Avdeling for **T**eknologi

HOVEDPROSJEKT:

TITTEL

Dynamisk informasjonskanal i Macromedia Flash 5

FORFATTERE:
Berit Benjaminsen
Janne Rødsand
Gro Vogt Sæther

Dato: Mai 2001

Avdeling for **T**eknologi

SAMMENDRAG AV HOVEDPROSJEKT

Tittel:	Dynamisk informasjonskanal i Macromedia Flash 5	Nr. :
		Dato : 22.05.01
Deltaker(e):	Janne Rødsand	
	Gro Vogt Sæther	
	Berit Benjaminsen	
Veileder(e):	Paal Børre Sveum	
Oppdragsgiver:	Framfab	
Kontaktperson:	Martin Forsberg Lie	
Stikkord (4 stk)	MS SQL Server, ASP, Macromedia Flash 5, prototype	

Antall sider:	Antall bilag:	Tilgjengelighet (åpen/konfidensiell):

Kort beskrivelse av hovedprosjektet:

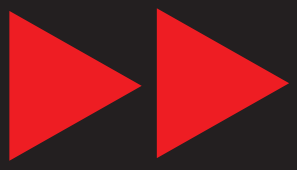
Prosjektet har gått ut på å finne ut om vektorteknologier som Macromedia Flash 5 fungerer til dynamisk presentasjon av multimediell informasjon. Det ble utviklet en prototype på en dynamisk informasjonskanal for å teste ut problemstillingen.

Systemet ble bygd opp med en MS SQL Server-database til lagring av informasjon. Macromedia Flash 5 ble benyttet til visuell presentasjon av innholdet i databasen og ASP ble benyttet til kommunikasjon mellom databasen og Flash. Systemet skulle kunne håndtere tekst, lyd, bilde og video.

Informasjonskanalen skulle hente opp informasjon i tilfeldig rekkefølge fra databasen. Denne rekkefølgen skulle imidlertid kunne overstyres av informasjon som det var ønskelig å vise på et bestemt tidspunkt. Informasjonen skulle presenteres på en TV-skjerm.

Det må påpekes at gruppa, ved prosjektets start, var klar over at applikasjonen Macromedia Generator benyttes som et mellomledd på serveren mellom en dynamisk Flashfil og en database. Macromedia Generator er et kostbart verktøy og studentene fikk fra flere hold høre at det var noe ustabilt i bruk. Studentene var derfor interessert i å finne ut om det var mulig å oppnå det samme ved å bare bruke ASP i kombinasjon med Flash.

Dynamisk informasjons-kanal i Macromedia Flash 5



Hovedprosjekt våren 2001

Forord

I siste semester av grafisk ingeniørutdanning ved Høgskolen i Gjøvik skal det gjennomføres et hovedprosjekt som utgjør 6 vekttall. Dette tilsvarer en arbeidsmengde på omtrent 360 timer per prosjektdeltaker.

Vi var i kontakt med flere bedrifter for å finne oppdragsgiver til prosjektet. Framfab var den bedriften som utmerket seg positivt fordi de hadde en konkret oppgave å tilby. Derfor valgte vi å samarbeide videre med Framfab.

Denne rapporten inneholder dokumentasjon fra arbeidet med å utvikle en prototype på en dynamisk informasjonskanal basert på vektorteknologi. Vi har gjennom arbeidet med rapporten ønsket å belyse både tekniske og organisasjonsmessige aspekter ved prosjektarbeidet. Dette er fordi målet med prosjektet for prosjektgruppas del har vært å oppnå økt kompetanse innen disse områdene.

Rapporten er en del av besvarelsen av prosjektoppgaven. Det er meningen at rapporten skal kunne leses av studenter og forelesere med interesse innen fagområder som dynamisk presentasjon av informasjon og vektorteknologi.
Til slutt vil vi rette en takk til alle som har bidratt med råd og vink underveis i prosjektet:
- Martin Forsberg Lie, som har vært gruppas kontaktperson hos oppdragsgiver
- Jørn Madsen, som har vært gruppas tekniske kontaktperson hos Framfab
- Paal Børre Sveum, som har vært studentenes veileder
- Øvrige ansatte i Framfab, som har bidratt med hjelp til å løse tekniske problemer
- Øivind Kolloen, som har tatt seg tid til å svare på spørsmål underveis i prosjektet

Gjøvik, mai 2001

Janne Rødsand

Gro Vogt Sæther

Berit Benjaminsen

Innholdsfortegnelse

5 Utstyr benyttet i prosjektet

6 Kravspesifikasjon for systemet

7 Design

8 Utvikling

9 Testing

1 Innledning

1.1 Organisering av rapporten

De første tre kapitlene av rapporten retter seg først og fremst mot bakgrunnen og målet for prosjektet samt organisering og gjennomføringen av prosjektet. Fra kapittel fire er rapporten vinklet mot mer tekniske aspekter ved prosjektet og selve utviklingen av prototypen. De siste kapitlene i rapporten (kapittel 10-13) inneholder resultater, konklusjon og egenvurdering av prosjektet. Siden prosjektarbeidet har gått ut på å utvikle en prototype inneholder siste del av rapporten også et kapittel med tips til videreutvikling. Bakerst i rapporten finnes en ordliste med forklaringer av begreper som er benyttet i rapporten. Det anbefales å slå opp i denne dersom man skulle møte på fagord som man ikke kjenner til. I tillegg følger all kildekode og annen dokumentasjon med som vedlegg.

1.2 Oppgavedefinisjon

1.2.1 Problemstilling

Kan vektorbaserte bevegelsesteknologier som Macromedia Flash 5 fungere som et rammeverk for dynamiske multimedieinformasjonskanaler i kombinasjon med Active Server Pages (ASP)? Oppgaven vil ta utgangspunkt i å trekke erfaringer fra et prototypearbeid hvor prinsippene i problemstillingen blir utprøvd i mindre skala.

1.2.2 Resultatmål

Resultatet skal være en prototype på en dynamisk informasjonskanal i Macromedia Flash 5 som kommuniserer med databaser gjennom ASP. Informasjonskanalen skal kunne tas i bruk i Framfabs lokaler i Halden hvor det visuelle skal presenteres på TV-skjerm. Prototypen skal danne grunnlag for eventuell videreutvikling av systemet. I tillegg skal prosjektet danne grunnlag for vurdering om hvor godt Macromedia Flash 5 fungerer i samspill med databaser og ASP, i første rekke som rammeverk for en dynamisk informasjonskanal.

1.2.3 Gruppas effektmål

Effektmålet for gruppa har vært å tilegne seg gode fagkunnskaper innen ASP, databaser og Flash. I tillegg vil gjennomføringen av prosjektet føre til at gruppas medlemmer får erfaring med å jobbe i et langvarig prosjekt hvor man fordyper seg innenfor et fagområde. Dette innebærer erverving av kunnskap og erfaringer innen planlegging av oppgaver, struktu-

rering av arbeidet, oppfølging av prosjektets framdrift, vurdering av eventuelle tiltak som må igangsettes, samarbeid mellom prosjektdeltakerne, oppdragsgiver og veileder, dokumentering av arbeidet samt rapportskriving.

1.2.4 Framfabs effektmål

Ut i fra prosjektets konklusjon, bør man vurdere om det vil være mulig å benytte prinsippene i problemstillingen i andre sammenhenger, som for eksempel for internettprosjekter. Dersom teknologien egner seg godt, vil dette åpne for nye muligheter når det gjelder dynamisk presentasjon av informasjon. Man kan også tenke seg automatisk produsering av internettinnhold basert på eksisterende databaser, men vel så viktig er det å kunne skape et rammeverk for kommunikasjon, uten at teknologien i seg selv skal være et hinder.

1.2.5 Oppgavebeskrivelse

Prototypen skal baseres på vektorteknologi. Systemet skal kjøres i en loop, eller en repeterende løkke, som henter informasjon fra databaser i en tilfeldig rekkefølge. Loopen skal i tillegg kunne overstyres av informasjon som skal vises på bestemte tidspunkt. Informasjonskanalen skal håndtere video, tekst og bilder, og skal i tillegg tilrettelegges for lyd. Innholdet som skal presenteres skal ligge i MS SQL Server-databaser. Selve systemet skal utvikles i Macromedia Flash 5. ASP skal benyttes som et mellomledd på serveren slik at Flashakomponenten skal kunne kommunisere med databasene.

Informasjonen skal vises på TV-skjerm. Oppløsningen som skal benyttes i Flash-fila må tilpasses de antall linjer som vises på en TV-skjerm. Alt det visuelle skal være i henhold til Framfabs designmanual. Ved hjelp av ASP og HTML skal det lages et administrasjonsverktøy hvor man kan styre det som skal vises i informasjonskanalen samt legge til ny informasjon. En manual som inneholder en kort innføring i bruken av systemet skal følge prototypen.

1.2.6 Avgrensning

Det faller utenfor oppgaven å produsere innholdet som skal vises fram. Den pedagogiske tilretteleggingen av innholdet inngår heller ikke i prosjektet. Systemet vil ikke bli testet på forskjellige nettlesere. Problematikken med ulike nettforbindelser, som for eksempel modem kontra bredbånd, vil heller ikke bli tatt med i betraktningen. Studentene er ikke ansvarlig for den fysiske monteringen av systemet i Framfabs lokaler. Prototypen vil bli overlevert i den stand den foreligger på framføringsdagen, 31. mai 2001 og etter dette har ikke studentene noe videre ansvar for systemet.

1.3 Målgruppe for rapporten

Studenter og forelesere ved Høgskolen i Gjøvik samt interesserte ansatte i Framfab vil være rapportens målgruppe.

1.4 Målgruppe for prototypen

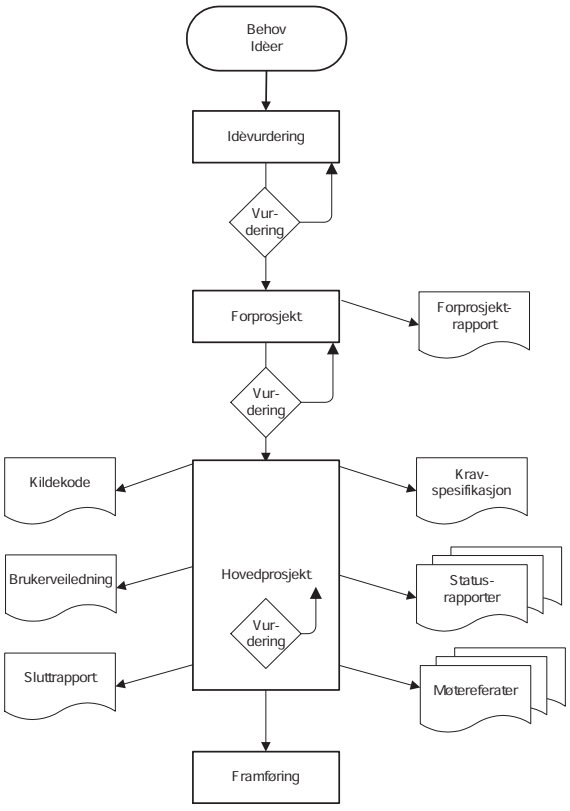
Målgruppen for prototypen er Framfabs ansatte i Halden.

1.5 Brukere av prototypen

Brukeren er den eller de personene i Framfab som skal administrere systemet. Det skal utvikles et administrasjonsverktøy med et web-grensesnitt. Dermed trenger ikke brukeren å ha database- eller programmeringskunnskaper for å benytte systemet. Det vil følge en brukerveiledning med prototypen, se vedlegg A. Øvrig dokumentasjon på hvordan prototypen fungerer vil være å finne i prosjektrapporten.

1.6 Arbeidsformer

Studentene har hatt hovedansvar for utføringen av ulike deler av prosjektet, mens teknisk design og strukturoppbygning har foregått i fellesskap. I forkant av utviklingsfasen ble det gjennomført en egenlæringsperiode som for det meste besto av litteraturstudier. Gruppa forsøkte å ha det nokså klart hvordan systemet skulle fungere i teorien før selve kodingen og utviklingen startet. Dette for at man skulle slippe å bruke unødvendig mye tid til prøving og feiling før man fant en god løsning.
Prosjektforløpet er illustrert nedenfor:



Figur 1.1 Gangen i prosjektet

Jobbingen med prosjektet ble delt inn i flere faser; oppstart, forprosjekt, egenlæring, utvikling, testing, rapportsammenstilling og organisering av framføring. Det ble i utgangspunktet planlagt at utviklingen av databasen, ASP- og Flash-dokumentene skulle foregå parallelt. Dette ble ikke tilfelle. Det viset seg at implementeringen av databasen gikk forholdsvis raskt. Dermed gikk siste del av utviklingsperioden bare med til implementering av ASP og Flash.

1.7 Terminologi

Det har blitt lagt vekt på å benytte et lettfattelig språk i rapporten med mest mulig norske ord. Det har til tider vært vanskelig å finne gode norske uttrykk når terminologien innenfor data hovedsaklig baserer seg på engelsk. Norsk Språkråd sine anbefalinger er benyttet under skriving av rapporten.

2 Bakgrunn for prosjektet

2.1 Gruppas bakgrunn

Gruppa hadde et ønske om å jobbe med vektorbaserte teknologier for å lære mer om dette. Derfor henvendte studentene seg til ulike bedrifter for å høre om de hadde prosjekter innenfor området. Framfab ble valgt da de hadde et konkret og interessant prosjekt innenfor dette temaet.

2.2 Oppdragsgivers bakgrunn

Framfabs avdeling i Halden flyttet inn i nye lokaler i begynnelsen av januar 2001. I disse lokalene ønsket de å ha en form for dynamisk informasjonskanal som skulle komme til uttrykk gjennom TV-skjermer. Informasjonskanalen skulle rettes både mot de ansatte og for kunder som kommer innom kontoret. Fram til nå hadde ingen av de ansatte hatt tid til å ta seg av utviklingen av en slik informasjonskanal. Derfor passet det fint at studentene var interessert i denne oppgaven.

2.3 Studentenes faglige bakgrunn

Prosjektdeltagerne går tredje og siste året på Grafisk Ingeniørutdanning, studieretning for digital medieteknikk, ved Høgskolen i Gjøvik. Studiet gir kompetanse rettet mot produksjon av web- og multimedieapplikasjoner samt andre former for elektronisk publisering. Studentene har jobbet med medietyper som tekst,

bilder, video og lyd i ulike prosjekter tidligere og hadde grunnleggende kunnskaper innen programmeringsspråkene C og Java. I tillegg hadde de noe kunnskap om databaseteori, samt noe erfaring fra mindre prosjektarbeider.

3 Prosjektgjennomføring

3.1 Praktisk framgangsmåte

3.1.1 Oppstartsfase

Planleggingsarbeidet ble så vidt påbegynt i midten av november med det samme gruppa hadde en oppdragsgiver. Gruppas medlemmer hadde ikke samarbeidet tidligere. Derfor ble denne perioden benyttet til å bli bedre kjent samt at gruppa ble enige om hvordan jobbingen med prosjektet skulle utføres. Det ble utarbeidet et sett med grupperegler som skulle gjelde hele prosjektperioden, se vedlegg N.

3.1.2 Forprosjekt

Januar måned gikk med på å gjennomføre forprosjektet for selve hovedprosjektet, se vedlegg H. Da ingen av gruppas medlemmer hadde noe særlig kunnskaper om de teknologiene som skulle benyttes i prosjektet var dette en utfordrene oppgave. I forprosjektet ble det lagt vekt på planleggingen av gjennomføringen av selve hovedprosjektet, både med hensyn på arbeidsmetoder, fordeling av arbeidsoppgaver samt planlegging av tidsforbruk på de ulike aktivitetene. I tillegg ble det definert hva som inngikk i prosjektoppgaven og hva som ville falle utenfor samt at det ble utarbeidet mål for prosjektet.

3.1.3 Egenlæring

De tre første ukene etter innleveringen av forprosjektrapporten ble satt av til egenlæring. Gruppas medlemmer fikk tildelt hvert sitt faglige ansvarsområde og konsentrerte seg om dette. Tilegnelsen av kunnskap foregikk stort sett ved å lese litteratur. Ved oppstarten var det vanskelig å oppdrive litteratur på området Flash i kombinasjon med ASP, derfor ble internett benyttet. I tillegg fulgte studentene forelesningene i faget klient- og serversideprogrammering på Høgskolen i Gjøvik. Der var et av temaene ASP.

3.1.4 Utviklingsfase

Da egenlæringsfasen var over startet selve utviklingen av prototypen. Det ble utviklet en kravspesifikasjon for systemet. Denne ble lagt til grunn for informa-

sjonskanalens funksjonalitet. Dette var en tidkrevende periode og innebar mer koding enn det studentene på forhånd hadde regnet med. Utviklingsfasen ble avsluttet av en testperiode.

3.1.5 Sammenstilling av rapport

Da forprosjektet var fullført startet planleggingen av rapportens struktur og innhold. Dette fungerte bra fordi da fikk alle gruppemedlemmene en formening om hva sluttrapporten skulle inneholde tidlig i prosjektet. Det første utkastet har selvfølgelig blitt revidert underveis i prosjektet etter som studentene har følt behov for å ha med flere elementer, omorganisere på rekkefølgen og så videre.

De siste ukene før rapporten skulle leveres, ble benyttet til sammenstilling av rapporten. Selv om prosjektdeltakerne hadde forsøkt å være flinke til å dokumentere og skrive deler av rapporten underveis i prosjektgjennomføringen, var dette en tidkrevende prosess. Utfordringen lå i å skrive en enhetlig rapport, samt å unngå at innholdet ble gjentatt i flere avsnitt.

3.1.6 Planlegging av framføring

Planleggingen av framføringen ble diskutert på et tidlig tidspunkt under utviklingsfasen i prosjektet. Da ble det kommet fram til at det kunne være problematisk å få vist produktet på Høgskolen i Gjøvik. Problemet var at man ikke kunne koble seg opp til oppdragsgivers nettverk fra skolen, annet enn via oppringt forbindelse (ISDN). Dette ville kunne skape problemer med overføringen av data i og med at systemet ikke er tilpasset bruk på linjer med liten båndbredde. For å komme rundt denne problemstillingen ble det vurdert å lage en slags «dummy» på kommunikasjonen mellom databasen og informasjonskanalen hvor alt innholdet ble lagret statisk i en Flashfil. På denne måten ville man kunne gi tilhørerne et inntrykk av hvordan systemet så ut i praksis.

Etter å ha snakket med oppdragsgiver ble det klart at framføringsproblematikken kunnes løses nokså enkelt. Ved å installere en desktopserver på en bærbar PC og kopiere nødvendige filer og databaser over på denne ville hele systemet kunne kjøres lokalt på denne PC-en under framføringen. Dermed var det ikke nødvendig å lage en «dummy».

3.2 Organisering av kvalitetssikring

Gruppa har benyttet Framfabs server i Halden under prosjektperioden. Alt arbeid ble lagret på denne serveren. Det tas sikkerhetskopi av serveren daglig. Derfor trengte ikke gruppa å tenke noe videre over dette. Hver enkelt har vært ansvarlig for å ta sikkerhetskopier av det som ble lagret andre steder.

Alle gruppemedlemmene har tatt vare på viktige e-poster som ble sendt og mottatt i løpet av prosjektet. Dermed hadde studentene all slik dokumentasjon tilgjengelig da rapporten skulle settes sammen.

I begynnelsen av prosjektperioden hadde en av prosjektdeltakerne hovedansvaret for å lagre alle dokumentene som hadde tilknytning til prosjektet på et samlet sted. Da alle prosjektdeltakerne var koblet til Framfabs server hadde hver enkelt ansvaret for å lagre de dokumentene de selv hadde opprettet på fellsområdet på serveren.

All aktivitet i gruppa har blitt dokumentert i en loggbok, se vedlegg K. Denne loggboka har vært til stor hjelp under skrivingen av rapporten da det har vært mulig å gå tilbake i den og kontrollere hva som faktisk ble gjort.

3.3 Kommunikasjon

Kommunikasjonen med oppdragsgiver foregikk i hovedsak via e-post, men under selve utviklingen ble det i tillegg benyttet telefon. Ved behov ble det holdt møter internt i gruppa, med oppdragsgiver eller med veileder på skolen. Et møtereferat ble distribuert til møtedeltagerne så raskt som mulig etter møtet. Møterefaratene er å finne i vedlegg L. Det ble holdt statusmøter hver fjortende dag. Referat fra disse møtene ble sendt kontaktpersonen hos oppdragsgiver samt veileder på skolen. På denne måten hadde oppdragsgiver og veileder mulighet for å følge prosjektets framdrift. Statusrapportene er å finne i vedlegg M.

3.4 Framdriftsplan

Hovedplanleggingen av prosjektet med hensyn på framdrift ble foretatt i forbindelse med forprosjektet. Da ble alle milepæler fastsatt samt at tidsbruk på de ulike deloppgavene som skulle føre fram til målet ble beregnet. For å presentere dette på en oversiktlig måte ble det benyttet et ganttskjema, se vedlegg I. Vedlegg J viser virkelig tidsforbruk på aktivitetene i forhold til den planlagte. Dette gjorde det enklere å holde oversikten over de arbeidsoppgavene som måtte utføres til enhver tid for at målet skulle bli nådd. Ganttskjemaet ble utdypet underveis i prosjektet etter hvert som nye oppgaver dukket opp.

Gruppa har kjørt nokså hardt på de tidsfrister som var planlagt. Dette er noe studentene fikk igjen for i sluttperioden av prosjektet. Dermed ble ikke siste innspurt med rapporten et hastverksarbeid. Se vedlagte ganttskjema for det planlagte tidsforbruket på de ulike aktivitetene og den virkelig benyttede tiden.

3.5 Erfaringer fra prosjektgjennomføringen

– Kommunikasjon er ikke alltid like lett, spesielt ikke når gruppa består av tre personer. Det er lett at to personer prater sammen og glemmer å informere den tredje. Dette ble gruppen klar over tidlig i prosjektet og var deretter veldig bevisste på at alle måtte informeres om alt som hadde med prosjektet å gjøre.

- Samarbeidet innad i gruppa har stort sett gått bra, men det oppstod en del kommunikasjonsproblemer i den siste perioden av prosjektet, da ikke gruppemedlemmene lenger satt samlet og arbeidet. Vi har derfor erfart at det er lettere, og mindre tidkrevende å kommunisere ansikt til ansikt enn via e-post og telefon.

- Det er vanskelig å gjøre et bra forprosjekt. Det er lett å være etterpåklok og si at undersøkelsene foretatt i forprosjektet burde vært gjort grundigere enn det de faktisk ble. Selv om det finnes lite kunnskap og litteratur på området burde kanskje studentene ha funnet ut at det var problematisk med slike filtyper litt tidligere i prosjektet, men på bakgrunn av den kunnskapen studentene satt inne med på forhånd var dette vanskelig. Den lærdommen som kan trekkes ut av dette er at arbeidet med forprosjektet er den viktigste delen av et prosjekt. Her legges grunnlaget for hele prosjektet, og dersom ikke alle aspekter er nøye gjennomtenkt kan det hende at man får noen ubehagelige overraskelser underveis. På en annen side finnes det prosjekter som er av mer forskningsmessig art, hvor man ønsker å belyse nye sider ved en sak og finne nye løsninger på et problem. I disse tilfellene kan man ikke undersøke absolutt alt i et forprosjekt, men må ta erfaringene etter hvert som de kommer. I dette prosjektet er vel en mellomting mellom de to metodene nevnt ovenfor benyttet. Det ble foretatt en del undersøkelser i forkant, men mange av erfaringene måtte komme etter hvert.

- Å lage en kravspesifikasjon er tidkrevende, men veldig nyttig. Det bør legges stor vekt på å få denne klar så tidlig som mulig i prosjektforløpet. Dette fordi en kravspesifikasjon ideelt sett skal godkjennes før selve utviklingen kan starte.

- Den måten studentene jobbet med selve utviklingen av prototypen burde kanskje vært annerledes. I ettertid kan man tenke at det kanskje ville vært mer effektivt å konsentrere seg om å få en type informasjon gjennom hele systemet fra en test-database og ut på TV-skjermen før det komplekse systemet skulle klare det. Dette ville vært mulig dersom man hadde benyttet seg av en arbeidsmetode som minner mer om prototyping. Da kunne løsninger på problemområdene knyttet til skjæringspunktet mellom ASP og Flash blitt funnet på et tidligere tidspunkt.

- Det er lurt å utarbeide dokumenter hvor man kan samle punkter som kan være nyttige å ha når prosjektforløpet skal dokumenteres. Gruppa opprettet dokumenter som inneholdt problemer som har oppstått under veis, hvordan disse ble løst samt beslutninger som ble tatt underveis i prosjektet. I tillegg ble det benyttet en loggbok. Dette var faktorer som gjorde det enklere å utarbeide rapporten.

- Det kan lønne seg å opprette et sett med regler for hvordan rapporten skal skrives med hensyn til ordbruk, valg av grammatiske regler og lignende. Dersom disse retningslinjene er gode samt at de blir fulgt vil rapporten bli noe mer helhetlig. Dette var en metode studentene benyttet og følte var nyttig.

– Det ble erfart at det er vanskelig å utarbeide en rapport med god flyt og uten gjentagelser. Dette fordi rapporten skrives av flere personer som legger vekt på ulike elementer samtidig som de har ulike måter å formulere seg på.

4 Prinsipper og teori

Dette er et kapittel hvor det vil bli gitt en kort innføring i fagfeltene som er benyttet for å utvikle systemet som dette prosjektet har resultert i.

4.1 Innføring i databaser

4.1.1 Hva er en database?

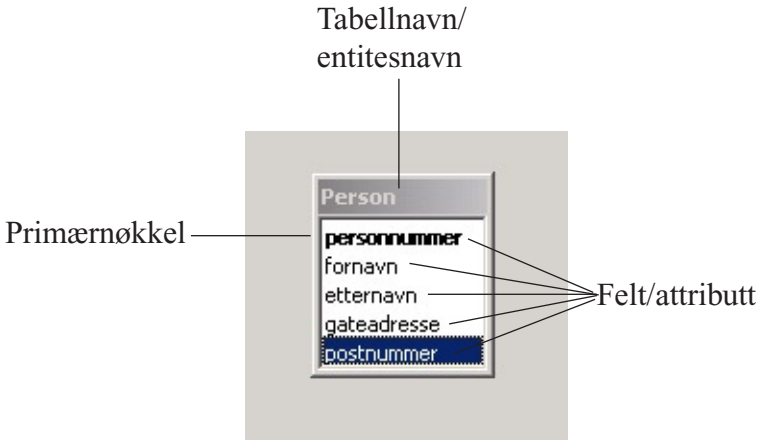
En database er en samling av data, som er organisert slik at opplysningene lett kan hentes opp igjen. Formålet med en database er at den skal fungere som et «sted» hvor man kan lagre informasjon som man på et senere tidspunkt ønsker å benytte.

4.1.2 Databaseteori

En database består av en eller flere tabeller som er inndelt i ulike felter. Det er i disse feltene selve informasjonen blir lagret. I databaseterminologien omtaler man ofte en tabell som en entitet. Feltene som hører til entitetene kalles attributter.

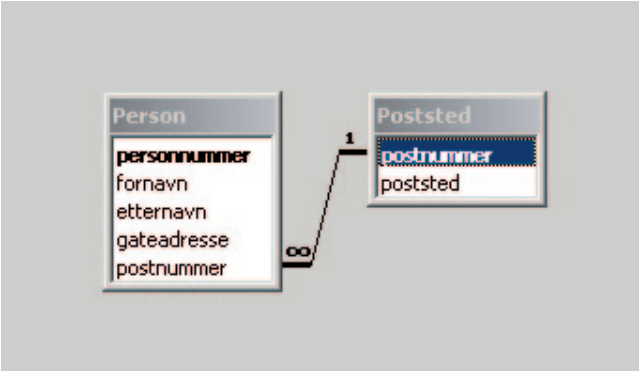
En tabell skal inneholde felter som på en logisk måte hører sammen, og hver tabell må gis et navn. Dette navnet bør gjenspeile den type informasjon som alle feltene i tabellen skal inneholde. Det er også lurt å gi feltene i tabellene logiske navn slik at det er lett for en utenforstående person å forstå hva slags informasjon de ulike feltene inneholder.

En tabell må alltid inneholde en primærnøkkel. En primærnøkkel fungerer som en unik identifikator for hver post som legges inn i databasen. Ett, eller en kombinasjon av flere, av feltene i hver tabell må defineres som en primærnøkkel. Dette gjør det mulig å skille dataene som legges inn i de ulike tabellene fra hverandre. En post tilsvarer all data som ligger lagret om et element i en tabell.



Figur 4.1 eksempel på oppbygningen av en databasetabell

Som regel består en database av flere tabeller. Disse tabellene kobles sammen og man sier at det opprettes en relasjon mellom dem. Dette gjøres ved at primærnøkkelen i hovedtabellen også inngår som et felt i den tabellen man ønsker å binde sammen med hovedtabellen. I den relaterte tabellen kalles dette feltet for en fremmednøkkel. De tabellene som skal relateres til hverandre må altså begge ha et felt som inneholder samme informasjon. Når man skal legge inn data i en tabell som inneholder en fremmednøkkel kan man kun legge inn en verdi i fremmednøkkelfeltet så lenge denne verdien allerede ligger i primærtabellen.



Figur 4.2 Eksempel på en relasjon mellom to tabeller

Når man skal opprette en database, gjelder det å organisere og systematisere de dataene som man ønsker å lagre på en logisk måte. Dette gjøres enklest ved å sortere elementene man ønsker å lagre i grupper. Da vil hver gruppe bli til en tabell og hvert element til et felt. Deretter kan man ta utgangspunkt i det forslaget man har og følge normaliseringsreglene:

1. normalform: Fjerne grupper med repeterende data og flytte dette til en ny tabell med en egen primærnøkkel.
2. normalform: 1. normalform må være oppfylt
Hver kolonne må avhenge av hele primærnøkkelen og ikke kun den ene dersom man har en delt primærnøkkel. Det vil si at primærnøkkelen består av mer enn ett felt.

3. normalform: 2. normalform må være oppfylt
Ingen felter skal ha noen sammenheng med andre felter med unntak av primærnøkkelen.

4.2 Innføring i SQL (Structured Query Language)

SQL er et databaseprogrammeringsspråk. Det skiller seg fra andre programmeringsspråk som C++ og Visual Basic ved at det består av relativt få «statements», men denne syntaksen kan til gjengjeld være meget kompleks. SQL er opprinnelig et spørrespråk, men er også et komplett språk som kan brukes for å programmere eller definere en relasjonsdatabase.

For å opprette, vedlikeholde, endre og legge inn data i en database finnes det en rekke ulike applikasjoner som kan benyttes. Noen av de mest kjente er Microsoft SQL Server og Microsoft Access. I disse applikasjonene har man tilgang til et grafisk grensesnitt som gjør det mulig å arbeide med databasen uten å kunne SQL. Det er likevel en fordel å kunne SQL dersom man skal arbeide med databaser.

Det finnes flere ulike dialekter av SQL. De ulike databaseplattformene benytter forskjellige SQL-dialekter. I dette prosjektet er det jobbet mot MS SQL Serverdatabaser og denne databasetypen støtter dialekten T-SQL (Transact SQL).

SQL-språket består av to komponenter, DML og DDL.

1) DML - Data Manipulation Language
   Denne delen består av de vanligste SQL-kommandoene som SELECT, INSERT, UPPDATE og DELETE. Disse uttrykkene brukes for å gjenfinne data, legge inn data, oppdatere eller slette data i en database.
2) DDL - Data Definition Language
   Denne delen består av de uttrykkene man bruker for å opprette objekter i databasen og for å sette egenskaper og attributtverdier på selve databasen. CREATE, ALTER og DROP er eksempler på slike uttrykk.

Oppbygningen av et SQL-statement likner måten man skriver engelske setninger på. Hvert uttrykk starter med et SQL-nøkkelord som blir etterfulgt av argumenter

Eksempel på oppbygningen av et SELECT-statement i SQL:

```
SELECT [<tabellnavn>].<feltnavn>, [<tabellnavn>].<feltnavn>
FROM <tabellnavn>, <tabellnavn>
[WHERE <betingelse>]
[ORDER BY <uttrykk>]
```

Etter SELECT er det kun nødvendig å ta med feltets tabellnavn dersom søket skal returnere data fra flere tabeller. Feltnavnet må alltid spesifiseres så sant man ikke ønsker at informasjonen i alle feltene i databasen skal returneres. Da benytter man * som wildcard.

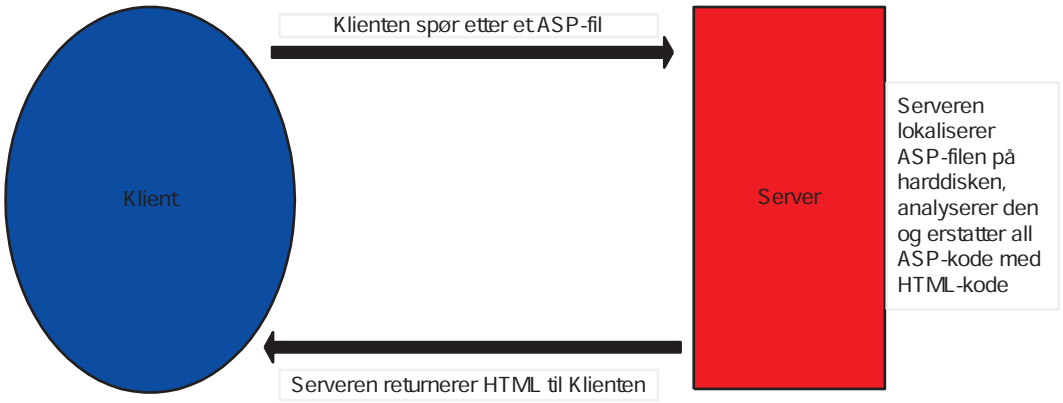
Etter FROM spesifiserer man fra hvilke tabeller man ønsker å hente ut data. Dette må alltid være med i et SELECT-statement

Etter WHERE kan man legge inn en betingelse som fører til at man ikke får returnert så mange rader fra databasen, men kun de man er ute etter. Dette er derfor en fin måte å begrense informasjonsmengden som returneres dersom databasen inneholder store mengder med data.

Etter ORDER BY kan man legge inn et uttrykk, slik at man får sortert informasjonen man ønsker å hente ut fra databasen enten alfabetisk eller etter tallstørrelse.

4.3 Innføring i ASP (Active Server Pages)

Microsoft introduserte Active Server Pages i desember 1996. Det er en åpen standard som kan kombineres med flere andre programmeringsspråk. Dette gjør det enkelt å integrere databaser og tyngre programmer med en nettside. I kjernen av ASP benyttes VBScript som standard skriptspråk, men programmeringsspråk som Perl, C++, Java kan benyttes for å lage egne komponenter. All ASP-kode blir kjørt på serversiden, se figur 4.3.



Figur 4.3 Prinsippet for behandlig at et ASP-skript

Når en nettside, som består av både HTML- og ASP-kode, blir kalt vil den delen av koden som står inne i ASP-modus (se eksempel nedenfor) bli kjørt på serveren og omformet til HTML-kode. Deretter blir koden sendt til klienten som spurte etter det bestemte ASP-dokumentet. Dermed inneholder siden bare HTML-kode når nettleseren mottar den. Dette fører til at ASP-koden er nettleseruavhengig.
Eksempel på kode hvor man skifter mellom HTML-modus og ASP-modus:

```
<%@ Language = VBScript %> 'Sier hvilket skriptspråk

<html>
<title>Eksempel</title>
<body>
Nå er jeg inne i HTML-modus
<% Response.write «Nå er jeg inne i ASP-modus»%>
</body>
</html>
```

Her kan man se at det brukes mindre-enn-tegn og prosenttegn når det skal skiftes til ASP-modus. For å komme tilbake til HTML-modus brukes prosenttegn og større-enn-tegn. Dette gjør det enkelt å stadig bytte mellom disse modusene og gjør koden meget fleksibel.

Eksempler på hva ASP kan brukes til:
- Nettsider basert på databasesøk
- Personifiserte nettsteder
- Nettbutikker
- Behandling av skjemaer på nettet
- Passordbeskyttelse
- Spill

For å arbeide med ASP trenger man enten en Internet Information Server (IIS) eller en Private Web Server. Dette er Microsoft produkter. Dersom man vil kjøre ASP under Netscape Enterprise Server, Lotus Notes Domino og noen UNIX-varianter må man benytte seg av Chilisoft sine produkter.

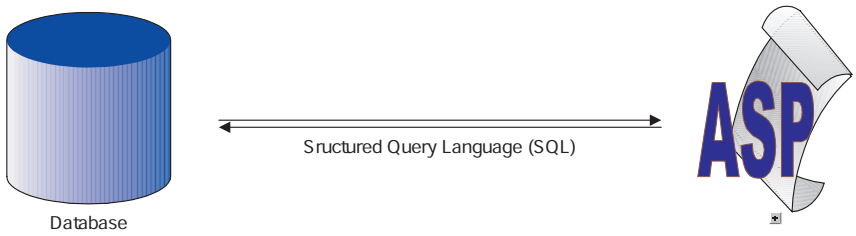
I ASP finnes det en del innebygde komponenter som forenkler programmeringen:
- **Response-objektet:** Tar seg av alt med sending av tekst, data og cookies til nettleseren, og kontrollerer hvert enkelt steg i overføringen av siden.
- **Server-objektet:** Har metoder for å opprette serverkomponenter. Komponentene er en samling av relaterte objekter som man kan bruke på sine sider. Det ligger også konverteringsfunksjoner og overordnet skriptkontroll i dette objektet.
- **Request-objektet:** Tar seg av det som har med lesing av data fra klienten å gjøre. Slik som innhenting av informasjon fra skjemaer og cookies.
- **Session-objektet:** Tar vare på verdiene til en klient gjennom hele sesjonen som kan være på flere sider.
- **ASPError-objektet:** Dette er nytt i IIS 5.0 og lar deg samle opp informasjon om feil som har oppstått i skriptet.

I VBScript har man samme type kontrollstrukturer som i de fleste andre programmeringsspråk; slik som if-setninger, select-setninger, for-løkker, do-løkker, while-løkker og lignende. Syntaksen forøvrig er veldig lik Visual Basic, og er lett å lære.

ASP kombineres gjerne med databaser. Ett hvert ASP-skript som skal kobles opp mot en database må først åpne den aktuelle databasen på serveren. Det er to måter å gjøre dette på:
1) Ved hjelp av en Data Source Name (DSN): En DSN-forbindelse krever at den som administrerer serveren setter opp en DSN på serveren ved å bruke kontrollpanelet. Det er også mulig å benytte en tredjepartskomponent som lar ASP-skriptet gjøre endringer i registeret på serveren. Dette gjør det mulig å lage egne DSN-oppsett etter hvert som man trenger de. En DSN-forbindelse krever vanligvis et minimum av DSN-navn, et brukernavn og passord.
2) Uten DSN: Da må man kjenne navnet på databasefilen (for filbaserte databaser som Access, Paradox og lignende) eller navnet på datakilden dersom man har med en SQL Server-database å gjøre. I tillegg må man vite hvilke drivere man må benytte og hvor på serveren databasen ligger.

Etter at oppkoblingen er i orden kan man begynne å kommunisere med databasen. Da benytter man som regel Structured Query Language (SQL). Ved hjelp av dette språket er det enkelt og legge inn, hente ut, oppdatere og slette informasjon i databasen. SQL-statement legges inn i ASP-dokumentet sammen med VBScript og ofte litt HTML for å formatere nettsiden.

Resultatet fra en SQL-spørring blir lagret i et ASP-objekt som heter «recordset». Dette gjør det enkelt å benytte dataene som er hentet fra databasen. Et recordset kan inneholde en eller flere rader fra en database avhengig av hvor mange rader som svarer til søket man utfører.



Figur 4.4 Kommunikasjonen mellom en database og ASP foregår ved hjelp av SQL

4.4 Innføring i Macromedia Flash

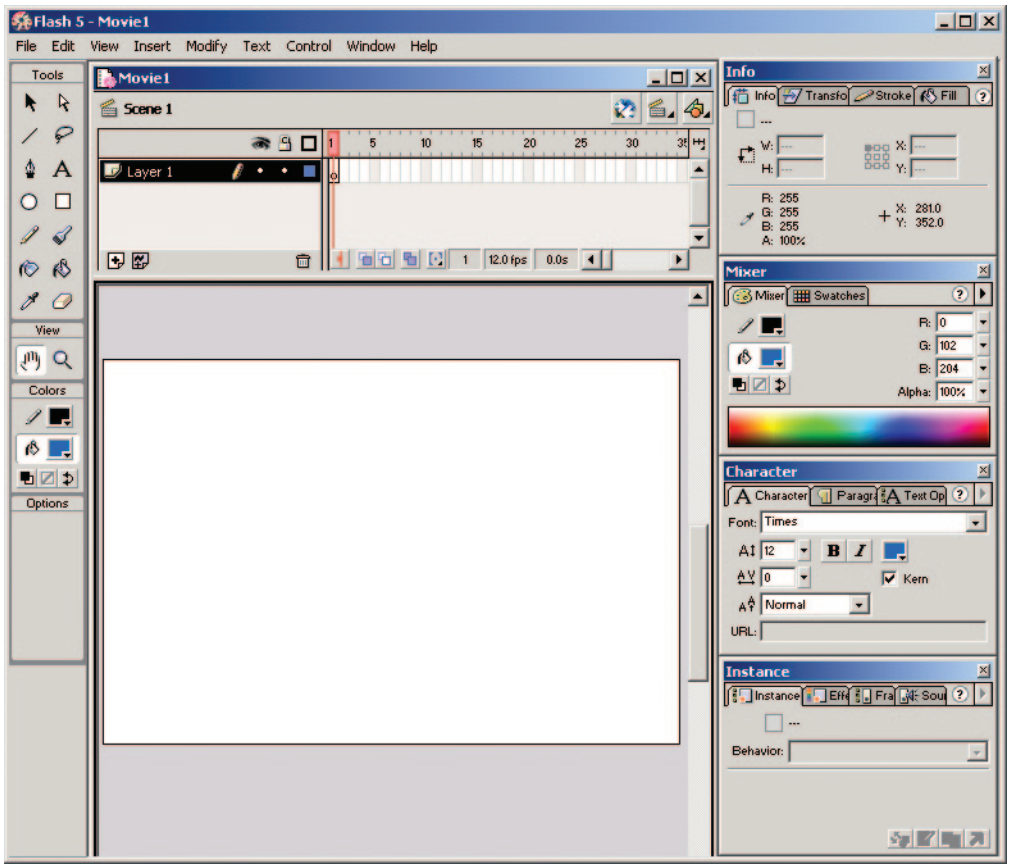
Macromedia Flash er et verktøy for å utvikle nettsider med animasjoner og vektorgrafikk, og har etter hvert blitt en de facto standard. Flash er praktisk talt nettleseruavhengig. Alt som trengs er en programutvidelse (plug-in) til nettleseren. De aller fleste nyere nettlesere har denne programutvidelsen, men den kan også lastes ned gratis fra Macromedias nettsider.

Det har lenge vært et stort behov for en standard for vektorgrafikk som kan benyttes på internett. Flash bygger på vektorgrafikk, det vil si at linjer beregnes med utgangspunkt i koordinater og kurver - såkalte vektorer. Datamaskinen regner om disse koordinatene og vektorene når elementer forstørres eller forminskes. Dette gjør at kvaliteten på grafikkelementet blir like bra uansett hvor mye det skaleres. Filstørrelsen på vektorbilder er veldig liten sammenlignet med bitmapbilder og nedlastningstiden på vektorbilder er betraktelig kortere enn for tilsvarende bitmapbilder. Flash egner seg bra til publisering av animasjoner på internett da filstørrelsen bare utgjør en brøkdel av det som er vanlig for gifanimasjoner.

Selv om Flash i utgangspunktet er et program som baserer seg på vektorer er det mulig å inkludere andre filtyper som for eksempel jpg, gif, mov eller mp3.

4.4.1 Grensesnittet i Macromedia Flash

Flash er et komplekst verktøy med mange muligheter for brukeren. Brukergrensesnittet er delt opp i flere ulike områder som har forskjellige egenskaper, se figur 4.5. Hovedgrensesnittet består av en verktøyboks, ei tidslinje, en scene samt flere ulike paneler.



Figur 4.5 Brukergrensesnittet i Flash 5

Verktøyboksen inneholder ulike verktøy som brukes til designet av Flashdokumentet, som for eksempel penneverktøy–, maleboks og forstørrelsesglass.

Flash er et tidsbasert program. En tidslinje brukes til å styre hvilke hendelser som skal inntreffe på de ulike tidspunktene i Flashfilmen. Et Flashdokument kan bestå av en eller flere tidslinjer som igjen består av flere «rammer», eller «frames» som det heter på engelsk. For å få en bedre oversikt kan man velge å bygge opp ei tidslinje med flere lag med logiske grupperinger. Disse lagene fungerer omtrent på samme måte som i Freehand eller Photoshop. Lagene kan legges til og trekkes fra, legges oppå hverandre, skjules, låses eller man kan stokke om på rekkefølgen av dem. En forståelse for hvordan lag fungerer er essensielt for utviklingen av større dokumenter. Tidslinja styres av en markør som går fra frame til frame i et tempo som brukeren bestemmer. Det vanlige er 12 fps (frames per second). Hver frame kan inneholde ulik informasjon, eller flere kan inneholde samme informasjon. Dette er det opp til brukeren å bestemme.

Scenen er det området som vil vises i Flashspilleren eller i nettleseren. Her plasseres grafikkelementene som bygger opp dokumentet. Man kan sammenligne scenen med en teaterscene. Ulike scener inneholder ulike elementer; scenen fra parken inneholder busker, blomster, benker og kanskje en fontene, mens scenen fra stuen inneholder ulike møbler. Teateranalogien kan være et hjelpemiddel til å forstå hvordan et Flashdokument bygges opp med ulike scener til ulike tidspunkt.

Den siste hoveddelen av brukergrensesnittet er de ulike panelene,. Disse består av skillekort som er delt inn i logiske grupper med elementer som ofte brukes i sammenheng. For eksempel består panelet «Characters» av skillekort for redigering av tekst og avsnitt, samt et skillekort for å sette teksten til dynamisk, statisk eller «input» (det vil si at brukeren av et nettsted kan skrive inn tekst i en tekstboks hvis «input» er valgt).

4.4.2 Innføring i ActionScript

ActionScript er Flash sitt eget skriptspråk. ActionScript brukes til å kontrollere objekter og utvider Flash slik at det er mulig å lage interaktive Flashfilmer og applikasjoner til internett.

ActionScript er et objektorientert skriptspråk. Det vil si at actions kontrollerer objekter når en spesiell handling inntreffer, som for eksempel et museklikk, at tidslinjen når en spesiell frame eller lignende.

I objektorientert programmering er objekter organisert i klasser. Hver klasse har et sett av egenskaper (properties) og metoder (methods) knyttet til seg. I dette prosjektet er følgende klasser brukt:

- **MovieClip-klassen:**Hver tidslinje i et Flashdokument er en instans av denne klassen.
- **Date-klassen:** Denne klassen brukes i alle sammenhenger der tidsregninger brukes.
- **Boolean-klassen:** Klassen brukes når det skal testes om en påstand er sann eller ikke.

Et ActionScript kan knyttes til frames, movie-clips, tekstbokser, knapper eller grafikkelementer.

Flashfilmer kan sende informasjon til og hente informasjon fra andre type filer som for eksempel serversideskript, tekstfiler og XML-filer. Serversideskript kan spørre etter spesifikk informasjon i en database og sende denne informasjonen til Flash. Serversideskript kan skrives i flere ulike språk; noen av de mest vanlige er ASP og PHP. Ved å lagre informasjonen i en database og benytte seg av et serversidescript kan man skape dynamiske og personifiserte Flashfilmer.

Det finnes flere metoder for å sende informasjon til og fra en Flashfilm. For å sende eller hente opp variabler brukes loadVariables eller getURL. For å laste inn en ekstern Flashfilm brukes loadMovie.

5 Utstyr benyttet i prosjektet

5.1 Maskinvare

Prosjektgruppa har i hovedsak benyttet tre bærbare PC-er av typen Dell under gjennomføringen av prosjektet.

Spesifikasjonene på disse er:
- Dell Latitude CPt
- Celeron 600MHz
- 128MB RAM
- 6 GB Harddisk
- 14,1» TFT Skjerm

De serverne som er benyttet er av typen Compaq.
Spesifikasjonene på disse er:
- Compaq ProLiant ML350
- Pentium III 600MHz
- 512 MB RAM
- 3´18 GB Harddisk

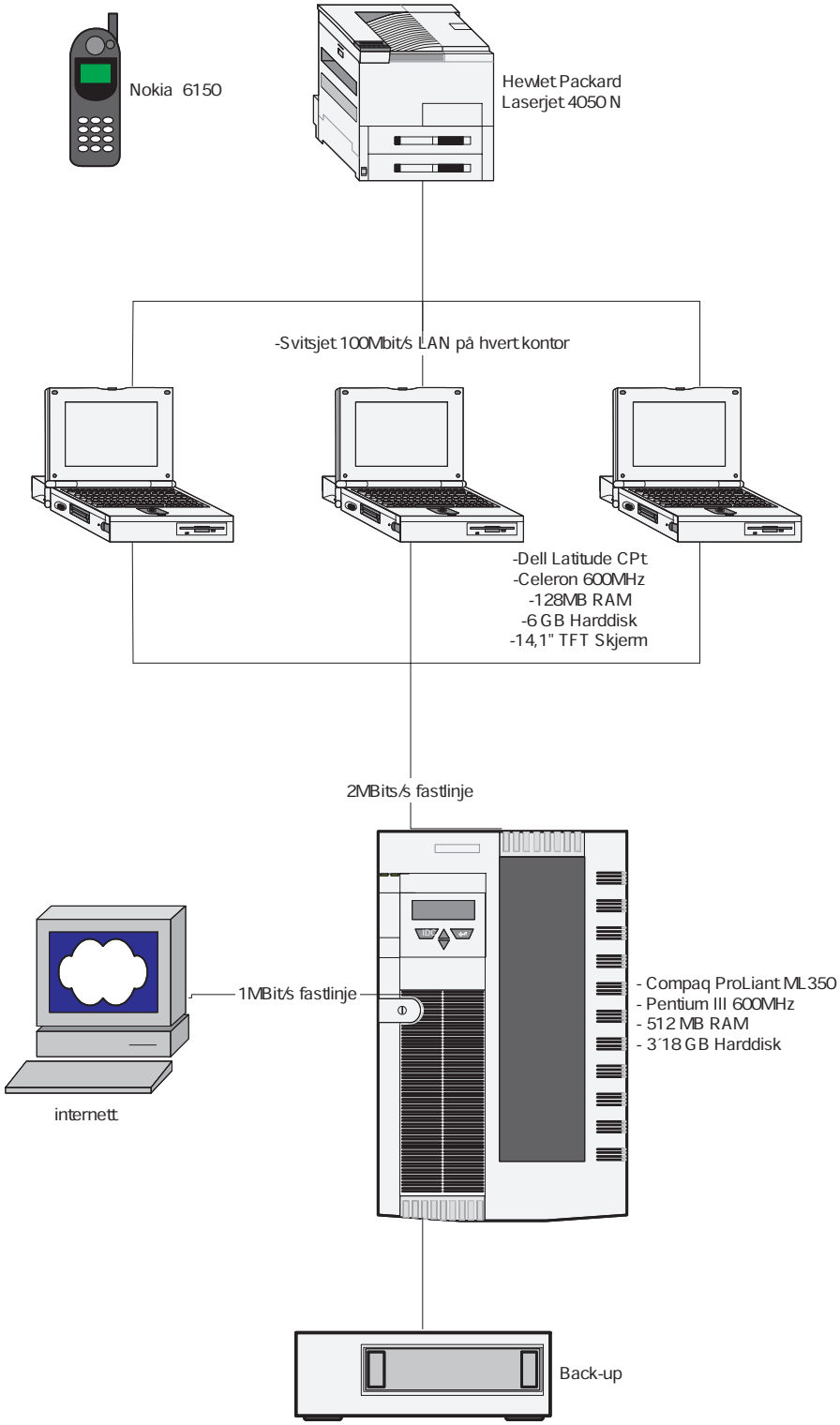
Studentene har sittet tilkoblet Framfabs interne nettverk, se figur 5.1.
Spesifikasjonene på dette er:
- Svitsjet 100Mbit/s LAN på hvert kontor
- 2 MBit/s fastlinje mellom kontoret i Oslo og kontoret i Halden som er knutepunktet
- 1 MBit/s fastlinje for tilgang til internett fra Halden. Denne linja dekker også tilgangen til internett fra kontoret i Oslo

5.2 Programvare

Her følger en liste over programvare som har blitt benyttet under prosjektet:
- Windows 2000 NT
- Microsoft Office 2000
- Microsoft Internet Explorer 5.0
- Microsoft SQL Server 2000
- Microsoft Visio 2000
- Microsoft Project 2000
- Microsoft Outlook 2000
- Adobe Illustrator 7
- Adobe InDesign 1
- Adobe Photoshop 5,5
- Macromedia Flash 5
- UltraEdit

5.3 Utviklingsarkitektur

Figuren under inneholder en oversikt over utviklingsmiljøet som er benyttet:



Figur 5.1 Utviklingsarkitektur

6 Kravspesifikasjon for systemet

Det ble laget et førsteutkast til kravspesifikasjonen på bakgrunn av forarbeidet som ble gjort i forbindelse med det tekniske designet av systemet. Her ble det forklart hva systemet skal kunne gjøre og hvordan det fungerer med vekt på de tekniske detaljene. Dette utkastet ble sendt til veilederne for tilbakemelding, og responsen var at de funksjonelle kravene var godt spesifisert. Derimot manglet en del informasjon rundt selve prosjektet og en del ikke-tekniske aspekter. Gruppa tok dette til etterretning og utdypet kravspesifikasjonen med mer informasjon om pålitelighet, anvendbarhet, ytelse og kompatibilitet. Det ble også lagt inn skjermdumper for å illustrere oppbyggingen av systemet.

Ideelt burde kravspesifikasjonen vært klar og godkjent av både utviklere og oppdragsgiver før implementeringen ble satt i gang. Dette var ikke tilfellet i dette prosjektet. De tekniske detaljene var klare tidlig og studentene fikk tilbakemelding om at de var greie, men det ble ikke foretatt noen formell godkjenning. Dermed startet utviklingen av systemet før resten av kravspesifikasjonen var klar, noe som kan være uheldig. Dette bør helst ikke skje i et reelt prosjekt hvor man må ta med økonomiske betraktninger i beslutningen.
Under følger kravspesifikasjonen i sin helhet:

6.1 Krav til administrasjonsverktøyet

6.1.1 Administrasjonsverktøyets navigasjonsstruktur

Figur 6.1. inneholder en oversikt over navigeringsstrukturen og funksjonaliteten til administrasjonsverktøyet. Vedlegg B inneholder en oversikt over alle filene som inngår i administrasjonsverktøyet samt en kort forklaring om hvliken funksjon de har

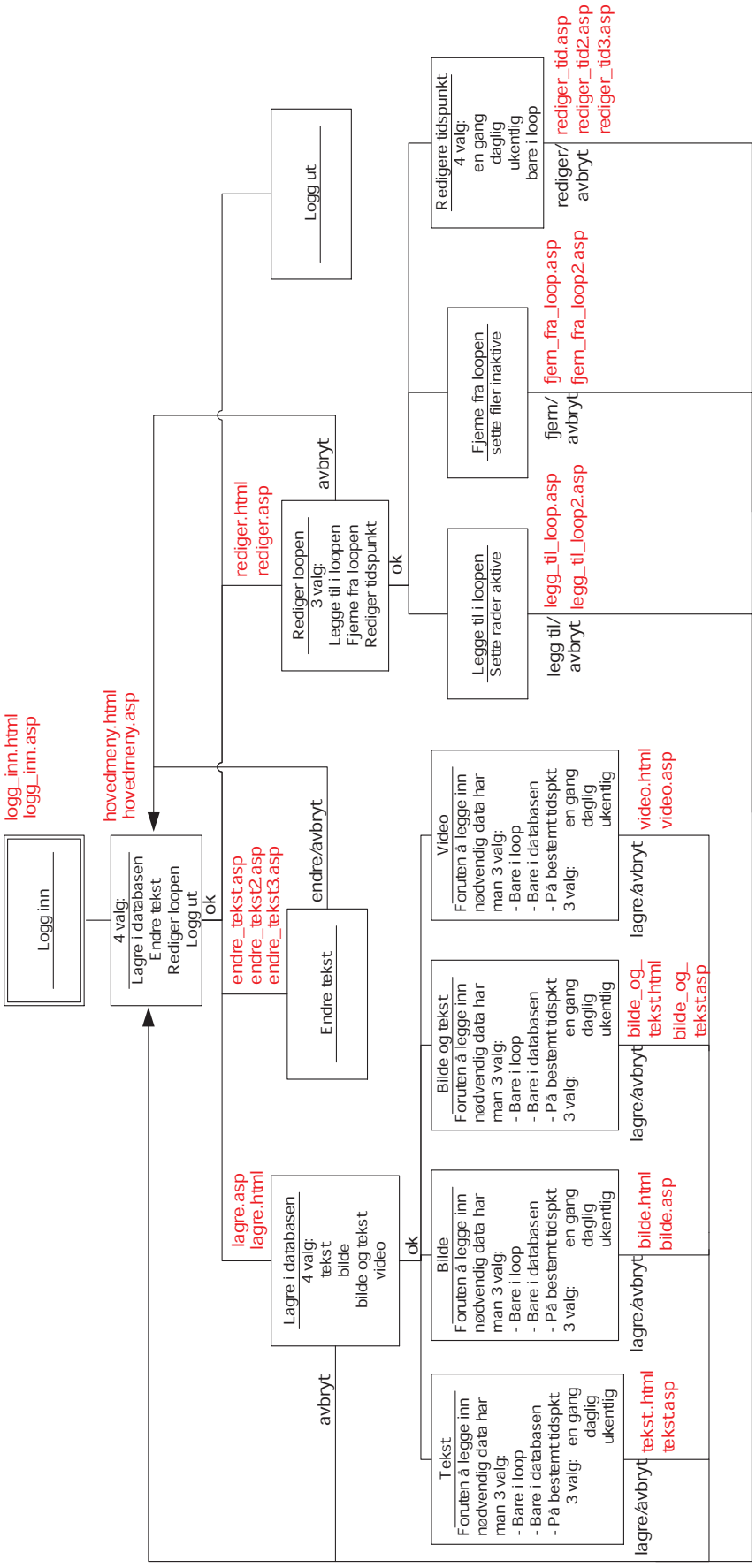
6.1.2 Pålogging

Administrasjonsverktøyet skal spørre etter brukernavn og passord for å kunne kontrollere hvem som har tilgang til administrasjonsverktøyet. Systemet har kun ett brukernavn og passord. Dette oppgis i brukerveiledningen, vedlegg A.

6.1.3 Lagring i databasen

Administrasjonsverktøyet skal kunne:

- lagre bilde, video og tekst i databasen, eventuelt med tilhørende lyd. Teksten skal plasseres direkte i databasen mens det kun skal legges referanser til lyd-, bilde- og videofilene i databasen. Samtidig skal filene bli lagret i riktig mappe på serveren.
- legge til en tekst i databasen som skal vises samtidig med et bilde
- endre overskrifter og tekster som ligger i databasen



Figur 6.1 Navigasjonsstruktur for administrasjonsverktøyet

6.1.4 Styring av elementer

Administrasjonsverktøyet skal kunne administrere hva som skal vises i loopen og hva som ikke skal vises i loopen (aktiv/inaktiv i databasen).

6.1.5 Tidsstyring av elementer

Med administrasjonsverktøyet skal det være mulig å bestemme

- tidspunkt for visning av det som ligger i databasen (dato og klokkeslett). Den informasjonen som har et eksakt tidspunkt koblet til seg vil avbryte loopen.
- at informasjon som ligger i databasen skal vises jevnlig (daglig, eller ukentlig). Denne informasjonen vil også avbryte loopen.
- hvor lenge ting som legges inn på bestemt tidspunkt skal vises i loopen

6.1.6 Administrasjonsverktøyets forutsetninger

- Det forutsettes at det kun er en person som bruker administrasjonsverktøyet av gangen.
- Filtyper som systemet skal kunne legge inn i databasen: jpg, gif, swf, mov, mpg, wav og mp3.

6.2 Krav til Flashdokumentet

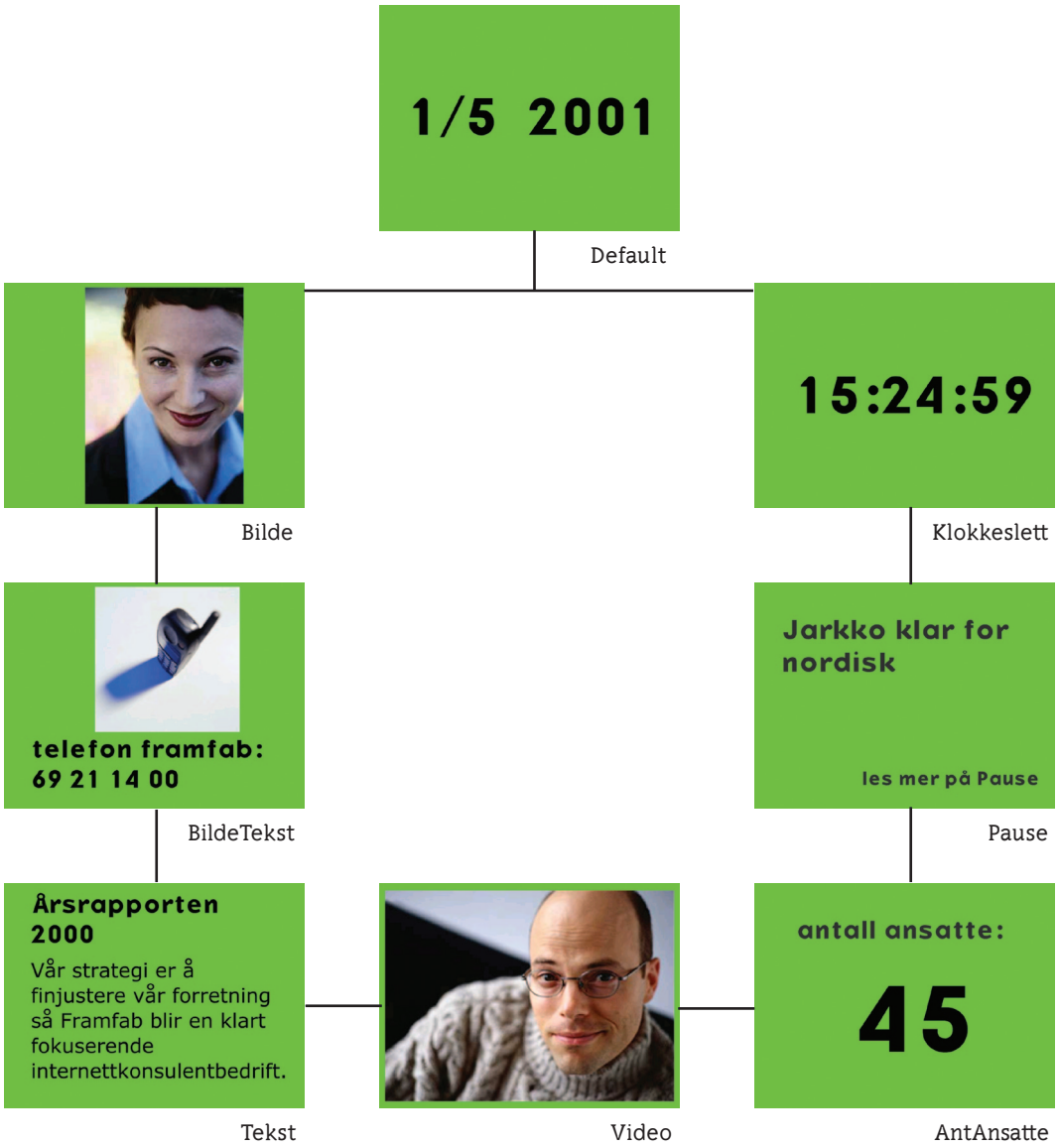
6.2.1 Funksjonell Flashspesifikasjon

Informasjonskanalen skal gå i loop etter et fast mønster som vist på figur 6.2. Mellom klokka 17:00 og 09:00 skal informasjonskanalen stå på skjermbilde «Default». Mellom klokka 09:00 og 17:00 skal informasjonskanalen vise resten av skjermbildene i følgende rekkefølge: Bilde – BildeTekst – Tekst – Video – AntAnsatte – Pause – Klokke. Når alle skjermbildene er vist en gang starter loopen på nytt på Pause. Slik går loopen helt til klokken 17:00.

Loopen kan imidlertid avbrytes. Dette kan skje ved at administratoren for systemet går inn i informasjonskanalens administrasjonsverktøy og definerer tidspunktet informasjonen skal vises. Dersom noe skal vises på et gitt tidspunkt vil loopen avbrytes. Systemet vil finne ut hva slags informasjon som skal vises og gå til det respektive skjermbildet. Etter at denne informasjonen er vist fortsetter loopen automatisk til det etterfølgende skjermbildet.

Eksempel:
Hvis administrator ønsker å vise teksten «Ekstraordinært allmøte kl. 13:00» på informasjonskanalen fra klokka 12:00 til 13:00, må han koble dette tidspunktet med beskjeden i databasen. Dette gjøres enklest ved hjelp av administrasjonsverktøyet, men kan også gjøres direkte i databasen. Da vil loopen hoppe til skjermbilde «Tekst» klokka 12:00 uavhengig av hvilket skjermbilde som ble vist i forkant av dette. Klokka 13:00 vil loopen fortsette på det etterfølgende skjermbildet, altså «Video», og loope videre derifra.

Det er viktig å merke seg at visningstida for det elementet som blir vist fram til klokka 12:00 blir redusert i forhold til det som er spesifisert i databasen. Dette er spesielt uheldig dersom det vises en video som ikke er rekker å bli ferdig. Dersom loopen akkurat hopper videre til et nytt skjermbilde rett før noe skal vise på bestemt tid vil man bare få et raskt glimt av et skjermbilde som man ikke rekker å lese før skjermbildet som skal vises på et bestemt tidspunkt dukker opp. Dette vil stort sett ikke være noe problem i og med at hvert skjermbilde skal vises med en standardtid på fem minutter. Dette fører til at det vil være nokså sjelden man ikke har rukket å se hva skjermbildet viser før det blir avbrutt.



Figur 6.2 Skjermbilder i informasjonskanalens loop

Forklaring av de ulike skjermbildene:

- **Default:** Viser dagens dato**
- **Bilde:** Viser et bilde*
- **BildeTekst:** Viser et bilde og en tekst*
- **Tekst:** Viser en overskrift og en tekst*
- **Video:** Viser en videosekvens
- **AntAnsatte:** Viser et tall som angir antall ansatte i Framfab samt en ledetekst**
- **Pause:** Viser en overskrift fra internavisa «Pause» samt en ledetekst**
- **Klokke:** Viser klokkeslett**

* Disse skjermbildene skal kunne spille lyd dersom det er koblet en lyd til den informasjonen som skal vises.

** Disse skjermbildene kan ikke vises på et spesielt tidspunkt

6.2.2 Teknisk Flashspesifikasjon

Innholdet skal presenteres dynamisk, det vil si at innholdet ikke legges inn i selve Flashfila, men hentes opp fra databaser. Flashfila skal kalle opp ulike ASP-dokumenter som vanligvis skal returnere det Flashfila ønsker så sant det ikke finnes noe i databasen som skal vises akkurat da. I disse tilfellene vil nødvendig informasjon til det elementer som skal vises på bestemt tidspunkt bli returnert til Flashfila. På grunnlag av hva ASP-dokumentene returnerer skal Flashfila vise den returnerte informasjon etter et bestemt mønster. Informasjonskanalen skal kunne presentere video, bilde og tekst med eventuell tilhørende lyd forutsatt at det ligger informasjon av slik type i databasene som systemet henter informasjon fra.

Visningen skal styres av ei Flashfil med ulike «labels» (markører i Flashfilmen) som filmen kan hoppe mellom ettersom hvilken filkategoriid den mottar fra ASP-dokumentene.

6.2.3 Forutsetninger for Flashspesifikasjonene

Det forutsettes at systemet kjøres gjennom nettleseren Internet Explorer 5.0. med Flash 5 plug-in og QuickTime 4 plug-in.

6.3 Krav til ASP-dokumentene

6.3.1 Teknisk ASP-spesifikasjon

ASP benyttes som mellomledd mellom databasen og Flashfila. ASP-dokumentene skal, på forespørsel fra Flashfila, hente opp informasjon fra Studentdatabasen og Framfabs Kompetanse- og Pause-database. Den informasjonen som blir hentet opp fra databasene returneres så på en forståelig form tilbake til Flashfila.

Alle ASP-dokumentene skal først sjekke om det er noe som skal vises på tid akkurat nå. Denne koden ligger i et eget dokument som blir inkludert i alle de andre ASP-dokumentene. Hvis det ikke er noe informasjon som skal vises på tid på det aktuelle tidspunktet, returnerer ASP-dokumentene det de blir spurt om (det som ligger for tur i loopen i Flashfila).

De ulike ASP-dokumentne returner Følgende informasjon:

Dokumenter som henter informasjon fra Studentdatabasen:
- **bilde.asp:** Returnerer filkategoriid = 1, lagringsstedet til bildet, varigheten og eventuelt lagringsstedet til en lydfil.
- **tekst.asp:** Returnerer filkategoriid = 5, overskrift, tekst, varigheten og eventuelt lagringsstedet til en lydfil.
- **bilde_tekst.asp:** Returnerer filkategoriid = 2, lagringsstedet til bildet, varigheten, tekst og eventuelt lagringsstedet til en lydfil.
- **video.asp:** Returnerer filkategoriid = 4, lagringsstedet til videoen og varigheten.
- **klokke.asp:** Returnerer filkategoriid = 9 og varigheten. Dette dokumentet skal bare sjekke om det er noe som skal vises på tid de neste fem minuttene. Hvis ikke skal det bare vises en klokke i Flashfila.

Dokumenter som henter informasjon fra Framfabs databaser:
- **ant_ansatte.asp:** Returnerer filkategoriid = 7, antall ansatte og varigheten.
- **pause.asp:** Returnerer filkategori = 8, den nyeste overskriften i Pause og varigheten

6.3.2 Forutsetninger for ASP-spesifikasjonene

- Den informasjonen som hentes fra Framfabs databaser kan ikke vises på et gitt tidspunkt eller kombineres med lyd.
- Det er en forutsetning at klokka på serveren og klokka på klienten er synkronisert og at det faktisk ligger informasjon i databasen som svarer til søkene ASP-dokumentene foretar.

6.4. Krav til Studentprosjektdatabasen

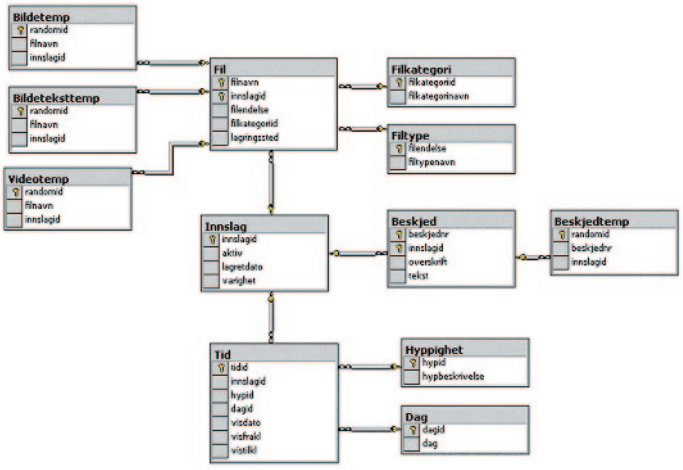
6.4.1 Teknisk databasespesifikasjon

Studentprosjektdatabasen skal
- inneholde tekstbeskjeder av en viss størrelse (antall karakterer er bestemt) samt en tilhørende overskrift. Dette lagres i Beskjed-tabellen
- inneholde lagringsstedet til bilde-, lyd- og videofiler som skal vises samt informasjon om hvilken kategori fila tilhører (eks bilde, video, lyd eller bilde kombinert med tekst). Dette lagres i Fil-tabellen
- inneholde beskjed om en post i databasen skal inngå i loopen eller ikke. Det vil si om informasjonen skal kunne plukkes ut av en randomfunksjon eller ikke. I tillegg skal databasen inneholde informasjon om hvor lenge et

element skal vises og når det er lagret i databasen. Alle disse opplysningene lagres i Innslag-tabellen.
- inneholde opplysninger om når et element skal vises, både dersom den kun skal vises en gang eller den skal vises flere ganger. Dette lagres i Tid-tabellen. Det vil kun ligge informasjon i tidtabellen dersom det ligger elementer i databasen som skal vises på forhåndsbestemte tidspunkt.
- holde styr på hva som er vist og hva som ikke er vist i loopen. Dette gjøres ved at det opprettes egne tabeller (temp-tabeller) for henholdsvis bilde, video, tekst og bilder som skal kombineres med tekst. Disse tabellene skal til enhver tid kun inneholde informasjon om de radene i databasen som ikke er vist den siste tida. Når for eksempel et bilde er vist i loopen vil dette slettes fra Bildetemp-tabellen. På denne måten sikrer man at ikke det samme bildet vises flere ganger etter hverandre. Når temptabellene er tomme må det genereres nye ut fra innholdet i Fil- , Beskjed- og Innslag-tabellen.

6.4.2 Databasestruktur

Databasen Studentprosjekt har følgende struktur:



Figur 6.2 Strukturen over Studentprosjektdatabasen med tabeller, felt og relasjoner

6.4.3 Forutsetninger for databasespesifikasjonene

- Det forutsettes at det til enhver tid ligger minst et element av hver kategori i databasen som skal vises tilfeldig i loopen (har aktiv = 1). Dette innebærer at studentdatabasen må inneholde informasjon om minst et bilde, en video, et bilde som er kombinert med tekst og en beskjed. I tillegg må det finnes minst en ansatt i Kompetansedatabasen og en nyhet i Pausedatabasen.
- Det forutsettes at det ikke legges inn at to ulike elementer i databasen som skal vises på eksakt samme tid. Da vil ikke systemet vite hvilket element som skal hentes opp. Det er ikke lagt inn begrensninger for dette i systemet, så det er administratoren av systemet som må holde styr på dette.

6.5 Krav til systemet som helhet

6.5.1 Systemets levetid

Systemet er en prototype og skal fungere som et grunnlag for en eventuell videreutvikling. En omfattende rapport, samt en brukerveiledning (vedlegg A), vil leveres sammen med systemet. Rapporten skal blant annet inneholde tips til videreutvikling.

6.5.2 Systembegrensninger

- Bildene som skal vises gjennom prototypen må tilpasses systemet. Det vil si at prototypen ikke kan vise bilder med størrelse over 580×435 piksler. Skal bildet kombineres med tekst må bildet være mindre enn dette.
- Administratoren av systemet har ingen mulighet for å styre det visuelle utseendet som for eksempel font, fontstørrelse, fontfarge, bakgrunnsfarge osv. Dette er forhåndsinnstilt, og i henhold til Framfabs designmanual.
- Dersom administratoren skriver inn for mye tekst i forhold til det systemet takler vil ikke den overskytende tekstmengden bli vist på skjermen. Administratoren vil heller ikke få noen feilmelding om dette når han legger inn teksten.

6.6 Pålitelighet

- Prototypen vil ikke oppfylle alle spesifikasjoner dersom man kobler opp flere systemer mot den samme Studentdatabasen.
- Det vil ikke bli lagt inn noen begrensninger for at to eller flere personer kan benytte administrasjonsverktøyet samtidig.
- Systemet tar utgangspunkt i at brukeren følger brukerveiledningen nøye. Det vil ikke bli foretatt testinger på hva som skjer dersom brukeren legger inn ugyldig eller motstridende data i databasen.
- Prototypen vil bare bli testet på den TV-en som studentene har tilgjengelig i Oslo. Det kan derfor ikke garanteres for den visuelle presentasjonen på andre TV-apparater.

6.7 Anvendbarhet

- Feilhåndtering og tilbakemelding til brukeren vil ikke bli prioritert.
- Administrasjonsverktøyet er med på å gjøre systemet enklere i bruk. Man slipper da å gå inn i selve databasen og foreta endringer, og det forhindres at ugyldig eller motstridende data blir lagt inn i databasen.

6.8 Ytelse

- For å bedre ytelsen til databasen vil store filer som bilde, lyd og video ikke ligge i selve databasen, men i egne kataloger på serveren.
- For å oppnå maksimal ytelse anbefales det å benytte komprimering, som for internett, av bilde-, lyd- og videofiler.

6.9 Kompatibilitet

- Systemet vil kun bli tilpasset PC-er med Windows 2000 operativsystem. I tillegg kreves det at maskinen som skal kjøre systemet har Macromedia Flash 5 plug-in, QuickTime 4 plug-in samt Internet Explorer 5.0.
- Systemet vil kun bil testet opp mot databaser av typen MS SQLServer 2000
- Det visuelle skal kunne presenteres på en TV-skjerm.

7 Design

7.1 Teknisk design

7.1.1 Databasedesign

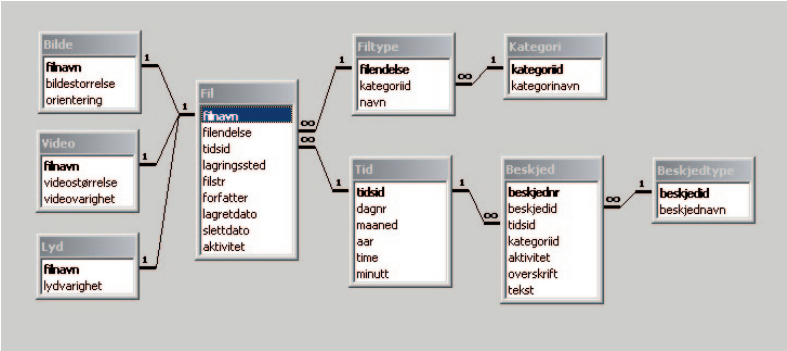
Databasen Studentprosjekt danner grunnlaget for hele systemet. Det ble derfor jobbet mye for å få databasestrukturen klar så tidlig som mulig i prosjektet. Det var en omfattende prosess for å få opprettet en struktur som skulle kunne håndtere alle kravene som ble stilt til systemet. Alt måtte tenkes gjennom i minste detalj for at man kunne være sikker på at databasen holdt mål. Dette var vanskelig i og med at det ble gjort så tidlig i prosjektet, men var noe studentene hadde igjen for senere. Arbeidet førte til at gruppa fikk klarhet i hvordan systemet måtte organiseres for å kunne oppnå ønsket resultat tidlig i prosjektet.

Det ble tegnet mange forslag til databasestruktur før gruppa kom fram til en endelig løsning. De første forslagene ble tegnet med penn og papir og ble diskutert på møter hvor alle gruppemedlemmene deltok. På den måten kom man nærmere og nærmere et resultat. Etter hvert ble Microsoft Word og senere Microsoft Visio benyttet for å tegne databasestrukturen. Dette var mer lettvint og gjorde det lettere å få innspill fra utenforstående personer.

Når det gjelder navngiving av tabeller og felt er det viktig å være konsekvent samtidig som man benytter seg av logiske navn. I dette prosjektet er det valgt å benytte den navngivingsstrukturen som oppdragsgiver allerede benytter i sine

databaser. Dette fordi prosjektets system skulle integreres med to av Framfabs databaser, Pause og Kompetanse. Navnet på selve databasen ble skrevet med stor forbokstav og resten med små bokstaver. Det samme gjelder alle tabellnavn. Feltnavn ble kun skrevet med små bokstaver.

Figur 7.1 inneholder et av de aller første forslagene til databasestruktur som ble laget. Dette var et nokså bra utgangspunkt, men det var en del som ble endret og forbedret etter hvert. Det var vanskelig å vite nøyaktig hva slags data det var nødvendig at databasen skulle inneholde og hva som var overflødig informasjon. I begynnelsen ble det derfor forsøkt å plassere flest mulig tenkelige opplysninger inn i databasen.



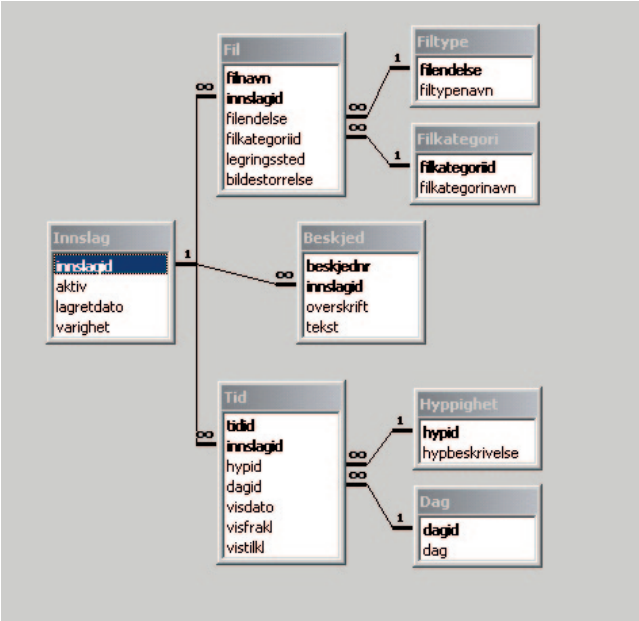
Figur 7.1 Ett av de første utkastene til databasestruktur

Det var naturlig å dele informasjonen som skulle vises i informasjonskanalen i to hoveddeler, filer og beskjeder. Derfor ble det opprettet to tabeller med disse navnene. I Filtabellen var det tenkt at man lagret informasjon som var felles for alle filer, uansett om det var snakk om video, bilder eller lyd. Fil-tabellen ble knyttet til tre subtabeller. I disse kunne man lagre informasjon som kun gjaldt filer av en bestemt type.

Både Beskjed- og Fil-tabellen måtte relateres med en tabell som inneholdt informasjon om når et element eventuelt skulle vises. Dersom man ikke hadde noe bestemt ønske om når et element skulle vises ville det ikke være nødvendig å legge inn informasjon i Tid-tabellen. På dette tidspunktet var det stor usikkerhet knyttet til hvordan servere, databaser og ASP håndterer tidsinformasjon. Derfor var det vanskelig å vite hvordan tidspunktet for når et element skulle vises burde lagres i databasen.

Tabellen Filtype, Filkategori og Beskjedtype ble opprettet med tanke på at det skulle være lettere å søke i databasen etter bestemte elementer.

Figur 7.2 inneholder databasestrukturen som forelå den dagen da gruppa hadde frist for at endelig databasestruktur skulle være ferdig (milepæl 2). På dette tidspunktet mente studentene også at de hadde klart å overholde denne milepælen og at den endelige databasestrukturen skulle se slik ut. Det viste seg likevel at det måtte noen endringer til for at det skulle være mulig å lage et system med ønsket funksjonalitet.



Figur 7.2 Databasestrukturen slik den forelå 1.mars

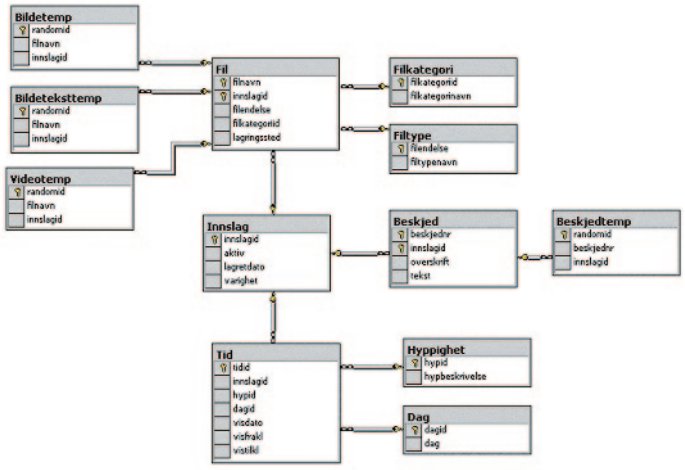
Det er vanlig at en database har en «inngang». Dette var ikke tilfelle på de tidligere utkastene. Derfor ble det opprettet en tabell som knytter alle tabellene sammen og dermed fungerer som en inngang for informasjonen som ligger i databasen. Denne tabellen ble kalt innslag. For hvert nye element som skal legges inn i databasen må det derfor opprettes en innslagid før mer nøyaktig informasjon om element kan legges inn.

Fra oppdragsgiver ble det anbefalt å kutte ut subtabellene Bilde, Video og Lyd. Det ble sagt at det ikke hadde så stor betydning om man hadde felt i Fil-tabellen som kun gjaldt den ene filkategorien. For de filene hvor denne parameteren var uvesentlig kunne man bare legge inn verdien NULL. Tabellen Filkatergori ble flyttet, slik at den fikk direkte kontakt med Fil-tabellen. Dette ville gjøre det enklere å foreta søk i databasen med hensyn på hva slags type fil man ville ha tak i.

Lagringen av tidspunkt i databasen er også en del forandret i forhold til det første forslaget. Dette fordi det hadde blitt gjort en del undersøkelser for å finne ut hvordan det var mest hensiktsmessig å lagre denne typen informasjon.

Det var et krav at informasjonen som skulle hentes ut fra Studentprosjektdatabasen skulle plukkes ut i tilfeldig rekkefølge samtidig som et element ikke skulle kunne hentes ut tilfeldig flere ganger etter hverandre. Det finnes ingen randomfunksjon i SQL som gjør dette mulig. For å løse problemet kom man fram til at det måtte opprettes egne tabeller som holdt styr på hva som skulle hentes ut og hva som allerede var hentet ut. Derfor ble tabellene Bildetemp, Bildeteksttemp, Beskjedtemp og Videotemp opprettet. Disse tabellene skal i utgangspunktet være tomme. Når systemet startes opp vil de automatisk bli fylt med alle elementene som skal hentes opp i tilfeldig rekkefølge og vises på TV-skjermen. Systemet vil derfor gå inn og hente et element fra en av disse tabellene avhengig av hva slags type informasjon systemet skal vise. Etter at elementet er hentet opp vil det bli slettet fra tabellen. På denne måten unngår

at det samme elementet hentes opp flere ganger etter hverandre. Når en temptabell er tom, vil den automatisk bli fylt med ny informasjon. Rekkefølgen elementene blir hentet opp fra temptabellene vil variere fra gang til gang fordi primærnøkkelen som hvert element får, vil være et randomtall som genereres for hver gang tabellene fylles opp. Figur 7.3 viser den endelige databasestrukturen som er benyttet i dette prosjektet.

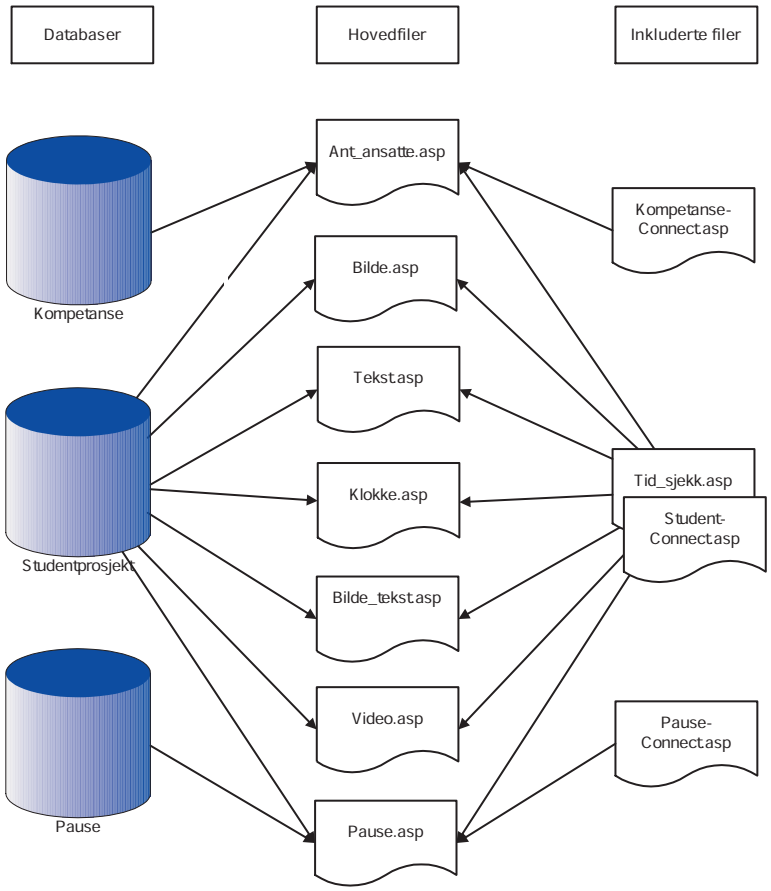


Figur 7.3 Den endelige databasestrukturen som en benyttet i prosjektet

7.1.2 Design av ASP-struktur

ASP-strukturen ble utarbeidet på bakgrunn av databasestrukturen og på grunnlag av hvilken måte Flash kunne ta imot informasjon. Det første som ble gjort var å sette opp pseudokode for et ASP-dokument som skulle hente informasjon fra databasen og sende dette til Flash, se vedlegg C. Deretter måtte pseudokoden utvides til å kunne sjekke om noe skulle vises på tid akkurat nå. For å ikke få alt for lang og innviklet kode ble det opprettet et dokument for hver type informasjon Flashpresentasjonen skulle vise. Koden ble effektivisert ved at den koden som var felles for flere dokumenter ble trukket ut i egne dokumenter som ble inkludert der hvor det var behov. På denne måten var det enkelt å gjøre forandringer og man unngikk å gjøre feil slik at den samme koden fungerte i et dokument, men ikke i et annet. Dette ASP-dokumentet kunne benyttes som en mal for de øvrige dokumentene.

Alle ASP-dokumentene er koblet til Studentprosjektdatabasen siden de må sjekke om det er noe som skal vises på tid før de kan sende den informasjonen det opprinnelig blir spurt om tilbake til Flash. De dokumentene som skal hente informasjon fra Framfabs databaser er derfor koblet opp mot to databaser. Dette fordi de må sjekke på tid først og dersom det ikke ligger inne noe på tid i Studentprosjektdatabasen, kan de gå inn i Framfabs database og hente informasjon derifra. Dokumentet Tidsjekk.asp inneholder den koden som sjekker om det er noe som skal vises på tid. Dette dokumentet er inkludert i alle hovedfiler sammen med koden som oppretter forbindelsen til Studentprosjektdatabasen, se figur 7.4.



Figur 7.4 Skjematisk oversikt over de ASP-filene som inngår i informasjonskanelen

7.1.3 Design av Flashstruktur

Visningen av elementene i informasjonskanalen styres av ei Flashfil. Denne fila skal hente opp variabler fra ASP og på grunnlag av variablenes verdi vise ulik informasjon. Det er med andre ord Flashfila som kontrollerer hele informasjonskanalen.

Det fins ulike måter å bygge opp en struktur for et Flashdokument. De vanligste struktrurformene er:

1) legge inn alle hendelser i ei og samme tidslinje
2) bruke en Flashfilm med flere tidslinjer
3) bruke flere Flashfilmer med en hovedfilm som kaller andre mindre filmer.

En struktur trenger ikke nødvendigvis være «bedre» enn en annen. Noen av de faktorene som kan spille inn på hvordan man bør organisere Flashinnholdet er: omtrent hvor stort og komplekst dokumentet vil komme til å bli, hvor mange som skal jobbe med innholdet samtidig, interaktivitet og animasjoner.

Før gruppa kom fram til den strukturen som ble valgt, ble det sett på ulike fordeler og ulemper ved de ulike struktureringsformene.

Om det skulle være mulig for flere å arbeide med utviklingen av Flashdokumentet samtidig ville det være best å bygge opp dokumentet med en hoved-

film som styrer og henter opp andre mindre filmer. For eksempel kunne en person tatt seg av oppbyggingen og struktureringen av hovedfilmen, mens en annen kunne stå for utviklingen av små-filmene som inneholdt funksjoner for visning av bilder, bilder og tekst, beskjeder, videoer og så videre. En slik oppbygning vil gi en total økning av filstørrelsen, noe som vil gå på bekostning av nedlastningshastigheten.

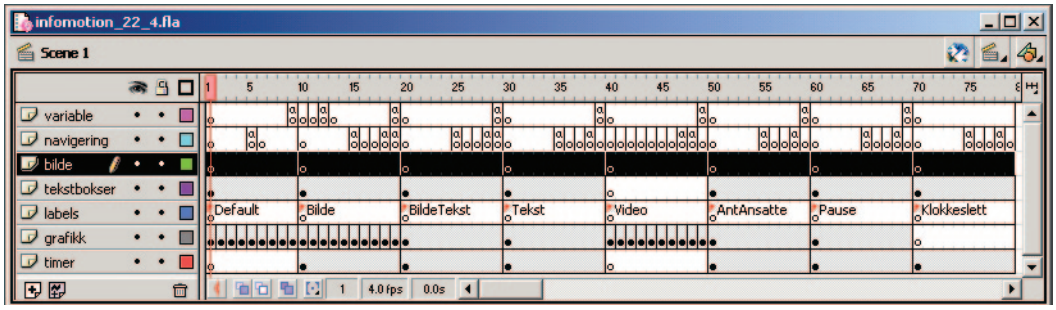
I dette prosjektet skulle ikke flere jobbe parallelt med ulike deler av Flashfilmen. Derfor ble det besluttet å benytte en struktur som baserer seg på en Flashfilm og ei tidslinje. Tidslinja ble i utgangspunktet delt opp i følgende åtte sekvenser: Default, Bilde, BildeTekst, Tekst, Video, AntAnsatte, Pause og Klokkeslett. Sekvensene Bilde, BildeTekst og Tekst skulle i tillegg ha funksjonalitet for å kunne spille en lydfil.

Hver sekvens skal på grunnlag av de variablene som hentes opp fra ASP kunne vise ulik informasjon. Det ble satt opp pseudokode for hvordan Flashdokumentet skulle spørre etter og behandle variabler fra ASP samt hvordan markøren skulle bevege seg ut fra disse variablenes verdier. Se vedlegg D.

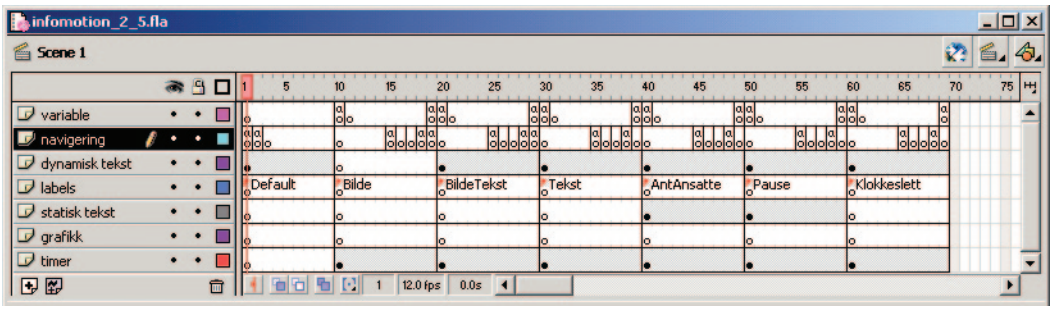
Det er de to variablene visTilKlokka og filkategoriid som styrer hva som skal vises til enhver tid. Flashdokumentet spør alltid etter et bestemt ASP-dokument. ASP-dokumentene sender alltid en filkategoriid med den øvrige informasjonen. Filkategoriiden er kjennetegnet for hva slags type informasjon som overleveres. Flashdokumentet tester alltid på filkategoriiden den får tilsendt av ASP. På denne måten vet Flashfila hva slags informasjon den har fått og til hvilken markør spillehodet må sendes for at informasjonen skal presenteres korrekt.

Allerede da den grunnleggende strukturen for Flashfila ble satt opp, var gruppa klar over at det kunne bli problematisk å hente opp og presentere video på en dynamisk måte i systemet. Det var ikke tid til å teste ut dette nærmere, og Flashdokumentet var satt opp på en slik måte at det ikke ville by på store problemer dersom man måtte gå bort fra dynamisk presentasjon av videoer. I et slikt tilfelle kunne man slette de sekvensene i Flashdokumentet som ikke ville bli brukt.

Figur 7.5 og 7.6 inneholder skjermdumper av tidslinja i ulike faser i utviklingen av Flashdokumentet. Som skjermdumpene viser ble det etter en periode forkastet å vise video. For en nærmere forklaring rundt dette valget, se punkt 8.3.3.



Figur 7.5 Tidslinja i en tidlig versjon av Flashdokumentet. Her er videosekvensen fortsatt er med.



Figur 7.6 Tidslinja i en av de siste versjonene av Flashdokumentet. Videosekvensen er tatt bort etter at det ble besluttet å ikke ha med video i informasjonskanalen.

7.2 Visuelt design

Framfabs designmanual er lagt til grunn for utforming av det visuelle utseendet i prosjektet. I denne manualen ligger det retningslinjer for farge- og fontbruk.

Det er lagt liten vekt på utforming av det grafiske grensesnittet til informasjonskanalen. Dette fordi prosjektet kun skulle resultere i en prototype og hovedmålet med prosjektet var å undersøke om teknologien fungerte tilfredsstillende til det formålet som var ønskelig. Dermed faller arbeid med design på siden av problemstillingen. Dersom det skulle være ønskelig å gjøre utseendet på prototypen mer spennende er det muligheter for dette. Man kan benytte alle de funksjoner som Flash 5 har for å skape en visuell opplevelse.

Det har blitt tatt hensyn til at informasjonskanalen skal kjøres på en TV-skjerm. Måten som benyttes for å presentere et skjermbilde er forskjellig på PC- og TV-skjerm. Vanligvis kan en PC-skjerm vise mer informasjon enn en TV-skjerm. Dette har ført til at mengden tekst som vises per skjermbilde i informasjonskanalen er begrenset. Tekstmengden begrenses naturlig nok også fordi TV-skjermene skal stå i et stort lokale og betraktes på lang avstand. Dette har ført til at det har vært nødvendig å teste leselighet på en viss avstand fra skjermen.

En TV har et høyere lyshetsnivå enn en PC-skjerm. Av denne årsaken brukes ofte hvit tekst på sort bakgrunn når tekst skal presenteres på en TV. Gruppa har testet leseligheten av sort tekst på grønn bakgrunn slik som det er brukt i informasjonskanalen for å forsikre seg om at dette var leselig. Det var det.

Når man skal lage en noe, som skal presenteres på en TV-skjerm, ved bruk av en datamaskin må man ta hensyn til at TV-skjermer har mye lavere oppdateringsfrekvens enn en PC-skjerm. En europeisk TV-skjerm opererer med en oppdateringsfrekvens på 50 Hz, mens en PC-skjerm benytter en frekvens fra 75-90 Hz. Dersom man ikke tar hensyn til denne forskjellen vil bildet som presenteres på TV-en flimre. Det er derfor viktig å tilpasse frekvensen på oppdateringen av skjermen på den PC-en som skal benyttes for å kjøre systemet slik at det visuelles presenteres på en korrekt måte.

8 Utvikling

8.1 Utvikling av Studentprosjektdatabasen

8.1.1 Oppretting av Studentprosjektdatabasen

Oppdragsgiver benytter databaser av typen MS SQL Server. Det var derfor ønskelig at også databasen som skulle opprettes i forbindelse med dette prosjektet var av denne typen. Prosjektgruppa benyttet MS SQL Server 2000 og selve databasen som ble opprettet ble lagret på samme server som oppdragsgiver hadde de øvrige databasene sine.

Det finnes flere måter å opprette en database i SQL Server. Den kan opprettes ved hjelp av et grafisk grensesnitt i en applikasjon som heter Enterprise Manager, den kan opprettes ved hjelp av SQL-syntaks i en applikasjon som heter Query Analyzer eller man kan benytte en kombinasjon av disse metodene. Begge applikasjonene er en del av MS SQL Server. Det spiller ingen rolle for databasens utseende eller funksjon hvilken metode man benytter. I dette prosjektet ble det valgt å benytte Query Analyser for oppretting av databasen. Dette fordi det var ønskelig å lære mest mulig SQL i løpet av prosjektet. For å gjøre småendringer på tabellene og feltene har både Query Analyzer og Enterprise Manager blitt benyttet.

I begynnelsen var det litt usikkerhet omkring hvordan man skulle koble seg til databasen. Det finnes to måter å gjøre dette på i MS SQL Server, Windows authentication og SQL Server authentication. Det ble gitt beskjed fra oppdragsgiver om at Windows authentication skulle benyttes. Dette var greit, fordi da slipper man å ha et eget brukernavn og passord for å koble seg til databasen. Har man først logget seg på nettverket, får man automatisk tilgang til databasene også. Dette førte etter hvert til noen uforutsette problemer. Databasen var opprettet og så ut til å fungere greit. Problemet var imidlertid at databasen var blitt opprettet med kun den personen som hadde laget selve databasen som eier. Dette førte til at det var umulig å hente ut, legge inn og endre data i databasen via ASP. Derfor måtte databasen opprettes en gang til. Da ble SQL Server authentication benyttet ved pålogging og alle tabellene i databasen fikk dbo som databaseeier. Betegnelsen dbo står for database owner. Objekter i en database bør opprettes med dbo som eier for at flere enn bare den som har opprettet objektene skal kunne ha tilgang til den informasjonen som ligger i databasen. Opprettingen av Studentprosjektdatabasen den andre gangen gikk en del raskere enn den første i og med at da hadde gruppa allerede erfaring med oppretting av en database i SQLServer.

Under opprettingen av databasen var det usikkerhet omkring hvordan dato og klokkeslett skulle lagres i databasen. Det var ønskelig å lagre datoen i ett felt og klokkeslett i ett annet. Det finnes to datatyper for lagring av tidspunkter i en MS SQL Server-database; datetime, og smalldatetime. Datetime lagrer tidspunktet med en nøyaktighet på tre hundredels sekund mens smalldatetime

lagrer tidspunktet med en nøyaktighet på ett sekund. Dermed finnes det ingen måte å kun lagre en dato alene i et felt i en database. Problemet ble løst ved at datatypen datetime ble benyttet i alle felt som skulle lagre et tidspunkt. Når systemet så skal hente opp enten en dato eller et klokkeslett benyttes SQL-setninger som kun plukker ut den delen av feltet som er ønskelig.

Eksempel på en enkel SQL-setning som kun plukker ut klokkeslettet fra en dato:

```
SELECT convert (varchar(12),getdate(),108)
```

Eksempel på en SQL-setning som henter det elementet i tidtabellen som skal vises akkurat på det tidspunktet som spørringen foretas:

```
SELECT innslagid FROM Tid
WHERE ((substring(convert (varchar(12),visfrakl,108),1,5) =
substring(convert (varchar(12),getdate(),108),1,5)) AND
(convert (varchar(12),visdato,104) = convert
(varchar(12),getdate(),104)))
```

Ved hjelp av substring-funksjonen kan man definere at kun timen og minuttet som tidspunktet ligger lagret med i databasen skal være lik det klokka på serveren er. Dette for å sikre at man får ett treff selv om det tar litt tid fra spørringen foretas til elementet finnes i databasen. Det er convert-funksjonen som bestemmer hvilken del av feltet som ligger i databasen som skal hentes ut, altså om det er klokkeslettet eller datoen som skal hentes ut.

Databasen skulle også inneholde informasjon om hvor lenge hvert element skulle vises på TV-skjermen dersom ikke noe spesielt klokkeslett var definert. Det var noen uklarheter omkring hvordan dette skulle lagres. Kunne man bruke datatypen datetime eller var datatypen integer et alternativ? Datetime inneholder et bestemt tidspunkt og egner seg derfor ikke for lagring av den informasjonen som var ønskelig i dette tilfellet. Det ble derfor valgt å benytte datatypen integer. Verdien i dette feltet inneholder varigheten til elementet i sekunder. Dette fordi en integer kun kan inneholde et heltall. Derfor må brukeren av systemet regne om varigheten til sekunder før informasjonen legges inn i databasen.

Studentprosjektdatabasens hovedoppgave er å holde styr på den informasjonen som skal vises på TV-skjermen. Dette innebærer både ren tekstinformasjon og informasjon som bilde-, lyd- og videofiler. Sistnevnte type av informasjon omtales ofte som BLOB, Binary Large Objects. Det var en del usikkerhet knyttet til hvordan dette skulle løses. Derfor ble det foretatt undersøkelser for å finne ut hva som var vanlig. Det kom da fram at det finnes to måter å håndtere BLOB-informasjon i forbindelse med en database:

1) Informasjonen kan lagres direkte i databasen. Det vil si at hele fila inkluderes i databasen.
2) Filene ligger lagret på et bestemt sted på en server og databasen inneholder kun referanser til hvor disse filene ligger lagret.

Det første alternativet vil føre til at databasen blir meget stor, i og med at store filer blir inkludert i selve databasen. Dermed vil databasen bli treg å jobbe med og det vil ta lang tid å få returnert data. Fordelen er at man kan være helt sikker

på hvor filinformasjonen ligger. Alternativ to derimot, vil holde databasen på en grei størrelse i og med at feltene som inneholder referanser til de eksterne filene kun er tekstdata. Ulempen er at det ikke finnes noen garanti for at fila det ligger en referanse til i databasen fortsatt ligger lagret på det stedet som er angitt når man skal hente den opp. Derfor krever det siste alternativet at ingen endrer eller flytter de respektive filene på serveren. I dette prosjektet ble det valgt å benytte alternativ to. Dette er den mest vanlige metoden å benytte dersom man ønsker å lage en database som holder styr på BLOB-data.

Det var lenge uklart hvordan tabellene Bildetemp, Bildeteksttemp, Beskjedtemp og Videotemp skulle fungere i praksis. Det ble først forsøkt å opprette en standardverdi for feltet som skulle opprette en randomid ved hjelp av funksjonen RND(). Dermed ble dette feltet fylt med en tallverdi som ble generert automatisk. Problemet var at RND()-funksjonen returnerer et flyttall med veldig mange desimaler. Det er ikke heldig å benytte desimaltall som primærnøkler i databaser. Derfor ble det forsøkt å multiplisere RND()-funksjonen med eksempelvis 1 000 000 000 for å bli kvitt desimalene. Dette fungerte heller ikke optimalt. Problemet var at søk på eksakt samme informasjon i databasen ga forskjellig resultat med Enterprise Manager og Query Analyzer. I Enterprise Manager ble noen av tallene rundet av, mens i Query Analyzer ble de ikke det. Hva dette skyldtes ble det aldri funnet ut av. Derfor ble det valgt å løse problemet på en annen måte. Ved at primærnøkkelen randomid ikke fikk en standardverdi som genererer seg selv, men man er nødt til å legge inn verdien som skal ligge i dette feltet manuelt unngikk man problemstillingen. Dette gjøres ved hjelp av ASP idet ASP-dokumentene fyller temptabellene.

8.1.2 Innlegging av data i Studentprosjektdatabasen

Da databasen var ferdig opprettet var det klart for å legge inn testdata i databasen. Til dette ble Enterprise Manager benyttet. I tillegg ble administrasjonsverktøyet benyttet da det var ferdig. I begynnelsen ble innleggingen av testdata gjort nokså usystematisk. Det viktigste var å se om informasjonen ble hentet opp av ASP-dokumentene og vist fram i Flashpresentasjonen. Etter hvert ble det nødvendig å legge inn informasjon på en strukturert måte slik at det ble lettere å teste om det faktisk var riktig informasjon som ble hentet opp. Derfor ble alt innholdet som lå i databasen slettet og det ble lagt inn informasjon etter det nye systemet. Da testingen av systemet var fullført ble de systematiske testdataene erstattet med informasjon som det gikk an å vise for en forsamling under presentasjonen av selve hovedprosjektet.

8.1.3 Erfaringer fra utviklingen av Studentprosjektdatabasen

- Alle objekter i en database bør opprettes med dbo som databaseeier
- Det er planleggingen av en databasestruktur som tar lang tid. Selve opprettingen av en database i MS SQL Server er relativt raskt gjort.
- Dersom det er ønskelig å inkludere store eksterne filer i en database vil det, i de fleste tilfellene, være mer lønnsomt å legge referanser til filene i databasen, mens selve filene ligger i en mappestruktur på en server
- Det lønner seg å være konsekvent med hensyn til navngiving av databaser, tabeller og felt. Dette gjør det mye enklere å skrive SQL-uttrykk. Da

trenger man ikke å kontrollere syntaksen på de ulike elementene man skal benytte underveis. Derfor bør det opprettes et sett med navngivingsregler før implementeringen av en database begynner. Disse reglene bør så følges konsekvent.

8.2 Utvikling av ASP-dokumentene

8.2.1 Utvikling av ASP-dokumentene som administrasjonsverktøyet består av

Arbeidet med administrasjonsverktøyet ble integrert i en oppgave i et valgfag på skolen. Dette førte til at det ble litt mer omfattende enn problemstillingen til dette prosjektet skulle tilsi. Fra oppdragsgivers side var det et krav at det skulle være mulig å styre selve informasjonskanalen. Prosjektgruppa har i tillegg utarbeidet en metode for å legge inn og endre elementer i databasen ved hjelp av administrasjonsverktøyet.

Først ble det utarbeidet en behovanalyse for administrasjonsverktøyet, se kravspesifikasjonen punkt 6.1. Deretter ble det jobbet en god del med layout og oppbygging av strukturen. Hva som skulle til for at systemet skulle være så enkelt som mulig i bruk og hva som skulle være på de forskjellige sidene var det litt vanskelig å finne gode løsninger på. Da dette var løst ble det satt opp en oversikt over de filene som måtte inngå i systemet, både HTML- og ASP-filer. Til slutt ble det laget en grafisk navigasjonsstruktur med oversikt over hvor alle filene skulle være, se vedlegg B).

Så begynte implementeringen. Det ble startet med den delen av administrasjonsverktøyet som legger informasjon inn i databasen. Her skjer innlogging og lagring av elementer i databasen. Systemet har bare ett brukernavn og passord som er hardkodet inn i selve ASP-dokumentet. Det sjekkes om det er riktig innloggingsinformasjon og dersom det er det blir dette lagret i et sesjonsobjekt. På hver nye side blir dette sesjonsobjektet sjekket. Dette forsikrer man er pålogget før man får mulighet til å endre noe. Dersom noen skriver inn adressen til en av sidene i administrasjonsverktøyet direkte i nettleseren uten å være pålogget blir de videresendt til innloggingssiden. På denne måten er det ikke er mulig å gå inn i systemet uten å ha skrevet inn brukernavn og passord. Til slutt ble sidene hvor man redigerer informasjonen implementert samt de sidene hvor man styrer den informasjonen som skal vises på et spesielt tidspunkt. Vedlegg B inneholder en oversikt over alle filene som inngår i administrasjonsverktøyet samt en kort beskrivelse av de ulike filenes hovedoppgaver.

Da denne delen av systemet ble implementert på Gjøvik, og i forbindelse med en oppgave i et fag på skolen, var koden tilpasset en Access-database. Dette førte til litt problemer når verktøyet skulle tilpasses Studentprosjektdatabasen som er en SQL-database. Koden kunne ikke overføres direkte, men det måtte gjøres om en del på SQL-spørringene. Anførselstegnene skal settes på en annen måte i forbindelse med SQL-databaser enn Access-databaser. Det var litt vanskelig å holde styr på hvor disse anførselstegnene skulle være til en hver tid.

8.2.2 Utviklinę av ASP-dokumentene som informasjonskanalen består av

Først ble det utarbeidet en behovanalyse for informasjonskanalen. Denne sammen med databasestrukturen dannet grunnlaget for ASP-strukturen. Det ble fastsatt hvilken informasjon som skulle sendes til Flash, og hvor mange dokumenter dette ville kreve. Neste steg var å sette opp pseudokoden til ASP-dokumentene. Denne ble sendt til Framfab for tilbakemelding.

Dokumentet Bilde.asp var det første dokumentet som ble kodet. I begynnelsen var det en del problemer med å få opprettet en forbindelse til databasen. Studentene hadde bare erfaring med Access-databaser fra før, og dette var en SQL-database. Dette ble etter hvert løst med god hjelp fra Framfab.

Deretter oppstod det vanskeligheter i forbindelse med å hente ut variabler fra databasen som gruppa hadde opprettet. Det ble brukt lang tid på å løse dette. Grunnen til at dette tok så lang tid, var at serveren ikke genererte feilmeldinger. Dermed var feilen vanskelig å oppdage. Grunnen til at serveren ikke genererte feilmeldinger var at det var litt problemer med noe programvare på serveren. Problemet ble løst ved at det ble installert en desktopserver på studentenes maskiner.

Den neste utfordringen var å hente opp informasjon fra databasen i tilfeldig rekkefølge. Dette krevde en god del undersøkelser. Først ble det forsøkt å løse dette i selve ASP-koden, men det viste seg etter hvert at det ble vanskelig. Løsningen var å utvide databasestrukturen med fire ekstra temptabeller og å generere randomiiden til disse tabellene ved hjelp av ASP.

Dato og klokkeslett viste seg å være vanskelig å arbeide med. Formatering av slik informasjon varierer mellom de forskjellige applikasjonene. For dette systemets del må man legge inn dato på amerikansk vis. Det vil si måned først, så dag og så år med skråstrek imellom. Når det gjelder klokkeslett må dette legges inn med time, minutter og så sekunder og med kolon i mellom. Dette viste seg å være veldig viktig for hvis det ikke kom på korrekt form ble ingenting lagret i databasen.

Det mest frustrerende og tilbakevendende problemet gjennom heler utviklingen av systemet var hvordan anførselstegn skal plasseres inne i SQL-uttrykk i ASP-koden. Selv om SQL-uttrykkene virker i Query Analyzer, så er det ikke sikkert at de virker når de blir satt inn i ASP-koden. ASP krever annerledes plassering av anførselstegnene enn SQL gjør. Det ble satt opp en liste over slike plasseringer:

Dersom variabelen er:

- Et tall → « & variabel & »
- En tekst → ’ « & variabel & » ‘
- En dato → # « & variabel & » #

Men, ingen regel uten unntak. Det var ikke alltid dette virket heller, så det gjaldt bare å være tålmodig.

Da dokumentet Bilde.asp endelig var ferdig var det ganske fort gjort å kode de andre dokumentene som kommuniserer med Studentprosjektdatabasen. Det eneste som var litt vrient var at det ble kopiert en del kode, uten at alt ble tilpasset det nye dokumentet. Det er lett og overse små detaljer underveis. Dermed var det en del som ikke fungerte en stund. Men etter flere grundige gjennomganger ble småfeilene luket ut og ting fungerte som det skulle.

Det bød ikke på særlige problemer å kode de dokumentene som kommuniserer med Framfabs databaser. Her stilte Framfab med noen SQL-setninger som gruppa kunne bruke, da utarbeidelsen av disse krever oversikt over hvor de ulike dataene ligger lagret i disse databasene.

8.2.3 Erfaringer fra utviklingen av ASP-dokumentene

- Lag en grafisk oversikt over systemet med alle dokumenter og sammenhenger samt pseudokode før man starter selve kodingen. Dette er til stor hjelp.
- Vær nøye med rettighetene når man implementerer en database
- Sjekk hvilket språk serveren opererer med, så slipper man formateringsproblemer med datoer.
- Det lønner seg ikke å bruke flyttall i koding dersom dette kan unngås.
- Hold tunga rett i munnen når det gjelder anførselstegn i SQL-uttrykk i ASP-kode.
- Kopiering av kode er farlig. Det er lett å glemme å endre verdiene som gjør de ulike kodedelene litt forskjellige.

8.3 Utvikling av Flashdokumentet

8.3.1 Flashdokumentets oppbygging

Utviklingen av Flashdokumentet ble startet ved å sette opp et rammeverk med inndeling i ulike sekvenser: Default, Bilde, BildeTekst, Tekst, Video, Ansatt, AntAnsatte, Pause og Klokkeslett. Hver del gikk over 10 frames og det ble satt en såkalt «label» eller merkelapp på den første framen i hver del. På dette tidspunkt var det usikkert om 10 frames var tilstrekkelig for hver enkelt del, men det skulle vise seg å være et bra utgangspunkt, og det har ikke vært behov for forandringer i etterkant.

Det ble så satt inn klokke- og datofunksjonalitet. Dette gikk forholdsvis raskt og greit og fungerte med det samme.

Neste steg var å hente opp tekst dynamisk. Det var blitt testet en del på dette under egenlærings- og struktureringsperioden. Derfor gikk dette nokså problemfritt. For å sette inn dynamisk tekst må man kalle et ASP-dokument som sender en variabel som inneholder tekst. I ActionScript ser dette slik ut: loadVariablesNum (”tekst.asp”, 0 , ”GET”); Deretter er det bare å tegne opp en tekstboks, velge «dynamic» på tekst-panelet og skrive inn variabelnavnet, for

eksempel «overskrift», i feltet «variabler». Koden kan man enten knytte til tekstboksen, eller til en frame. I informasjonskanalen er alle ASP-dokumentene kalt opp på en frame fordi dokumentene inneholder flere variabler som skal benyttes i ulike funksjoner i Flashdokumentet.

Alle sekvenser, med unntak av «Default» kaller opp hvert sitt ASP-dokument med ulike variabler. Samtlige ASP-dokumenter som sender variabler til Flash sender med variablene filkategoriid og visTilKlokka. Disse styrer hva som skal vises i informasjonskanalen til enhver tid.

Variabelen filkategoriid styrer loopen slik at spillehodet hopper til ulike labels ettersom hvilken filkategoriid som hentes opp. Det var en del problemer med å få dette til å fungere. Etter å ha strevd en del endte det med at all kode ble fjernet og satt inn på ny. Da fungerte loopen som ønsket. Det er vanskelig å si hvor feilen lå, men det samme ble erfart da variabelen visTilKlokka skulle brukes.

Variabelen visTilKlokka skal styre hvor lenge spillehodet skal stå og loope på hver enkelt sekvens av Flashfilmen. For å teste dette sjekkes visTilKlokka opp mot en variabel sek som teller hvor lang tid det er gått siden spillehodet kom til den enkelte sekvensen. Når sek blir like stor som visTilKlokka, går spillehodet videre til neste sekvens.

I slutten av hver sekvens er det lagt til funksjonalitet som sletter variabler for å frigjøre minne og for å hindre at variabler med samme navn men ulik verdi blandes. Alle variabler som lastes inn i begynnelsen av en sekvens med funksjonen loadVariablesNum() slettes med funksjonen delete.

For å få et ryddig og oversiktlig dokument ble de ulike delene i Flashdokumentet lagt i ulike lag. Følgende lag er brukt:

- **Variable:** variabler fra ASP-dokumenter lastes inn og ut
- **Navigering:** her ligger funksjonalitet for navigering rundt om i informasjonskanalen. Sjekk for hva klokka er, sjekk for verdien av filkategoriid, sjekk for om visTilKlokka er større eller lik tid. Dette laget er også brukt til å laste inn og ut eksterne Flashfilmer.
- **Dynamisk tekst:** her ligger alle dynamiske tekstbokser
- **Labels:** Her ligger alle labels
- **Statisk tekst:** Her ligger all statisk tekst
- **Grafikk:** Dette laget er tomt, men her kan man ved en eventuell videreutvikling legge til grafikkelementer.
- **Timer:** Her ligger MovieClip-ene som inneholder variabelen sek.

Under prosjektets gang har gruppa kommet fram at Flash 5 alene ikke fungerer som et verktøy for å skape dynamiske programmer som inneholder bilder, video og lyd. For å få til dette trengs et ekstra verktøy som for eksempel Macromedia Generator. Å sette seg inn i Generator ville vært å gå utenfor oppgavens grenser. Det ble bestemt at man skulle forsøke å gå omveier for å få vist bilder og video i informasjonskanalen likevel, men at man skulle gå bort fra å implementere lydfiler dersom det ikke viste seg å bli mye tid til overs. Grunnen til at lyd ble nedprioritert var at det i utgangspunktet kun var planlagt å legge til rette for lyd, uten at dette var planlagt brukt.

8.3.2 Håndtering av bilder og lyd på en dynamisk måte i Flash 5

Det er mulig å legge inn statiske jpeg-bilder i et Flashdokument. Skal derimot Flash 5 kunne håndtere dynamiske bilder må de på forhånd være konvertert til swf. Derfor ble bildene som skulle legges inn i informasjonskanalen importert i egne Flashfiler som så ble eksportert til swf-filer. På denne måten fikk man konvertert jpeg-bildene til et format Flash 5 taklet å håndtere dynamisk og det var mulig å presentere bilder på en dynamisk måte. Manuell konvertering av bildefiler til swf-filer er tungvint og tidkrevende. Løsningen vil være tungvinn for en informasjonskanal med flere hundre bilder. Dersom man kun skal benytte noen få bilder som ikke skiftes ut ofte, vil løsningen kunne fungere.

Denne framgangsmåten kunne også blitt brukt for lydfiler. Det vil si at man manuelt kan importere hver enkelt lydfil inn i Flash og eksportere disse til swf. Dette ble ikke prioritert på grunn av tidsmessige hensyn, samt at metoden ikke er virkelig dynamisk. Metoden er lite brukervennlig og det er lite sannsynlig at man ved en eventuell videreutvikling av prototypen ikke vil bruke egne verktøy for håndtering av konvertering av filer til swf.

8.3.3 Håndtering av videofiler på en dynamisk måte i Flash 5

Gruppa fant raskt ut at det å kjøre ut dynamisk video i Flash 5 ville bli et problemområde. Det ble sett på ulike mulige løsninger som er skissert under, men ingen av dem fungerte tilstrekkelig i forhold til dette prosjektets systemkrav.

Det er mulig å importere QuickTimefiler i Flash 5, men man kan ikke kjøre ut filene som swf-filer. Filer hvor man kombinerer QuickTime og Flash må eksporteres som QuickTime (mov) og ikke Flash (swf). Dette er på grunn av at QuickTime støtter Flash (Flash 3 vel og merke) og ikke omvendt. Å kjøre ut Flashfiler som QuickTime-filer var en uaktuell løsning i dette prosjektet fordi QuickTime kun støtter Flash 3 og dermed lite av funksjonaliteten i Flash 5 sin ActionScripting.

En annen mulig løsning er å eksportere video til sekvenser av jpeg-bilder, for deretter å importere disse inn i et movieclip (frame for frame), legge på styringslogikk, synkronisere eventuell lyd, og deretter eksportere til swf. Dette vil resultere i mange jpeg-bilder som tar mye plass i tillegg til at kvaliteten på videoen blir dårlig. Dette er heller ikke en operasjon som kan gjøres dynamisk, og ble dermed utelukket for informasjonskanalen.

Heller ikke Macromedia Generator kan utføre dynamisk konvertering av videoformater til swf. Generator kan foreløpig kun konvertere videoformater til QuickTime.

Det finnes programmer som gjør om videofiler til swf-filer. To slike programmer er «Wildform Flix» (http://www.wildform.com/flashsite.html) og «vid2swf» (http://www.video2flash.com/swf/console/). Om man skal bruke et av disse programmene, må hver enkelt film kjøres manuelt gjennom programmet og ut som en swf-fil før det legges i databasen. Også dette alternativet ble utelukket.

Et siste forsøk for å implementere video i informasjonskanalen ble gjort. Det ble forsøkt å bruke Macromedia Director og Shockwave som mellomledd. Macromedia Director er et verktøy som brukes til produksjon av CD-ROM, samt til å kjøre ut Shockwavefiler. Med Director kan man hente opp videofilmer dynamisk og kjøre ut Shockwavefiler som settes inn i en object-tag på ei HTML-side. Denne HTML-sida vil dermed kunne vise videofilmer dynamisk. Ved å kalle denne HTML-sida fra Flash var intensjonen å få vist videoer dynamisk. Dette fungerte også, men et annet problem meldte seg. HTML-sida skulle lukkes automatisk ved hjelp av et JavaScript og funksjonen close(). Dessverre viste det seg at dette ikke gikk. Det dukket opp en alert-boks i nettleseren som spurte om brukeren virkelig vil lukke vinduet da JavaScriptet ble kjørt. Dette er en innebygd sikkerhet i nettlesere som ikke gir mulighet for å lukke et vindu uten at brukeren aktivt velger lukking. Se figur 8.1. Det ble forsøkt med ulike nettlesere, men alle ga brukeren den samme alertboksen. Foruten Internet Explorer 5.0 ble det testet på Netscape Navigator, Netscape 5 og Netscape 6.

Figur 8.1 Alertboksen som dukket opp da javascriptet som skulle lukke videosida automatisk ble kjørt

Prosjektgruppa endte derfor med å gå bort fra dette alternativet og bestemte seg for å ikke implementere video i selve Flashfila som styrer informasjonskanalen. Alt ligger forøvrig til rette i databasen og i ASP-dokumenter for dynamisk visning av video, slik at det i framtida skal være mulig å legge til denne funksjonaliteten. Det vil nok ikke ta lang tid før Macromedia Generator også kan konvertere videofiler til swf, og da skal det være mulig å bruke Studentprosjektdatabasen og ASP-koden som er blitt utviklet til dette formålet.

8.3.4 Fullskjerms presentasjon

Da prototypen skulle vises på en TV-skjerm, var det ønskelig å unngå å gi seeren illusjoner i retning av en PC. PC-skjermen forbinder man med aktiv jobbing, mens TV-titting betyr fritid, passiv mottakelse av informasjon, underholdning med mer.

For å fremme TV-assosiasjonen kan man fjerne nettleserens grensesnitt. Derfor ble det lagt funksjonalitet for å vise informasjonskanalen i fullskjerm. For å oppnå fullskjermspresentasjon ble det opprettet et HTML-dokument som inneholdt JavaScriptkode som kaller HTML-dokumentet med Flashfilmen og viser dette i fullskjerm. I tillegg var det nødvendig med følgende kode i HTML-dokumentet med Flashfilmen sin body-tag: scroll=no

8.3.5 Erfaringer fra utviklingen av Flashdokumentet

– Flash 5 synes å ha tendenser til å «bugge» som det kalles på dataspråket. Når alle andre utveier er prøvd, kan det være en ide å forsøke å slette en kode og skrive inn denne på ny. Gruppa erfarte i enkelte tilfeller at denne framgangsmåten løste problemene.

– For å få hjelp til Flashrelaterte problemer fins det ulike fora på internett der man kan stille spørsmål. Som regel får man svar på disse spørsmålene innen kort tid. Det kan også lønne seg å gå gjennom foraene før man stiller et spørsmål da det er stor sannsynlighet for at noen har stilt det samme spørsmålet før deg. De to foraene som har blitt brukt flittig under utviklingen av dette prosjektet er følgende:
http://www.were-here.com
http://www.flashkit.com

9 Testing

9.1 Planlegging

Det ble planlagt at testingen skulle foregå kontinuerlig under utviklingen. I tillegg ble det avsatt tid til testing etter at prototypen skulle være ferdig. Når denne tiden kom, viste det seg at det var veldig mange faktorer som skulle testes. Derfor ble det utarbeidet egne skjemaer for å kvalitetssikre og effektivisere testingen, se vedlegg E, F og G.

9.2 Gjennomføring

Det har blitt foretatt testing kontinuerlig underveis i utviklingen, både av databasen, ASP-koden og Flashstrukturen. Da utviklingen av prototypen var ferdig ble det foretatt omfattende tester ved hjelp av testskjemaene, se vedlegg E, F og G. Bruken av disse testskjemaene gjorde det enklere å holde orden på hva som skulle testes. I tillegg forsikret dette at alle funksjonene i systemet ble grundig gjennomgått og at ingen ble glemt.

Systemet er kun testet ved hjelp av det utstyret prosjektgruppa har fått tilgjengelig av oppdragsgiver. Dette er nærmere spesifisert i kapittel 5 samt i kravspesifikasjonen, kapittel 6.

10 Resultater

Under arbeidet med forprosjektet ble det undersøkt om hvordan det var å kombinere Macromedia Flash 5 med ulike multimedieelementer. Det viste seg at applikasjonen Macromedia Generator er utviklet for dette formålet. Dette er en meget kostbar programvare og studentene fikk fra flere hold høre at det var noe ustabilt i bruk. Det ble foreslått å se på andre alternativer. Prosjektgruppa hadde et ønske om å finne ut om det gikk an å lage et dynamisk system i Flash 5 uten å være avhengig av Generator.

Det ble derfor bestemt at gruppa skulle forsøke å benytte ASP som eneste mellomledd på serveren, selv om dette kunne bli vanskelig. Det er ikke så mange som har erfaring med å kombinere ASP og Flash når det gjelder dynamisk presentasjon av multimedia. Derfor var det ingen av de som ble kontaktet som kunne si helt sikkert om prosjektets problemstilling var løsbar eller ikke med dette valget. Dette ble sett på som en utfordring uten at det ble fortatt nærmere undersøkelser for å finne ut om det virkelig ville være mulig.

Det viser seg at Flash 5 fungerer greit i kombinasjon med databaser og ASP når det gjelder å presentere tekst, bilde og lyd dynamisk. Forutsetningen for at dette skal fungere på en ønskelig måte er at alle bilde- og lydfiler må ligge lagret i systemet som filtypen swf. Riktignok er dette en tungvint løsning fordi swf ikke er et vanlig format for slike filer. Dersom man benytter en metode som gjør det mulig å konvertere vanlige filformater for lyd og bilder til swf automatisk på serveren vil dette problemet være løst.

Det er for så vidt mulig å modifisere for eksempel jpeg-formater eller andre formater ved hjelp av ASP slik at de får riktig dataformat og vil identifisere seg som swf-filer. Dette er imidlertid meget tungvindt. Macromedia Generator forandrer ikke på selve rådataene i filene, men modifiserer hodet i fila slik at den blir tolket som en swf-fil. Dette kan også gjøres ved hjelp av ASP hvor man da lager maler for modifisering av hodet til de forskjellige filformatene. Disse malene vil kun takle filer med en bestemt bildestørrelse. Dersom det skal benyttes bilder med flere forskjellige størrelser må det lages tilsvarende antall modifiseringsmaler. Det finnes derimot en mulighet for å generere slike maler automatisk, men da er man inne på å utvikle en ny applikasjon som ligner Generator. Dette er utenfor prosjektets problemstilling.

Video er derimot et problemområde i Flash 5. Det er ikke mulig å implementere video på en dynamisk måte. Dette fordi det ikke finnes noe metode for å konvertere videoformater som mov og mpeg til swf-filer. Derfor er det ikke mulig å benytte video som en del av den dynamiske informasjonskanalen som prosjektgruppa har jobbet med.

Prototypen fungerer bra med hensyn til tidsstyring av elementer i forhold til de kravene som ble stilt til systemet i utgangspunktet. Det er mulig å definere at et element skal vises et bestemt tidsrom med en hyppighetsfrekvens på daglig eller ukentlig. I tillegg kan et element vises en bestemt dato.

Prototypen foreligger som et resultat i forhold til undersøkelser gjort underveis i prosjektet, og er i samsvar med oppgavebeskrivelsen og kravspesifikasjonen så langt det har vært teknisk mulig. Det vil si at informasjonskanalen kan presentere tekst og bilder, hver for seg og sammen, på en dynamisk måte. Dette forutsetter at alle bilder er tilpasset systemet på forhånd. Bilde- og lydfilene må være av type swf samt at oppløsningen på bildene må være tilpasset TV-skjermen de skal presenteres på. Når det gjelder video er det ikke funnet noen god måte å implementere dette i Flash 5. Likevel er det lagt til rette for at dette skal fungere både i Studentposjektdatabasen, ASP-koden og administrasjonsverktøyet.

11 Konklusjon

Vektorbasert bevegelsesteknologi som Macromedia Flash 5 fungerer ikke som et optimalt rammeverk for en dynamisk multimedieinformasjonskanal i kombinasjon med ASP. Gjennom prototypen er det likevel bevist at det er mulig å få dette til å fungere når det gjelder bilde, tekst og lyd. Dersom systemet skal være dynamisk takler Flash kun filer av typen swf. Dette fører til at filer som skal inkluderes i en dynamisk Flashfilm først må konverteres til formatet swf. Slik konvertering er tungvint dersom det må gjøres manuelt. Det finnes ingen enkel og egnet måte å konvertere filene automatisk på serveren ved hjelp av bare ASP. Dessuten er ikke Flash 5 en egnet applikasjon for å vise fram video uansett om videofila ligger statisk i Flashfila eller den skal presenteres dynamisk. Derfor er ikke Flash 5 å anbefale som et rammeverk dersom målet er å vise fram multimedieelementer på en dynamisk måte kun ved hjelp av ASP.

12 Tips til videreutvikling

Hovedproblemet med prototypen slik den foreligger i dag, er at det ikke finnes noen god måte å konvertere bilde- video- og lydfiler, til et format som Flash 5 takler, på serveren ved hjelp av ASP. Det anbefales derfor å se nærmere på de mulighetene som ligger i verktøy som benyttes til generering av swf-filer. Det fins flere slike verktøy. Noen koster endel, mens andre er gratis. Under presenteres noen av de mest kjente verktøyene som kan gjøre lyd- og bildefiler om til swf-filer:

- Swift Generator (http://www.swift-tools.com) er et verktøy utviklet av Swift-tools. Programmet er forholdsvis rimelig. Dessuten er det lite, kun 178kB.
- FlashTurbine (http://www.blue-pac.com) er et verktøy utviklet av Blue Pacific. Flashinnhold lages dynamisk ut fra ASP-skript. Flash Turbine er utviklet uavhengig av Flash og trenger kun et Flash plug-in for å kjøres.
- JGenerator (http://www.flashgap.com). Dette verktøyet er utviklet av Dmitry Skavish. Kildekoden er åpen og det er fritt fram for en hver med programmeringskunnskap å bidra til videre utvikling av programmet. Verktøyet er gratis og er ment å skulle kunne tilsvare kostbare Macromedia Generator etter noe mer utvikling. Det er java-basert og jobber svært raskt.
- Macromedia Generator (http://www.macromedia.com/software/generator/) er den originale, men den mest kostbare, måten å lage dynamisk Flash. Macromedia Generator fungerer slik at en såkalt «template»-fil med variabler vil ligge på serveren og erstatte disse variablene med innhold. Generator skiller innhold fra design, og når innholdet forandres blir det automatisk reflektert enten med det samme eller til faste tidspunkt. Macromedia Flash brukes til å skape det visuelle utseende, elementer og templates til Generator, mens Generator brukes til å skape personifisert og dynamisk innhold.

Administrasjonsverktøyet har begrenset funksjonalitet. Det er derfor ikke mulig å ha full kontroll over informasjonskanalen ved å bare benytte seg av dette. Dersom det er ønskelig kan derfor administrasjonsverktøyet utvides. Under følger en liste med funksjoner som det kan være hensiktsmessig å tilordne administrasjonsverktøyet:

- muligheter for å legge inn og endre hvor lenge et element skal vises i informasjonskanalen selv om det ikke skal vises på bestemt tid
- muligheter for å slette et tidspunkt som er koblet til et element slik at elementet kun kommer tilfeldig i loopen igjen
- muligheter for å slette elementer i databasen og dermed også informasjonskanalen
- muligheter for å styre rekkefølgen elementene skal presenteres i loopen annet enn dersom det legges inn på klokkeslett. Slik systemet er lagt opp i dag er dette ikke mulig. Informasjonskanalen kjører i en bestemt rekkefølge så lenge den ikke blir avbrutt av noe som skal vises på tid

Beskjeder og tekster kombinert med bilder vises kun på ett skjermbilde slik prototypen framstår i dag. Dette fører til at kun et begrenset antall karakterer kan vises på et skjermbilde. Dersom det er ønskelig at det skal være mulig å presentere mer tekst på en gang kan det derfor være en idé å forsøke å tilordne systemet en «rulletekst»-funksjon.

For å sikre at informasjonskanalen gir ønsket effekt når det gjelder innhold bør det foretas en spørreundersøkelse blant de som har sett informasjonskanalen i bruk. Dermed kan en finne ut om den har noe effekt; hva folk får med seg og hva de ikke oppfatter, hvordan de opplever informasjonskanalen og så videre. I og med at prosjektgruppa kun skulle lage en prototype som bare inneholdt testdata falt dette utenfor studentenes oppgave.

13 Egenvurdering

Vi mener å ha løst oppgaven på en god måte i forhold til problemstillingen. Vi har tatt lærdom av de erfaringene vi har gjort underveis i prosjektet og føler vi er bedre rustet til å ta fatt på nye utfordringer enn det vi var før vi startet prosjektet. I utgangspunktet hadde vi lite fagkunnskaper innenfor det fagfeltet som trengtes for å løse oppgaven. Derfor er vi fornøyde med at vi har klart å takle denne utfordringen.

Vi kunne gjerne tenkt oss at det visuelle utseende på prototypen var mer spennende enn det resultatet som foreligger. Arbeidet med dette ble nedprioritert i forhold til den tekniske utviklingen. Da vi startet prosjektet så vi ikke for oss at det tekniske aspektet ville være så tidkrevende som det faktisk viste seg å være. Derfor regnet vi med at vi skulle få bedre tid til design enn det vi fikk.

Det har vært gøy å ha muligheten til å fordype seg i en reell oppgave over lengre tid. Vi føler at vi har lært mye, både i forhold til prosjektplanlegging og de fagområdene som inngått i prosjektet.

14 Litteraturliste

Bhangal, Sam – «Foundation ActionScript».

Brooks, Patton and Franklin, Derek – «Flash 5! Creative Web Animation».

Chun, Russel – «Flash Advanced For Windows and Mackintosh».

Clayton Crooks. – «Flash Alternatives: Swish and ASP Turbine». http://www.webtechniques.com/archives/2000/10/desirevu2/ (08.05.01)

Dalby, Joakim – «Databasehåndbogen.

Dobbelaar, Astrid – «Televising Cyberspace: and the web would never be the same» http://www.w3.org/Architecture/1998/06/Workshop/paper 37/ (04.05.01)

Emerton, David J. Scott Hamlin – «Flash 5 Magic».

Kaufmann, John. – «Beginning ASP Databases».

Kolloen, Øivind. – «Kompendium: Aktive server sider, ASP».

«Microsoft SQL Server 2000 Online Book».

Niederst, Jennifer. – «Web Design in a Nutshell».

Riordan, Rebecca M. – «Microsoft SQL Server 2000 programming step by step».

Mitchell, Scott and Atkinson, James – «Teach your self ASP in 21 days».

Viera, Robert – «Professional SQL Server 2000 Programming».

Westhagen, Harald – «Prosjektarbeid».

15 Ordforklaringer

ActionScrip
Skriptspråk som brukes i Macromedia Flash

ASP
Active Server Pages. Et programmeringsspråk. Koden kjøres på en server og genererer HTML

Attributt
En kolonne eller et felt i en databasetabell

Bitmapbilder
Digital representasjon av grafikk eller bilder ved at individuelle piksler arrangeres i rader og kolonner (horisontalt og vertikalt).

BLOB
Binary Large Objects

Cookie
Informasjon som lagres på klienten, men som brukes av serveren for å ta vare på sesjonsidentifikatorer og informasjon om enkeltbrukere av et system.

Datatype
Beskrivelse av hva slags type informasjon en variabel eller et felt i en database inneholder. F.eks heltall, flyttall, tekst, dato osv.

DSN
Data Source Name

Dynamisk presentasjon
En presentasjon hvor innholdet ikke ligger fysisk lagret i en fil, men det genereres et grensesnitt ut i fra innholdet i en database.

Egenskaper
Attributter som beskriver et objekt. I ActionScript har man for eksempel følgende egenskaper: _visible, _height, _width.

Entitet
En tabell i en database som er organisert i rader og kolonner

Fremmednøkkel
En eller en kombinasjon av flere kolonner i en tabell som refererer til primærnøkkelen i en annen tabell. Det er fremmednøkler som binder tabellene sammen vha relasjoner.

Handlinger (actions)
Uttrykk som instruerer en Flashfilm til å utføre en handling mens den spilles.

HTML
Hyper Text Markup Language. Kodespråk som benyttes for å lage internettsider

IIS
Internet Information Server. En type web-server

Instanser
objekter som tilhører en spesiell klasse. Hver instans av en klasse innehar alle egenskaper og metoder til klassen.

Integer
Datatype som representerer et heltall

Klient
Den delen som benytter tjenester fra serveren i forbindelse med et klient/server-system. Dette er vanligvis den delen som brukeren ser og foretar interaksjon med (ofte en nettleser).

Macromedia Flash
En applikasjon som er utviklet for å lage vektorbaserte animasjoner

Metoder
Funksjoner som hører til et objekt.

Normalisering
Inndeling av data i entiteter og attributter på en logisk måte slik at ikke data repeteres unødvendig i databasen. Inndelingen gjør det også enkelt å bringe dataene sammen igjen. Har til hensikt å fjerne redundans og anomali. Er en prosess hvor «utilfredsstillende» entiteter splittes opp i mindre entiteter.

Objekter
Samlinger av egenskaper; hvert objekt har dets eget navn og verdi. Objekter gjør det mulig å aksessere en viss type informasjon. For eksempel så inneholder Date-objektet informasjon fra systemklokka.

Piksel
Mål for oppløsning. Tilsvarer et punkt på dataskjermen

Plug-in
Lite program som tilfører et program spesielle tilleggsfunksjoner. Det er ikke uvanlig at en slik plug-in kan fungere sammen med flere programmer. En plug-in kan f.eks. brukes for å få ekstra funksjonalitet i en Web-leser.

Post
En rad i databasen dvs. All informasjon som hører til et element i databasen

Primærnøkkel
Unik identifikator for hver rad i en tabell.

Primærtabell
Den kontrollerende tabellen i en relasjon, altså den tabellen hvor det attributtet som binder to tabeller sammen er primærnøkkel

Protokoll
Et sett av regler og prosedyrer som er brukt til å formulere standarder for overføring av informasjon mellom enheter i et nettverk eller mellom en maskin og eksterne enheter tilknyttet maskinen.

Prototyping
Sirkulær arbeidsmetode hvor et system gradvis forbedres litt etter litt

PWS
Personal WebServer. En type web-server

Referanseintegritet
Et sett med regler som blir brukt for å sikre at relasjoner mellom poster i relaterte tabeller er gyldige og at man ikke kan slette eller endre relaterte data. Sikrer at en post i en sekundærtabell må refererer til en eksisterende post i den tilhørende primærtabellen.

Sekundærtabell
Den underordnede tabellen i en relasjon, altså den tabellen hvor det attributtet som binder to tabeller sammen er fremmednøkkel.

Sesjon
Flere oppkoblinger mot en server over tid som er gjort for å få utført en oppgave eller et sett med oppgaver.

SQL
Structured Query Language. Dette er et databaseprogrammeringsspråk. Det benyttes både til å opprette en database, legge inn, endre og slette data samt å hente ut data fra en database

SQL-database
Database av typen MS SQL Server 2000

Variabel
beholdere for verdier for enhver type data. Verdiene som lagres i variablene kan hentes opp og brukes i skript.

VBScript
Subset av programmeringsspråket Visual Basic

Vektorgrafikk
Grafikk bestående av vektorer som beskriver linjene mellom punkter. Det er dermed mulig å forstørre eller forminske grafikken uten at dette går ut over kvaliteten - i motsetning til grafikk bestående av enkeltpunkter (piksler).

Wildcard
Ett tegn som kan brukes for å representere en eller flere andre karakterer (tegn). F.eks brukes tegnet * eller % ofte for å representere flere andre tegn.