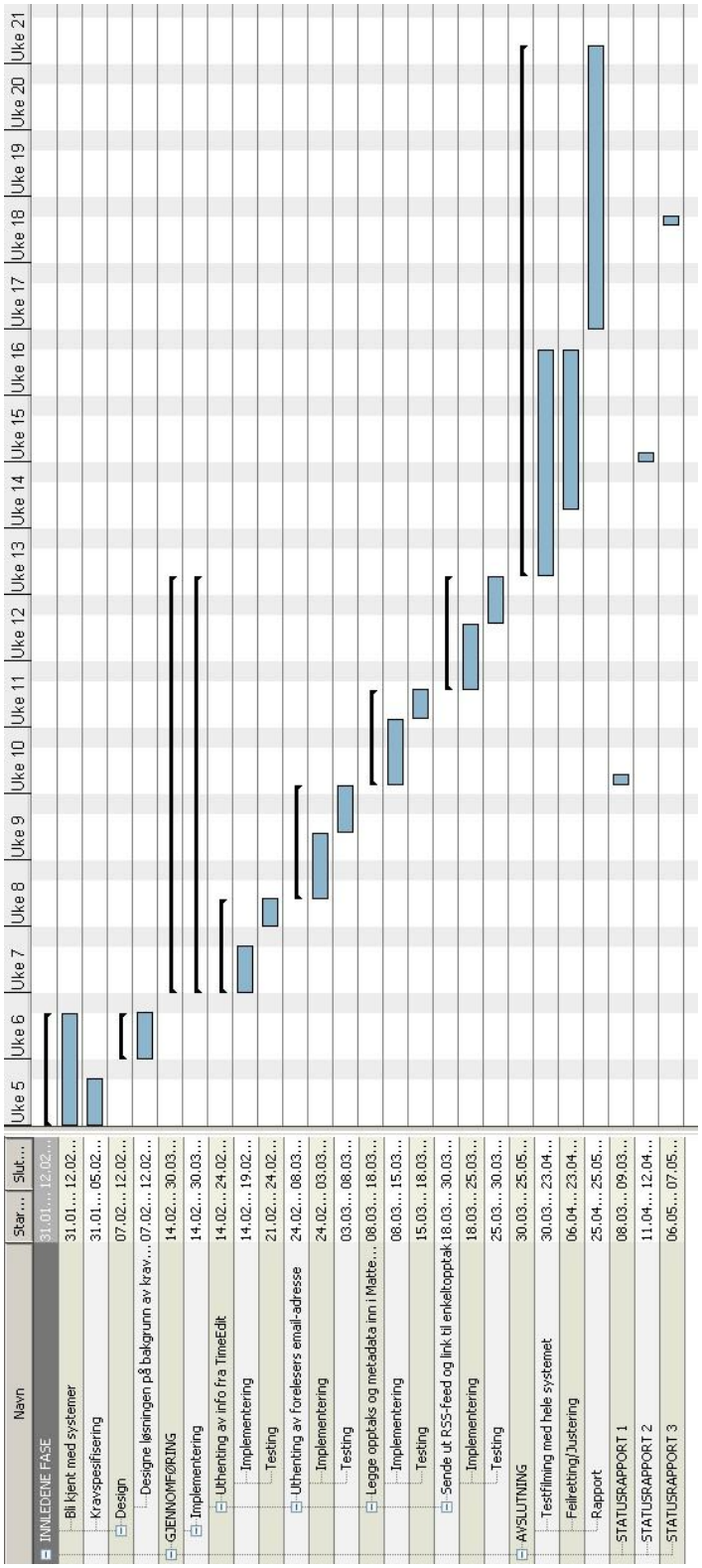
TimeEdit – Skolens timeplansystem

udp – User Datagram Protocol – En protokoll for sending av data. Det er ingen sjekk på om alle pakkene kommer fram, og de kan komme fram i ulik rekkefølge.

URL – Uniform Resource Locator – I denne sammenhangen nettadresser

VGA2USB – En boks fra Epiphan som tar i mot video via VGA og sender det til datamaskinen via USB.

Vedlegg B – Ganttskjema



Vedlegg C – Logg

Dato	Logg
12-27.05.11	Rapportskriving og finpussing
11.05.11	• Rapportskriving • 09:00-15:00
10.05.11	• Rapportskriving • 09:00-15:00
09.05.11	• Rapportskriving • 09:00-15:00
06.05.11	• Rapportskriving • GStreamer guide • 09:00-15:00
05.05.11	• Rapportskriving • Skrevet statusrapport • 09:00-16:00
04.05.11	• Rapportskriving • Møte med Kjell Are • 09:00-16:00
03.05.11	• Rapportskriving • Møte med Stian og Nina Tvenge • 09:00-16:00
02.05.11	• Rapportskriving • 09:00-14:00

29.04.11	• Rapportskriving • 09:00-15:00
28.04.11	• Rapportskriving • 09:00-14:00
27.04.11	• Rapportskriving • 09:00-15:00
26.04.11	• Rapportskriving • 10:00-15:00
15.04.11	• Jobbet med streaming mot matata.hig.no hele dagen. Vi sender data, men noe er feil... • 10:00-15:00
14.04.11	• ENDELIG fått e-post til å virke på ubuntumaskina. • Automatisk e-post når video er ferdig fungerer • Kommentert kode • 09:00-14:00
13.04.11	• Kjørt opptak i K105. • Kommentert kode • 09:00-15:00
12.04.11	• Satte opp kamera i K105 • Kjørte flere testopptak med timeedit og gui. • En av endringene våre fører til at lokalt opptak ikke fungerer etter timeedit-opptak. Dette må etterforskes. • 09:00-15:00
11.04.11	• Snakka med Are Strandli som gikk med på å teste til onsdag. • Tatt en kjapp test i k105, det ser ut til å fungere bra. • Skrevet statusrapport • 10:00-14:00

08.04.11	• Kobla opp utstyret i K105 • 11:00-14:00
07.04.11	• x-session • mediedagen • statestikkprøve
06.04.11	• Forsøkt å få PHP til å sende e-post. Det virker på windows, men ikke i linux. • E-postadresse hentes og skrives ut når det opprettes ny mappe • 09:00-16:30
05.04.11	• Endra litt i koden, så presenter blir e-postaddr • Fjerna red5 fra mattehorn. Nå fungerer streaming av videoer til ekstern maskin. • 09:00-15:00
04.04.11	• Endra GStreamer pipelinene til å streame udp til VLC. VLC skal så ta streamen videre til skolens streamingserver. Vi skal ikke komprimere bildet, siden båndbredde ikke er noe problem. • 09:00-15:00
01.04.11	• Satt sammen alt til en pipeline. Testa forskjellige utgaver og sammenligna ressursbruk. • 09:00-14:00
31.03.11	• 2 store gjennombrudd: - Ny pipeline på vga så vga-filmen ikke går i 5000x hastighet, men vanlig. - Fått til pipe med alle inkl streaming. Kamera og lyd i samme, vga i en egen. • 09:00-17:30
30.03.11	• Bestemt oss for å gå for streaming også. • Satt opp icecast-server og test-streama bare ett bilde • 10:00-14:00
29.03.11	• Satt opp capture-agenten bak k109. Vi har bilde fra prosjektor 1. • Undersøkt muligheter for streaming. Ikke funnet noen løsning som passer, vil se videre.

	• 10:00-17:00
28.03.11	• Kjørt tester på lokalt og timeplanlagte opptak om hverandre • 10:00 - 14:30
25.03.11	• Laget flytskjema • Laget how-to • Funnet navn på produktet • 09:00 - 15:00
24.03.11	• Skrevet installasjonsguide • Fiksa feil med ø i schedule • 12:00-16:00
23.03.11	• Samordna config-filene • Endra en del kode på daemon • 09:00 - 15:00
22.03.11	• Fiksa problem med flere rom • Jobba med problemet med at et opptak ikke kan starte samtidig som et annet slutter. • • 09:00 - 16:00
21.03.11	• Lyd rett fra dev • Opptak starter til riktig tid • “ “ er replaca med “_” • Sjekker at det er event i generateCalendarCode • 09:00-15:00
18.03.11	• Opptak fra audio • 09:00 - 16:30
17.03.11	• fikk vgga2usb til å fungere igjen • Lese inn audio/video og presenter/presentation • 09:00-15:00

16.03.11	• fjerna leading zero i manifesto.sh • fjerna statisk i opptak.sh • 09:00 - 15:00
15.03.11	• Det går nå med 2 kilder. Det vises i matterhorn. • 09:00-15:00
14.03.11	• I dag har vi satt opp ssh-server og bruker scp for å automatisk få over opptaka fra agenten til serveren. • 10:00-16:00
11.03.11	• Testa og retta opp en del feil i c-delen. • 09:00-15:00
10.03.11	• Satt sammen de ulike delene. • Samordna parameterne som sendes med. • 09:00-15:00
09.03.11	• Koda på xml-generatorene og socketserveren • 09:00-15:00
08.03.11	• Koda på xml-generatorene og socketserveren • 09:00-15:00
07.03.11	• Hatt statusmøte med oss selv, satser på å starte å teste delene sammen i løpet av uka. • Skrevet statusrapport • 09:00-14:00
04.03.11	•
03.03.11	• Fortsatt med koding på daemon og JavaScript. • 09:00-15:00
02.03.11	• Fått mellomvare.c til å sende med argumenter når det starter opptak.sh. • Koda på socket i PHP

	• GUI og JavaScript er ferdig for testing
01.03.11	• Koda på scriptet som stopper opptaket og lager metadatafilene og zipper dem. • Koda videre på javascritpet i GUI
28.02.11	• Koda videre hver for oss
25.02.11	• Testa med tcp-klient og server. Får starta script, men ikke sendt med variable riktig.
24.02.11	•
23.02.11	•
22.02.11	• Problemet lå i scheduler-fila der v brukte /n space, mens matterhorn krever /r/n space. Nå fungerer det å bruke vår egen feed. Opptakene vises i matterhorn. • 09:00 - 1600
21.02.11	• Forsøkt å forstå hvorfor schedule.ics ikke oppdaterer seg når vi bruker egenproduserte base64 attachments • Funnet ut hvilke endringer den godtar • Koda om ics-generatoren så det blir en fil per rom • 09:00 - 15:00
18.02.11	• testet feeden, det virker noen ganger, men andre ikke • ser ingen sammenheng i hvorfor det virker/ikke virker • 1030 - 1400
17.02.11	• Fått hjelp avJon til å lage vår egen kalenderfeed • 14:00 - 16:00
16.02.11	• Testa med .ics-filer som feed • 09:00 - 13:00

15.02.11	• Diskutert mer design av kalenderløsning • Bestemt at det skal være en ics-fil for hvert rom • De skal ha samme navn som rommet • 09:00-15:00
14.02.11	• Diskutert hvordan vi skal få ical-filene til å samarbeide med opptaks-agenten. • Satt opp Apache + PHP på serveren • Testa å kjøre PHP fra kommandolinja • 09:00-15:00
11.02.11	• Jobba med uthenting av e-postadresse fra LDAP og generering av xml-dokumenter. • 09:00 - 15:00
10.02.11	• Koda hver for oss på LDAP og TimeEdit-strømmen.
09.02.11	• Funnet ut hvordan vi gjør det med scheduling av TimeEdit-jobber. Fortsatt med LDAP og TimeEdit-strømmen. • 09:00 - 15:00
08.02.11	• Sett på design. Fått relativt god oversikt over hvordan de ulike delene må jobbe sammen. • 09:00-15:00
07.02.11	• Vi har funnet ut hvor de ferdige medie-filene blir lagret og mappe- og filstrukturen der. Vi har også testa hvordan vi kan overvåke disse automagisk ved hjelp av bash-script • Det virker som det lønner seg å legge e-post-adressa til foreleser som author. Dette for å få henta ut denne lettere seinere. • Lokal PHP-server er up and running • 09:00-15:00
04.02.11	• i dag har fått til å laste opp “mediapakker” til matterhorn. Dette er eksterne videoer tatt opp utenfor matterhorn som det er lagt metadata til og zippet som kan sendes til matterhorn for deretter å bli en del av matterhornsystemet. Dette er et stort fremskritt da vi ønsker å implementere et system som kan starte og stoppe opptak etter ønske uten å legge det i scheduleren til matterhorn.

	• vga2usb fungerer ikke lenger, dette er litt kjedelig siden dette var det første vi fikk til å funke. Etter å ha fått opp "Proof of concept" på opplasting av ekstern video har vi brukt så og si hele dagen på å identifisere problemet med vga2usb • 0900 - 1530
03.02.11	• Lest en del i kildekoden til matterhorn • Funnet ut hvordan videofilenen fra matterhorn-CA sendes over til admin-boksen • Starta å teste med våre egne filer • 0900-1500
02.02.11	• Møte med stian • skal lese i kildekoden til matterhorn for å se om vi kan utvide den programvaren til å gjøre det vi ønsker • hvis ikke skal i første omgang prøve å integrere med timeedit slik som først planlagt • deretter skal vi prøve å gjøre opptak "manuelt", på capture-boksen, dvs uten matterhorn CA-software. Og deretter pushe dette med metadata inn i matterhorn, vi må finne ut om dette fungerer for flere videokilder slik at dette kan spilles av samtidig i samme videoavspiller i kjent matterhornstil • Klientside blir utviklet med JavaScript • serverside blir utviklet med PHP • lagd highlevel kravspec • 0900 - 1500
31.01.11	• Snakket om hva slags løsning vi vil ha, om timeedit er pålitelig nok osv • satt opp en high-level kravspec for et system med start og stopp-knapp • fikk svar på e-post ang grønt bilde, men det var svar på et annet problem som vi ikke har... • sendt e-post om at dette ikke var problemet
28.01.11	• Siste touch på forprosjektrapport • Sliter fortsatt med grønt bilde har sendt e-post til matterhorn-miljøet om det
27.01.11	
26.01.11	• Møte med veileder • Levert kontrakt • Fått admin og opptaks-PC til å samarbeide

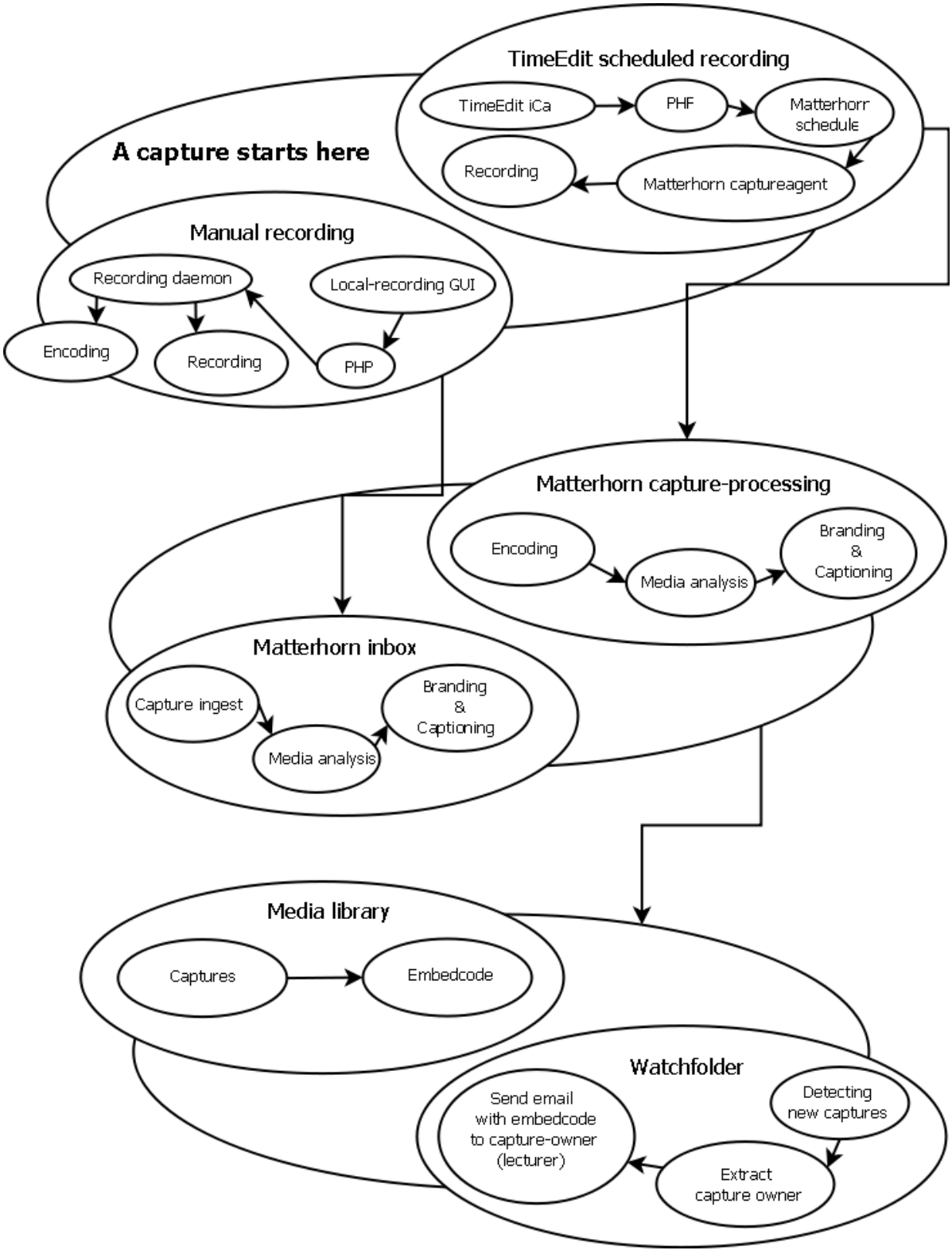
	"Prosjektor-opptak" fungerer. Når det gjelder opptak fra kamera, får vi videostrømmen, men det virker som GStreamer gjør om videosignalet til en grønn video. Det er riktig bilde når vi kjører rått fra kilden på opptaks-PCn. • 09:00-16:00
25.01.11	• Forprosjektrapport - presiseringer • Lært mer om matterhorn • Satt opp ny admin-PC for matterhorn • 1000-1600
24.01.11	• Risikoanalyse • jobba med/utforska matterhorn • 0900-1430
20.01.11	• Forprosjektrapport • Webside • 09:00-14:00
19.01.11	• Forprosjektrapport • lagd gantt-skjema, dog uten datoer • fått tips ang kamera fra Rolf • blitt enige om systemutviklingsmodell • 10:00-15:00
18.01.11	• Forprosjektrapport • kamerasøk • lastet ned opensource gantt-program, blitt kjent med • sendt e-post til rrolf@uni-osnabrueck.de som har vært med på utrulling av et matterhorn-prosjekt på en skole i tyskland • 0900-1530
17.01.11	• Sett etter analoge kameraer, dette viser seg å være vanskelig å finne info om på nettet(fastmontert osv) • Fant ut hvordan man henter ut RSS-feeder fra matterhorn. Dette må være en "serie" med opptak i matterhorn. • Begynt å installere vga-skjermdumpfangeren. Dette er en vga->usb enhet. Fant etter en del googling ut at denne enhetsdriveren ikke lar seg installere på denne linuxkjernen. Er ikke helt sikker på hva vi gjør med dette.

	• Begynte å teste uthenting av info fra TimeEdit. Dette ser ut til å være en grei sak.
	• Timer:
	  ○ Simen: 5
	  ○ Marius: 5

14.01.11	• Installert Ubuntu 10.04 LTS på Matterhorn-maskina
	• Installert Matterhorn V1.0 på Matterhorn-maskina
	• Lastet opp en videofil for å bli kjent med Matterhorn-SW
	• Opplastingen tok meget lang tid ettersom Matterhorn enkoder og gjør en del andre ting med fila.
	• Finner en mindre videofil nestegang...
	• Hadde med et kamera som vi skulle koble på, men manglet en overgang
	• Timer:
	  ○ Simen: 4
	  ○ Marius: 4

13.01.11	• Møte med veileder og delvis oppdragsgiver der vi diskuterte:
	  ○ kamera-> analog/digital
	  ○ Hvilket auditorium
	  ○ enkel eller dobbel videostrøm-> kamera og kanon
	  ○ lyd -> hva gjør vi under testing
	• Funnet ut at ubuntu er å foretrekke som OS http://opencast.jira.com/wiki/display/MH/Install+Binary+All+in+One+v1.0
	• Foreløpig støtter ikke Matterhorn IP-kamera, bruker analogt inntil videre
	• Timer:
	  ○ Simen: 6
	  ○ Marius: 6

Vedlegg D – Oversiktstegning over systemet, lagd for blogg



Vedlegg E – Forprosjektrapport uten vedlegg

Prosjektplan

Tittel: Automatiserte opptak av forelesninger

Navn: Marius Slåtsveen og Simen Bragen

Emnekode: IMT3912
http://www.hig.no/imt/bacheloroppgaver/retningslinjer

Studieretning: Bachelor i Ingeniørfag Data

Veileder: Kjell Are Refsvik

Dato: 28.01.11

1. MÅL OG RAMMER

1.1. Bakgrunn

I dagens samfunn driver mange studenter med flere aktiviteter parallelt med å gå på skole. Derfor er det viktig å ha et så fleksibelt undervisningsopplegg som mulig. Mange har en deltidsjobb som kanskje gjør at man går glipp en eller flere forelesninger.

Utover vanlige forelesninger finnes det forskjellige tilbud fra de mange foreleserne på skolen i dag.

- På avdelingen TØL holdes det mange sanntids videoforelesninger.
- En del lærere legger ut video av seg selv og/eller foiler med lyd.
- Andre tar manuelle opptak av sin egen forelesning og legger dette ut i ettertid.

1.2. Problembeskrivelse

Selv om det tildels brukes mye tid og ressurser på å gi studentene bedre forelesningstilbud er det en del problemer.

- Sanntids videoforelesninger passer i liten grad til fjernstudenter og deltidsstudenter.
- Lærere som legger ut video og/eller foiler med lyd bruker masse tid på å produsere dette materialet i tillegg til å forberede og holde vanlige forelesninger.
- De som tar manuelle opptak av forelesningene sine bruker mye tid og ressurser både før og etter forelesningene sine på dette.
- Det er teknisk krevende å produsere og legge slike forelesninger ut til studentene. Dette gjør det vanskelig for de foreleserne som ikke er kyndige nok til å få til dette.

Verktøyet som tilbys foreleserne ved HIG i dag er Elluminate. Dette er et veldig kraftig verktøy, men det faktum at det er så omfattende gjør at det krever opplæring og/eller en del erfaring før man kan ta det i bruk. Elluminate er i tillegg i stor grad lagt opp til livepublisering og manuell produksjon av video.

1.3. Målbeskrivelse

Målet vårt er å automatisere opptak av forelesninger og formidling av disse til studentene.

Vårt prosjekt skal være et lavterskeltilbud til de foreleserne som ikke har vilje eller evne til å bruke f.eks. Elluminate, og et supplement til Elluminate som i hovedsak er live-basert og eventuelt for produksjon av manuelt behandlede videoer.

Dette vil vi gjøre ved hjelp av Matterhorn og eksisterende datasystemer ved HIG. Vi vil også formidle opptakene til foreleseren slik at han/hun kan bestemme om og hvordan disse skal bli gjort tilgjengelig for studentene.

Se vedlegg for eksempler fra Elluminate.

Resultatmål

Vi skal lage rutiner som knytter sammen eksisterende systemer som muliggjør automatiserte opptak av forelesninger og se hvilke utfordringer som må jobbes videre med før dette kan bli en realitet.

Effektmål

Forenkle hverdagen til foreleserne ved å automatisere opptak og dermed også utvide studietilbudet til studentene.

1.4. Rammer

Dette er en bacheloroppgave ved Høgskolen i Gjøvik og tiden er dermed begrenset til bachelorperioden. Nødvendig hardware for utføring av oppgaven blir gitt av oppdragsgiver. Utviklingsverktøy finnes på skolen, ellers holdes det av studentgruppen.

Det eventuelle ferdige produkt eller den uferdige kildekoden vil bli gjort tilgjengelig under opensource-lisens.

2. OMFANG

2.1. Oppgavebeskrivelse

Vi skal utvikle et system som automatisk starter opptak av forelesninger. Etter opptaket er gjort skal foreleseren få mulighet til å enten legge opptaket tilgjengelig for studentene eller av ulike årsaker ikke gjøre det tilgjengelig. Dersom foreleseren vil gjøre alle fremtidige opptak tilgjengelig på en link får han mulighet til dette.

2.2. Avgrensning

Vi kommer ikke til å integrere mulighet for live-streaming i oppgaven. Oppgaven vil fungere som et "proof of concept", dvs. at vi vil fokusere på å få opp et system som fungerer for et rom og deretter se på muligheten for hvordan dette kan skaleres. Vi kommer ikke til å konsentrere oss om kvalitet på lyd og bilde i særlig stor grad under gjennomføringen av prosjektet, men heller bruke litt tid på slutten å reflektere og kommentere i rapporten over hva vi tror gir best resultat på området.

3. PROSJEKTORGANISERING

3.1. Ansvarsforhold og roller

Prosjektleder har overordnet ansvar for fremdriften og at fremdriftsplanen følges.

Simen Bragen har ansvaret for å opprette og oppdatere prosjekthjemmesiden jevnlig.

3.2. Rutiner og regler i gruppa

a) Ved uenighet om hvordan ting skal løses, tas beslutning i samråd med veileder og oppdragsgiver.
b) Marius Slåtsveen er Prosjektleder.
c) Dersom et gruppemedlem ikke utfører avtalt arbeid gis det mulighet til å ta igjen dette. Dersom arbeid ikke blir gjort fortest mulig se punkt d.
d) Ved sterkt mislighold av sine oppgaver kan et av gruppemedlemmene i samarbeid med veileder og oppdragsgiver bestemme at det andre gruppemedlemmet skal avskjediges.
e) Dersom det oppstår kostnader til reise eller andre nødvendige ting søkes dette om hos arbeidsgiver. Dersom arbeidsgiver ikke gir støtte til dette eller støtten ikke dekker 100% av kostnaden, fordeles kostnaden likt mellom gruppemedlemmene.
f) Både Simen Bragen og Marius Slåtsveen har rett til å signere på vegne av gruppen.

4. PLANLEGGING, OPPFØLGING OG RAPPORTERING

4.1. Hovedinndeling av prosjektet

Når man driver med et stort prosjekt er det viktig å ha god struktur på arbeidet. Derfor lønner det seg å følge en utviklingsmodell. Til vårt prosjekt har vi valgt inkrementell utviklingsmetode. I denne modellen deles arbeidet inn i ulike moduler. Hver modul utvikles og testes ferdig før vi går videre på neste. Dette passer oss bra, siden vi skal utvikle flere enkeltstående deler som skal jobbe sammen. Vi vil på denne måten oppdage feil og mangler i systemet tidlig i hver modul. Kravene til de ulike delene er bestemt tidlig i prosjektet, så vi vet hele tiden hva som må utvikles. I denne modellen er det også stort fokus på dokumentasjon. Det er viktig siden andre skal ta i bruk, og muligens videreutvikle systemet senere.

4.2. Plan for statusmøter/rapporter

Vi holder møter med veileder etter avtale. Dette vil være en time per uke eller f.eks. 2 timer annenhver uke.

Det skal leveres 3 statusrapporter i løpet av perioden. Disse skal leveres senest kl. 15.00 den 08.03.11, 11.04.11 og 06.05.11.

5. ORGANISERING AV KVALITETSSIKRING

5.1. Dokumentasjon, standardbruk og kildekode

I tekstdokumentene vil vi bruke Høgskolen i Gjøvik og IMT sine retningslinjer for rapportskriving. Vi dokumenterer fremgangen i en logg der vi skriver litt om hva vi har gjort hver dag og antall timer brukt den dagen.

Vi ser for oss at vi kommer til å kode i språk som JavaScript, PHP og lignende og vil følge internasjonale kodeskikker så langt det lar seg gjøre.

5.2. Konfigurasjonsstyring

Vi bruker Sub Version for styring av utviklingen vi gjør. Til diverse tekstbehandling underveis bruker vi Google Docs.

5.3. Risikoanalyse (identifisere, analysere, tiltak, oppfølging)

Risikobeskrivelse	Sannsynlighet	Konsekvens	Risikovurdering	Strategi
Sykdom i gruppen(Kortvarig)	3	2	6	Spise C-vitamin
Ikke kompatibel HW	1	4	4	Foreligger anbefalt HW
Ikke kompatibel SW	6	2	12	Hode kontakt med RR
Manglende kompetanse	8	3	24	Ta hensyn i tidsskjema
Problemer med lyd/bilde	2	2	4	Spiller ingen stor rolle, PoC
Personvern	6	1	6	Manuell offentliggjøring av opptak, PoC

Skala fra 1 til 10, der 10 er verst.

6. PLAN FOR GJENNOMFØRING

6.1 Metoder og midler

Vi skal ha regelmessig kontakt med oppdragsgiver og veileder for å sørge for at vi er på rett spor og for å få innspill. Videre kontakter vi representanter fra Uninett for å utveksle erfaringer med matterhorn-miljøet der. Utover dette har vi også kontakt med Rolf Ruediger ved universitetet i Osnabrück som er involvert i utvikling av matterhorn og opptak av forelesninger ved sitt universitet.

Selve kjernen i prosjektet vil være Matterhorn. Det er et opensource-prosjekt som får inn videostrømmen og lager ferdige opptak. Dette skal vi kjøre på en PC med Ubuntu Linux. Vi vil bruke info fra eksisterende systemer på skolen til å trigge opptak og hente metadata.

Vi kommer til å par-programmere mye slik at begge skal ha forståelse for alle deler av systemet.

6.2 Gantt-skjema

Se vedlegg

Vedlegg F – Statusrapporter

Statusrapport 08.03.11

Forelesningsopptak

Vi ble under kravspesifiseringen enige med arbeidsgiver om å, i tillegg til det som står i forprosjektet, utvikle en modul som gjør det mulig å gjøre opptak som er ikke er planlagt i TimeEdit. Dette gjøres ved å ta opptak lokalt på capture-agenten, for deretter å manuelt å dytte opptaket til matterhorn. Opptaket blir deretter en del av matterhorn på lik linje med opptak som er planlagt via matterhorn.

Vi ligger meget godt an tidsmessig i forhold til det opprinnelige Gantt-skjemaet. Den nye modulen gjør at tiden kniper litt, men vi har fått fordelt oppgavene mellom oss slik at vi er ganske sikre på å komme i mål med utviklingen før endelig testingen skal begynne.

For tiden arbeider vi med start og stopp av lokale opptak, i tillegg til generering av metadata for disse opptakene. Dette arbeidet er godt i gang, men vi har ikke kommet så langt at vi kan teste start/stopp og metadata-modulene sammen.

Vi synes arbeidet hittil har gått velig bra og er motiverte for å komme i mål innenfor tidsfristen.

Totalt sett ligger vi an til å overholde tidsfristen, selv med en ekstra modul.

Simen Bragen
Marius Slåtsveen 07.03.11

Statusrapport 13.04.11

Forelesningsopptak

Vi er nå inne i testperioden. Parallelt med testing har vi den siste tida jobba med en streaming-løsning. Den fungerer slik at når man tar manuelt opptak via GUI vil dette også streames på nett. Vi sender det via udp til VLC, så skal det fra VLC til skolens streamingserver. Oppdragsgiver ønsker da å kunne skrive for eksempel K105.hig.no og få streamen fra rommet dersom den er online. Dette blir utenfor vår oppgave, men vi må ta hensyn til at vår stream skal kunne brukes på denne måten.

Ellers er utstyret satt opp i K105, og Are Strandli tester det denne uka.

Opptak via vårt eget GUI fungerer som det skal. Det tas opp og vi generer metafiler. Disse opptaka vises i Matterhorn på lik linje med de automatiske fra TimeEdit.

Vi har enda ikke fått testa automatisk e-postutsendelse på grunn av problemer med å få satt opp riktig e-post-server mot PHP. De ulike delene fungerer, så vi håper å få dette i orden i løpet av neste uke.

Testperioden skal være ferdig 23. april og det ser ut til at vi klarer dette med en del jobb i påska.

Simen Bragen
Marius Slåtsveen 13.04.11

Statusrapport 09.05.11

Forelesningsopptak

Siden forrige statusrapport har vi satt opp e-posten på serveren riktig, og testet at dette fungerer som det skal. Videre har vi gått igjennom all kode og kommentert grundig.
Are Strandlie har gjort et testopptak av en dobbelttime i K105. Vi synes dette gikk meget bra og opptaket ligger ferdig prosessert i Matterhorn.

Vi holder for tiden på å skrive rapporten og har begynt å få en del ned på papir.
Vi har et førsteutkast på alle de store kapitlene, dvs innledning, kravspesifikasjon, design, implementering, testing og avslutning. Selve skallet er klart, men det mangler fortsatt noen punkter, og flere av punktene vi har må nok endres og/eller utvides.
Tidsskjemaet er overholdt ganske bra og vi håper å ha et godt produkt klart innen innleveringsfristen 25. mai.

I tillegg til rapporten lager vi en del figurerer som skal være med, og vi forsøker å få laget a3-plakten innen rapporten skal leveres så den kan være med som et vedlegg.

Marius Slåtsveen

Simen Bragen

09.05.11

Vedlegg G – Prosjektavtale

HØGSKOLEN I GJØVIK

PROSJEKTAVTALE

mellom Høgskolen i Gjøvik (HiG) (utdanningsinstitusjon),

Høgskolen i Gjøvik, IT-avdelingen
(oppdragsgiver), og

Tarjei Slåtveen

Simen Bungum
(student(er))

Avtalen angir avtalepartenes plikter vedrørende gjennomføring av prosjektet og rettigheter til anvendelse av de resultater som prosjektet frembringer:

1. Studenten(e) skal gjennomføre prosjektet i perioden fra 01.02.11 til 25.05.11.

   Studentene skal i denne perioden følge en oppsatt fremdriftsplan der HiG yter veiledning. Oppdragsgiver yter avtalt prosjektbistand til fastsatte tider. Oppdragsgiver stiller til rådighet kunnskap og materiale som er nødvendig for å få gjennomført prosjektet. Det forutsettes at de gitte problemstillinger det arbeides med er aktuelle og på et nivå tilpasset studentenes faglige kunnskaper. Oppdragsgiver plikter på forespørsel fra HiG å gi en vurdering av prosjektet vederlagsfritt.

2. Kostnadene ved gjennomføringen av prosjektet dekkes på følgende måte:
   - Oppdragsgiver dekker selv gjennomføring av prosjektet når det gjelder f.eks. materiell, telefon/fax, reiser og nødvendig overnatting på steder langt fra HiG. Studentene dekker utgifter for trykking og ferdigstillelse av den skriftlige besvarelsen vedrørende prosjektet.
   - Eiendomsretten til eventuell prototyp tilfaller den som har betalt komponenter og materiell mv. som er brukt til prototypen. Dersom det er nødvendig med større og/eller spesielle investeringer for å få gjennomført prosjektet, må det gjøres en egen avtale mellom partene om eventuell kostnadsfordeling og eiendomsrett.

3. HiG står ikke som garantist for at det oppdragsgiver har bestilt fungerer etter hensikten, ei heller at prosjektet
   blir fullført. Prosjektet må anses som en eksamensrelatert oppgave som blir bedømt av faglærer/veileder og
   sensor. Likevel er det en forpliktelse for utøverne av prosjektet å fullføre dette til avtalte

1

spesifikasjoner,
funksjonsnivå og tider.

4. Den totale besvarelsen med tegninger, modeller og apparatur så vel som programlisting, kildekode, disketter,
taper mv. som inngår som del av eller vedlegg til besvarelsen, gis det en kopi av til HiG, som vederlagsfritt kan benyttes til undervisnings- og forskningsformål. Besvarelsen, eller vedlegg til den, må ikke nyttes av HiG til andre formål, og ikke overlates til utenforstående uten etter avtale med de øvrige parter i denne avtalen. Dette gjelder også firmaer hvor ansatte ved HiG og/eller studenter har interesser.

Besvarelser med karakter C eller bedre registreres og plasseres i skolens bibliotek. Det legges også ut en elektronisk prosjektbesvarelse uten vedlegg på bibliotekets del av skolens internettsider. Dette avhenger av at studentene skriver under på en egen avtale hvor de gir biblioteket tillatelse til at deres hovedprosjekt blir gjort tilgjengelig i papir og nettutgave (jfr. Lov om opphavsrett). Oppdragsgiver og veileder godtar slik offentliggjøring når de signerer denne prosjektavtalen, og må evt. gi skriftlig melding til studenter og dekan om de i løpet av prosjektet endrer syn på slik offentliggjøring.

• Besvarelsens spesifikasjoner og resultat kan anvendes i oppdragsgivers egen virksomhet. Gjør studenten(e) i sin besvarelse, eller under arbeidet med den, en patentbar oppfinnelse, gjelder i forholdet mellom oppdragsgiver og student(er) bestemmelsene i Lov om retten til oppfinnelser av 17. april 1970, §§ 4-10.

• Ut over den offentliggjøring som er nevnt i punkt 4 har studenten(e) ikke rett til å publisere sin besvarelse, det være seg helt eller delvis eller som del i annet arbeide, uten samtykke fra oppdragsgiver. Tilsvarende samtykke må foreligge i forholdet mellom student(er) og faglærer/veileder for det materialet som faglærer/veileder stiller til disposisjon.

7. Studenten(e) leverer oppgavebesvarelsen med vedlegg (pdf) i Fronter. I tillegg leveres et eksemplar til oppdragsgiver.

8. Denne avtalen utferdiges med et eksemplar til hver av partene. På vegne av HiG er det dekan/prodekan som godkjenner avtalen.

9. I det enkelte tilfelle kan det inngås egen avtale mellom oppdragsgiver, student(er) og HiG som nærmere
regulerer forhold vedrørende bl.a. eiendomsrett, videre bruk, konfidensialitet, kostnadsdekning og økonomisk utnyttelse av resultatene.

Dersom oppdragsgiver og student(er) ønsker en videre eller ny avtale, skjer dette uten HiG som partner.

10. Når HiG også opptrer som oppdragsgiver trer HiG inn i kontrakten både som utdanningsinstitusjon og som oppdragsgiver.

11. Eventuell uenighet vedrørende forståelse av denne avtale løses ved forhandlinger avtalepartene i mellom. Dersom det ikke oppnås enighet, er partene enige om at tvisten løses av voldgift, etter bestemmelsene i tvistemålsloven av 13.8.1915 nr. 6, kapittel 32.

12. Deltakende personer ved prosjektgjennomføringen:

HiGs veileder (navn): Kjell Are Refsvik

Oppdragsgivers

kontaktperson (navn): Stian Husemoen

Student(er) (signatur): _____ dato 26.01.11

_____ dato 26.01.11

_____ dato

_____ dato

Oppdragsgiver (signatur): _____ dato

IMT Dekan/prodekan (signatur): _____ dato 02.02.2011

Versjon Januar 2011hb