Dynamiske grensesnitt i digitale bibliotek

- Et forslag til arkitektur for digitale bibliotek baserert på dynamiske grensesnitt mot heterogene samlinger

av
Ole-Martin Bækkeli

Hovedfagsoppgave i Informasjonsforvaltning
Institutt for Datateknikk og Informasjonsvitenskap
Norges Teknisk-Naturvitenskapelige Universitet

September 2002

Forord

Veien frem til dette ferdige produktet av min hovedfagsavhandling i informatikk har både vært lang og utfordrende, men ikke minst en lærerik og inspirerende prosess. Det har vært en spennende periode med både oppturer og motgang, som jeg føler jeg, både personlig og kunnskapsmessig, har vokst veldig på.

Jeg vil gjerne få benytte anledningen til å takke alle som har deltatt i denne prosessen. Spesielt vil jeg takke min veileder, Ingeborg Sølvberg, for all tillitt, tålmodighet og dyktig veiledning gjennom hele min tid som hovedfagsstudent.

Det er også mange andre som har bidratt på sine måter. Einar fortjener en stor takk for mange tilbakelagte timer på lesesalen, gjennomlesninger av avhandlingen og ellers faglige innspill gjennom mange år. Jeg vil også rette en stor takk til Eir for grundige gjennomlesninger og konstruktive tilbakemeldinger, Mats for profesjonell bistand ved utviklingen av det grafiske grensesnittet til prototypen, og ellers tusen takk til alle som har bistått med moralsk og faglig støtte!

NTNU, 20. September, 2002

Ole-Martin Bækkeli

Sammendrag

Denne avhandlingen tar for seg forholdet mellom grensesnitt og samling i digitale bibliotek. En arkitektur for digitale bibliotek som understøtter dynamiske brukergrensesnitt og heterogene samlinger blir foreslått og gjennomgått.

Arkitekturen viser hvordan man kan bygge opp et digitalt bibliotekssystem som støtter informasjonsgjenfinning på tvers av samlinger med ulik form og innhold (heterogene samlinger), og som samtidig også støtter et utvalg ulike grensesnitt. Arkitekturen benytter seg av en abstrahering av hver av samlingene og grensesnittene i forhold til resten av systemet ved hjelp av intelligente agenter og samlingsmetadata.

Avhandlingen går inn på hvordan flere uavhengige brukergrensesnitt (eksempelvis web-browsere, lommepc og mobiltelefoner) kan samles i en felles arkitektur for digitale bibliotek. Den tar også for seg hvordan det kan legges til rette for at samlinger med varierende medieformater og innhold, på samme måte som grensesnitt, også kan integreres i et felles digitalt bibliotek.

Et mål med avhandlingen er å belyse muligheter for tilpassede brukermiljø. Dette hovedsaklig ved at brukeren selv skal kunne påvirke hva slags type informasjon eller funksjonalitet som skal vektlegges innenfor grensesnittet. Dette målet ønskes også oppnådd ved at brukeren har et utvalg av samlinger til rådighet, helt uavhengig av grensesnittet som brukes. Brukeren velger dermed selv sitt ønskede grensesnitt, og tar del i det digitale biblioteket. i så stor grad som dette grensesnittet gjør det praktisk mulig.

Det sees også på hvordan integreringen av både grensesnitt og samlinger kan abstraheres i det digitale biblioteket, for dermed i stor grad å gjøre dem uavhengige av hverandre. Kjernen i et digitalt bibliotek håndterer dermed både samlinger og grensesnitt på et generelt grunnlag. Dette letter arbeidet med å fjerne og legge til grensesnitt og samlinger i et allerede operativt digitalt bibliotek og er også fordelaktig der man har samlinger og/eller grensesnitt som ikke alltid er tilgjengelige.

Avhandlingen tar for seg enkelte grunnbegreper innenfor digitale bibliotek, med spesiell vekt på **agenter**, **grensesnitt** og **samlinger**. Den belyser også aktuelle arkitekturer for digitale bibliotek som tar for seg problemstillinger, og mulige løsninger, innenfor dynamiske brukergrensesnitt (*ref. side 19, "Arkitekturer for digitale bibliotek"*).

En arkitektur for et agentbasert digitalt bibliotek foreslås og gjennomgås i detalj. Her sees det på forholdet mellom grensesnitt og samling i en tre-komponent arkitektur (*ref. side 34, "Arkitekturens hovedkomponenter"*), igjen med fokus på grensesnitt, agenter og samlinger.

En prototype basert på arkitekturen (*ref. side 53, "Prototype"*) og flere eksempel (*ref. side 43, "Informasjonsflyt" og side 65, "Casestudie av prototypen"*) vil illustrere nøkkelelementene i arkitekturen.

Innholdsfortegnelse

Forord ..... *iii*
Sammendrag ..... *v*
Liste over figurer ..... *xi*
Liste over eksempel ..... *xiii*
Liste over tabeller ..... *xv*

Kapittel 1. Innledning 1

1.1 Problemstilling og avgrensning 2
- 1.1.1 Bakgrunn og motivasjon
- 1.1.2 Problemidentifisering
- 1.1.3 Problemstilling
- 1.1.4 Mål med arkitekturen
- 1.1.5 Mål med prototypen
- 1.1.6 Tilnærming
- 1.1.7 Teoretisk basis

1.2 Avhandlingens oppbygning 6
- 1.2.1 Litteraturanalyse
- 1.2.2 Forslag til arkitektur
- 1.2.3 Prototype basert på arkitekturen
- 1.2.4 Diskusjon og evaluering

Kapittel 2. Terminologi 9

2.1 Begrepet digitale bibliotek 9
- 2.1.1 Digital Library Federation
- 2.1.2 D-Lib
- 2.1.3 Avhandlingens definisjon av digitale bibliotek

2.2 Brukergrensesnitt 12

2.3 Forholdet mellom arkitektur, modell og prototype 13

2.4 Dynamisk informasjon og dynamiske grensesnitt 14
- 2.4.1 Informasjon og data
- 2.4.2 Grensesnitt
- 2.4.3 Avhandlingens bruk av utrykket

2.5 Metadata 15
- 2.5.1 To definisjoner på metadata
- 2.5.2 Metadata om samlinger av data

2.6 Samlinger, databaser og datalager 16

2.7 Agenter og agentbaserte arkitekturer 17
2.7.1 Agenter
2.7.2 Agentbaserte arkitekturer
2.7.3 Avhandlingens definisjon

Kapittel 3. Arkitekturer for digitale bibliotek 19

3.1 SmartPush 20
3.2 Daffodil 21
3.2.1 Arkitekturen
3.2.2 Prototype
3.3 University of Michigan Digital Library (UMDL) 24
3.3.1 Retningslinjer og mål for arkitekturen
3.3.2 Agentklasser
3.3.3 Samarbeid og innkapsling
3.4 MyLibrary@NCState 26
3.4.1 Modell
3.4.2 Oversikt over implementasjonen
3.5 Sammendrag 31

Kapittel 4. Forslag til arkitektur for digitale bibliotek 33

4.1 Arkitekturens hovedkomponenter 34
4.1.1 Datalager
4.1.2 Agenter
4.1.3 Grensesnitt
4.1.4 Dynamiske grensesnitt og samlinger
4.2 Inngående beskrivelse av komponentene i arkitekturen 39
4.2.1 Basis-system
4.2.2 Datalager
4.2.3 Spesifikke samlingsagenter
4.2.4 Overordnet samlingsagent
4.2.5 Brukerinfo og profiler
4.2.6 Informasjonsmegler
4.2.7 Agenter med domenekunskap om grensesnitt
4.2.8 Brukeren og grensesnittene
4.3 Informasjonsflyt 43
4.4 Teknologi 50
4.4.1 Databaseteknologi
4.4.2 Integrering
4.4.3 Grensesnitt
4.5 Oppsummering 51

Kapittel 5. Prototype 53

5.1 Systembeskrivelse 54

5.2 Bakgrunn og historie 54
5.3 Målgruppe for prototypen 55
5.4 Agenter 56
5.4.1 Backend agenter
5.4.2 Grensesnittagenter
5.4.3 Samlingsagenter
5.5 Grensesnitt 57
5.5.1 Web-grensesnitt
5.5.2 PDA-grensesnitt
5.5.3 Forholdet mellom WEB og PDA
5.6 Datalager 61
5.6.1 Brukerdatabase
5.6.2 Tjenestedatabase
5.6.3 TV-Program
5.6.4 Økonomi
5.7 Casestudie av prototypen 65
5.7.1 Brukervinkel
5.7.2 Systemvinkel
5.8 Skalerbarhet 69
5.9 Teknologi 69
5.9.1 Datalagring
5.9.2 Informasjonsdistribusjon
5.9.3 Backend system

Kapittel 6. Erfaringer og analyse 73

6.1 En analyse av arkitekturene 74
6.1.1 Grensesnitt
6.1.2 Forholdet mellom grensesnitt og samling
6.1.3 Grensesnitt basert på profiler
6.1.4 Agentbaserte arkitekturer
6.2 Erfaringer 77
6.2.1 Arkitekturens fordeler
6.2.2 Ulemper med den foreslåtte arkitekturen

Kapittel 7. Evaluering og videre arbeid 81

7.1 Evaluering av arbeidet med avhandlingen 81
7.1.1 Prosjektarbeidet
7.1.2 Arkitekturens forhold til prototypen
7.1.3 Evaluering av måloppnåelse for arkitekturen
7.1.4 Arbeidet med avhandlingen
7.2 Utvidelser og videre arbeid 83
7.2.1 Utvidelser av emner avdekket i avhandlingen
7.2.2 Digitalt Biblioteksprosjekt (DigLib)
7.2.3 E-handel
7.2.4 Born-digital/deep searches

Kapittel 8. Bibliografi 87

1.1 Mal for biblografiske referanser 87
1.2 Artikkelreferanser benyttet i avhandlingen 87
1.3 Internettreferanser benyttet i avhandlingen 90
1.4 Ressurser benyttet under arbeidet med avhandlingen 91

Appendix A. PåVei Underholdning 93

A.1 Historie 93
A.2 Status 93

Appendix B. Datalager 95

Appendix C. Kildekode, WEB-grensesnitt 103

C.1 Oversikt 103
C.2 Kildekode 103

Appendix D. Kildekode, PDA-grensesnitt 145

D.1 Oversikt 145
D.2 Kildekode 145

Appendix E. Kildekode, backend-agenter 191

E.1 Oversikt 191
E.2 Backup 191
E.3 Cleanup 192
E.4 TV-program parser 192

Liste over figurer

Fig. 1-1 Aktørene, og inntresseområdene til disse innenfor digitale bibliotek **[GCES01]** 4
Fig. 2-1 Noen mulige grensesnitt og samlinger i et digitalt bibliotek 12
Fig. 2-2 Forholdet mellom arkitektur, modell og prototype 13
Fig. 2-3 Samlinger innenfor distribuerte digitale bibliotek **[CLDF98]** 17
Fig. 3-1 SmartPush: Agenter i innholdsleveransen 20
Fig. 3-2 SmartPush: Oversikt over arkitekturen 20
Fig. 3-3 Daffodil: Arkitektur **[FUHR00]** 22
Fig. 3-4 Daffodil: Prototype (Java) 23
Fig. 3-5 UMDL Arkitekturen **[BIR95]** (oversatt fra eng.) 25
Fig. 3-6 UNDL: to-stegs prosessering av søk **[WEI99]** 26
Fig. 3-7 "NCSU MyLibrary Model" 27
Fig. 3-8 "NCSU MyLibrary Technical Infrastucture" 28
Fig. 3-9 MyLibrary@NCState: Grensesnitt 30
Fig. 4-1 Arkitektur, Basiskomponenter 34
Fig. 4-2 Modell av arkitekturen 38
Fig. 4-3 Eksempel på informasjonsflyt mellom agentene 43
Fig. 5-1 PåVei Underholdning med tre ulike grensesnitt 53
Fig. 5-2 Skjermdump fra tidlig utgave av prototypen (PalmOS) 54
Fig. 5-3 Informasjonsflyt mellom bruker, WEB- og PDA-grensesnitt 58
Fig. 5-4 WEB grensesnitt 59
Fig. 5-5 PDA Grensesnitt (PocketPC) 59
Fig. 5-6 WEB til PDA - blå piler og haker er brukervalg - røde piler er system-valg 60
Fig. 5-7 Konfigurering av tjenester i WEB grensesnitt 65
Fig. 5-8 Bruksanvisning, PåVei / PDA 66
Fig. 5-9 Login dialog 66
Fig. 5-10 Teknologien som benyttes i prototypen 70
Fig. 6-1 Forholdet mellom grensesnitt og samling 75
Fig. B-1 Brukerdatabase <sql:brukere> 95
Fig. B-2 Chatboard <sql:chatboard> 95
Fig. B-3 TV-kanal mapping <sql:tv_conv> 96
Fig. B-4 Tjenestedatabase <sql:tjenester> 96
Fig. B-5 Loginfo <sql:logs> 97
Fig. B-6 Kinodatabase <sql:kino> 97
Fig. B-7 Aksjedatabase <sql:aksje> 98
Fig. B-8 Aksjefond-database <sql:aksjefond> 98
Fig. B-9 Stock: Equity <sql:stock_equity> 99
Fig. B-10 Stock News <sql:stock_news> 99
Fig. B-11 Stock Shareidx <sql:stock_shareidx> 100
Fig. B-12 Horoskop <sql:horoskop> 100
Fig. B-13 Været <sql:vaer> 101

Fig. B-14 Hitcompany <sql:hitcompany> 101
Fig. C-1 Oversikt, kildekode til WEB-grensesnitt 103
Fig. D-1 Oversikt, kildekode til PDA grensesnitt 145

Liste over eksempel

Eks. 4-1	Eksempel, B2C/B2B virksomhet 41
Eks. 4-2	Eksempel, To grensesnitt mot samme kilde 42
Eks. 4-3	Eksempel, "Søk: Mønster" 44
Eks. 4-4	Søkedialog i et PDA grensesnitt 45
Eks. 4-5	POST query mellom grensesnitt [G1] og grensesnitt-agent [A1] (noe simplifisert) 45
Eks. 4-7	Dataflyt mellom informasjonsmegler [A3] og brukerinfo/profiler [A5] 46
Eks. 4-8	Dataflyt mellom brukerinfo/profiler [A5] og informasjonsmegler [A3] 46
Eks. 4-6	Dataflyt mellom grensesnitt-agent [A1] og informasjonsmegler [A3] 46
Eks. 4-10	Dataflyt mellom overordnet samlingsagent [A4] og en samlingsagent (1) [A7.1] 47
Eks. 4-11	Dataflyt mellom overordnet samlingsagent [A4] og en samlingsagent (2) [A7.2] 47
Eks. 4-12	Dataflyt mellom samlingsagent (1) [A7.1] og tilhørende datalager [D1.1] 47
Eks. 4-13	Dataflyt mellom samlingsagent (2) [A7.2] og tilhørende datalager [D1.2] 47
Eks. 4-9	Dataflyt mellom informasjonsmegler [A3] og overordnet samlingsagent [A4] 47
Eks. 4-14	Dataflyt mellom samlingsagent (1) [A7.1] og overordnet samlingsagent [A4] 48
Eks. 4-15	Dataflyt mellom samlingsagent (2) [A7.2] og overordnet samlingsagent [A4] 48
Eks. 4-16	Dataflyt fra overordnet samlingsagent [A4] via inf. megler [A3] til gr.snitt-agent [A1] (fork.) 49
Eks. 4-17	Resultat på PDA 49
Eks. 5-1	Utdrag fra en tjenesteoversikt for en gitt bruker 63
Eks. 5-2	Bruksanvisning fra systemet (e-post) 67

Anmerkning: refererer til tekstbaserte eksempel

Liste over tabeller

Tabell 4-1 Oversikt over noen mulige interaktive grensesnitt 36
Tabell 5-2 Utdrag fra brukerdatabasen 62
Tabell 5-3 Tjenestedatabasen 62
Tabell 5-1 Brukerdatabasen 62
Tabell 5-4 Databasefelter for "TV Program" 63
Tabell 5-5 Databasefelter for "Fond" 64

1 Innledning

Vi lever i et samfunn som praktisk talt flyter over av informasjon. Dette gjelder de tradisjonelle og store mediekanalene (aviser, radio og tv) så vel som et stort mangfold av regelmessige og uregelmessige publikasjoner. Ikke minst gjelder dette internett, og da spesielt tjenester og informasjon som publiseres gjennom det som har blitt kjent som *WorldWideWeb* (heretter referert til som *web* eller *www*). Man sitter med et stort antall informasjonsleverandører, og bokstavelig talt et hav av potensielle brukere med et vidt spekter av informasjonsbehov og krav.

De tradisjonelle informasjons-formidlerne (TV-kanaler, aviser, radio, bibliotek, bokhandlere etc.) har i gjennom mange år opparbeidet visse faste metoder for informasjonsformidling og distribusjon. Disse er preget av blant annet *verdikjeder* (eksempelvis for musikk: låtskriver/artist, studio, distributør og sluttselger), *begrensninger i teknologi* (for eksempel at aviser må være skrevet i sin helhet før de går i trykken, og radiosendinger kun er begrenset til lyd) og *tradisjon* (eksempelvis formatering på bøker og avisartikler, og nyhetsopplesere i TV).

Å lage et helt spesifikt og statisk grensesnitt for å tilfredsstille alle potensielle brukere, er en vanskelig, og noen ganger nesten umulig, oppgave. Tradisjonelt kan man si at disse problemene er blitt løst, ofte gjennom en lang prosess, ved å enes om standarder som de fleste følger. Et eksempel på dette, er systemet aviser ofte benytter på sine artikler, med en inndeling i overskrift, ingress, forfatter og brødtekst. Dermed er det, på mange måter, hele tiden brukerne som i varierende grad må tilpasse seg formidlerne, mens formidlerne i beste fall tilpasser seg det som de regner med flertallet av brukerne ønsker. Vil man ha nyheter er to av alternativene at man enten kjøper avisen, selvsagt i åpningstiden til sin lokale avisselger, eller ser på nyhetssendingen, men da til et gitt tidspunkt og på en gitt kanal.

På mange måter kan man si at overgangen til IT-basert formidling av informasjon er sterkt preget av sine "forgjengere". Kjente metaforer er blitt overflyttet til det nye mediet, med liten grad av nytenkning. Nettbaserte aviser, digitale "en-til-mange" tv-sendinger og elektroniske bøker er bare noen av mange eksempler på dette. Dette er trekk man også tidligere har kunnet observere ved lanseringer av nye mediekanaler.

Innenfor området digitale bibliotek, vil det vi kan definere som et *formidlingsproblem,* også være en aktuell problemstilling. Det er ikke uvanlig å sitte med store mengder ulike samlinger, som hver har sine distinkte egenskaper. Det å skape en enhetlighet på tvers av disse samlingene, ved. f.eks. å plassere dem inn i det samme grensesnittet, og samtidig opprettholde en intuitiv kommunikasjon med brukeren, viser seg ofte å være en stor utfordring.

Denne avhandlingen tar for seg et forslag til arkitektur for et dynamisk digitalt bibliotek. Det tas utgangspunkt i at et digitalt bibliotek består av flere mulige grensesnitt, med flere mulige samlinger, og at interaksjonen mellom disse er en viktig faktor under oppbyggingen av et digitalt bibliotek.

1.1 Problemstilling og avgrensning

1.1.1 Bakgrunn og motivasjon

Forfatterens interesseområde innenfor fagfeltet digitale bibliotek er spesielt fokusert rundt HCI[1] og arkitekturer for digitale bibliotek som understøtter avanserte brukergrensesnitt.

Avhandlingens utgangspunkt ble til i forbindelse med prototyping av systemer for brukerdefinerte grensesnitt. Et utkast til et system basert på informasjonshåndtering på håndholdte maskiner (PDA) ble utprøvd i forbindelse med en større gjennomgang av mulige vinklinger for avhandlingen.

Utgangspunktet og motivasjonen, for dette arbeidet, var forfatterens personlige interesse den gang (desember 1999) for å finne et system som gjorde det mulig for enhver person som hadde en PDA[2], å få tilgang til TV-programmet via denne. Da disse første ideene til systemet ble til, fantes det ingen leverandør som kunne tilby akkurat denne type tjeneste, til tross for en økende etterspørsel i markedet etter slike tjenester[3].

En faktor som også hadde stor innvirkning på valg av utgangspunkt, var at et amerikansk firma, AvantGo **[W_AVA]** kort tid i forveien hadde lansert et system som gjorde det mulig å hente ned spesialformaterte websider, via internett, inn i en PDA - enten via en PC eller direkte med modem/mobiltelefon. Når disse sidene først var hentet ned, kunne man fritt bevege seg innenfor innholdet uten å være koblet opp mot tjenesteleverandøren.

En prototype ble på grunnlag av disse ideene implementert, og senere videreutviklet. Arbeidet med dette utkastet, og etter hvert en fungerende prototype, ble underveis generalisert ned til en generell arkitektur for digitale bibliotek.

1.1.2 Problemidentifisering

Utgangspunktet for denne avhandlingen er å komme frem til en arkitektur for digitale bibliotek som har følgende hovedegenskaper:

- ***Dynamiske og personifiserte grensesnitt***
  Grensesnittet skal ikke være fast. Brukeren skal ha mulighet til å ha en viss innflytelse på sitt eget brukergrensesnitt.
- ***Støtte ulike typer grensesnitt***
  Det skal være mulig å lage nye og fjerne eksisterende brukergrensesnitt, uten at det påvirker resten av systemet i stor grad.
- ***Dynamiske samlingsmengde***
  Et digitalt bibliotek basert på denne arkitekturen skal ikke være begrenset eller avhengig av et spesielt sett med samlinger. Nye samlinger skal kunne fjernes og legges til uten at det har stor innvirkning på stabiliteten til biblioteket.

1. HCI er en forkortelse for Human-Computer Interaction og omfatter bl.a design, evaluering og implementering av interaktive datasystemer for menneskelig bruk. **[PRE94]**
2. *"**P**ersonal **D**igital **A**ssistant"*, også populært referert til som *palm*, *pocketpc* og *lommepc*
3. Dette er basert på forfatterens deltagelse i PDA relaterte diskusjonsforum i tidsperioden

- ***Heterogene samlinger***
  Samlingene skal ikke være avhengige av én spesiell teknologi, eller behøve å være formet etter samme modell. Dette innebærer også at samlingen kan være ekstern i forhold til det digitale biblioteket.

1.1.3 Problemstilling

Ut ifra punktene fra problemidentifiseringen, kan problemstillingen som danner utgangspunkt for oppgaven formuleres slik:

> *"Hvordan bygge opp et digitalt bibliotekssystem for å legge til rette for dynamiske og medieuavhengige brukergrensesnitt, hvor brukergrensesnittet forholder seg til heterogene samlinger"*

1.1.4 Mål med arkitekturen

Forfatteren erkjenner at problemstillingen omfatter et stort omfang av mulige løsninger og krav som må tilfredstilles. Man vil dermed kunne regne det som en urimelig stor oppgave å få med alle aspekter av denne problembeskrivelsen inn en enkelt arkitektur, selv om man avgrenser seg til avhandlingens målsetning.

Arkitekturen som foreslås kan, på bakgrunn av dette, begrenses til å ta for seg tre konkrete hovedmål, og forsøke å ta for seg mulige løsninger på disse målene:

Mål 1: Grensesnittene
Det skal være lett å legge til og fjerne grensesnitt til en implementasjon, og arkitekturen skal understøtte at de samme dataene forholdsvis enkelt skal kunne fremvises på forskjellige grensesnitt og teknologiplattformer.

Mål 2: Samlingene
Det skal være lett å legge til og fjerne samlinger, i den grad dette er mulig uten å endre store mengder kode i en ferdig implementasjon.

Mål 3: Brukerne
Det skal i stor grad være mulig for brukeren å fritt velge et passende grensesnitt, og å være med på å bestemme innholdet og utseendet til dette grensesnittet.

Meningen med denne avgrensningen er ikke å sette begrensninger på arkitekturens funksjonalitet eller beskrivelse. Målet er å få frem konkrete løsninger på en avgrenset del av de totale problemområdene som finnes innenfor digitale bibliotek.

1.1.5 Mål med prototypen

En prototype som tar for seg nøkkelemner ved arkitekturen gjennomgås i avhandlingen. Utarbeidelsen av arkitekturen og implementeringen av prototypen har tidvis foregått parallelt. I mange tilfeller har arbeidet med arkitekturen og teorigrunnlaget påvirket endringer i prototypen. Andre ganger er det områder som har dukket opp under utviklingen som har påvirket deler av arkitekturen.

Det vil derfor nødvendigvis finnes forskjeller mellom teori og praksis, spesielt når man må ta hensyn til visse pragmatiske begrensninger. På grunnlag av dette settes to spesifikke begrensninger som sammen med arkitekturens målsetning (*kapittel 1.1.4*) definerer prototypens omfang:

Begrensning av mål 1: Grensesnittene
Prototypen konsentreres om to ulike grensesnitt.

Begrensning av mål 2: Samlingene
Flere ulike samlinger må benyttes, men kan implementeres i samme database

Begrensning av mål 3: Brukerne
Brukerne må bruke ett av to grensesnitt. Brukere kan for hver av samlingene velge hva slags type innhold som ønskes.

Disse begrensningene setter, i likhet med for arkitekturen, ikke begrensninger til hva som reelt er, eller kan bli, implementert i prototypen. Meningen er å avgrense hvilke deler av arkitekturen som prototypen illustrerer i avhandlingen.

1.1.6 Tilnærming

Problemstillingen gir anledning for flere mulige angrepsvinkler. I tillegg til å definere mål for arkitekturen, er det derfor viktig å se på hvilken angrepsvinkel som vil benyttes for å løse problemet.

Fuhr et. al. **[GCES01]** tar for seg aktørene som bidrar til forskning innenfor digitale bibliotek, hvilke bakgrunn de har, og hvilke deler av området de i hovedsak konsentrer seg om.

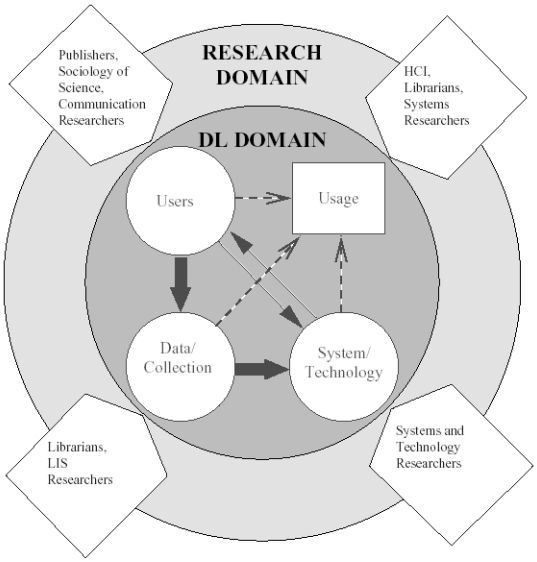

Fig 1-1 Aktørene, og inntresseområdene til disse innenfor digitale bibliotek **[GCES01]**

Figuren **[Fig 1-1]** identifiserer to hovedgrupper av aktører, som i seg selv har forskjellig syn og tilnærminger til digitale bibliotek. Disse gruppene vil begge ta med seg "sitt" syn fra sin fagkrets i alt fra oppbygning, til bruk, av et digitalt bibliotek.

- ***Forskermiljøet***
  Består av forskere innenfor områder som eksempelvis bibliotekforskning, informatikk og sosiologi.
- ***Brukerne***[4]
  Består eksempelvis av bibliotekarer og utgivere (publishers).

Fire tilnærminger identifiseres videre ut ifra en firedeling av kjernen i et digitalt bibliotek:

- ***Brukere*** (Users)
- ***System/teknologi*** *(System/Technology)*
- ***Bruk*** *(Usage)*
- ***Data/samlinger*** *(Data/Collection)*

Forskningen som Fuhr et. al. tar for seg vil også kunne sees på som ulike angrepsvinkler. Vi velger derfor her å snu litt om på denne definisjonen, og bruke denne oppdelingen som en mer generell identifikasjon på ulike tilnærminger til digitale bibliotek. Sett ut ifra modellen **[Fig 1-1]** vil denne avhandlingen derfor vektlegge spesielt **system/teknologi** og **brukere** og tilnærme seg problemstillingen i hovedsak ut ifra et teknisk forskerperspektiv.

1.1.7 Teoretisk basis

Avhandlingen baserer seg i stor grad på forfatterens egen forskning og utvikling innenfor digitale bibliotek, men det bygges også på teori og forskning innenfor digitale bibliotek.

I hovedsak kan teorien som ligger til grunn deles inn i følgende kategorier, og hovedkilder til informasjon:

- ***Arkitekturer for digitale bibliotek***
  - ⇢ SmartPush **[KUR99]**
  - ⇢ Daffodil **[FUHR00]**, **[FUHR02]**
  - ⇢ University of Michigan Digital Library **[BIR95]**, **[DUR97]**, **[WEI99]**
  - ⇢ North Carolina State University; MyLibrary **[MYL99]**
  - ⇢ Samlinger i distribuerte bibliotek **[CLDF98]**
  - ⇢ Arms, "An Architecture for Information in Digital Libraries" **[ARM97]**
  - ⇢ Aalberg/Hegna, "Arkitektur for digitale bibliotek" **[AAH00]**

4. I kilden benyttes termen "practitioners" (eng)

- ***Verk som omtaler digitale bibliotek***
  - ⇢ Kahn/Wilensky, Distribuerte digitale objekter **[KW95]**
  - ⇢ Definisjoner av digitale bibliotek **[SCH99]**, **[SDL98]**, **[WAT98]**, **[KW95]**
- ***Agenter og agentbaserte arkitekturer***
  - ⇢ Agentbaserte Arkitekturer **[BIR95][FUHR00]**
  - ⇢ Wooldridge/Jennings, Intelligente agenter **[WOO95]**
- ***Klassifisering og metadata***
  - ⇢ Cathro, "Metadata: An Overview" **[CAT97]**
  - ⇢ Fuhr et. al., Klassifisering og evaluering **[GCES01]**
  - ⇢ Bulterman, Adaptive hypermediapresentasjoner **[BUL98]**

Arbeidet bygger også rundt pågående forskningsarbeid **[AAL00]** og tidligere avhandlinger innenfor informasjonsforvaltning ved IDI/NTNU **[HOL98]**, **[LIC00]**, **[OPP00]**, **[KVA01]**.

1.2 Avhandlingens oppbygning

Avhandlingen er delt inn i følgende fire logiske deler:

- ***Litteraturanalyse***
- ***Forslag til arkitektur***
- ***Prototype basert på arkitekturen***
- ***Diskusjon og evaluering***

1.2.1 Litteraturanalyse

Kapittel 2 og 3 tar for seg aktuelle teorier som ligger til grunn for emnene som behandles i avhandlingen.

I *Kapittel 2* behandles noen sentrale begrep som benyttes i avhandlingen. For hvert av de utvalgte begrepene belyses aktuell teori innenfor emnet, og en definisjon av avhandlingens bruk av hvert enkelt begrep avklares.

Kapittel 3 tar for seg state-of-the art innenfor digitale bibliotek, med vekt på eksisterende arkitekturer. Det redegjøres her for egenskaper og særtrekk ved fire ulike arkitekturer for digitale bibliotek.

1.2.2 Forslag til arkitektur

Kapittel 4 tar for seg en mulig arkitektur for digitale bibliotek. Arkitekturen illustreres ved hjelp av en konkret modell (ref. side 38) som står veldig sentralt i beskrivelsen av komponentene i arkitekturen.

1.2.3 Prototype basert på arkitekturen

Kapittel 5 behandler implementeringen av en prototype basert på teorien og modellen gjennomgått i *kapittel 4*. Nøkkelelementene i prototypen (ref. kapittel 1.1.5, *"Mål med prototypen"*) beskrives her i detalj.

Kapittelet 5 inneholder også et case som illustrerer prototypens funksjon og virkemåte (ref. *kapittel 5.7*).

1.2.4 Diskusjon og evaluering

Kapittel 6 tar for seg en oppsummering av avhandlingen, og erfaringer med arkitekturen. Kapittelet setter den foreslåtte arkitekturen (kapittel 4) opp imot arkitekturene som gjennomgås i kapittel 3.

Kapittel 7 tar for seg evaluering av avhandlingen og peker på videre forskning innenfor emnet.

2 Terminologi

Dette kapittelet tar for seg terminologi som er i bruk innenfor digitale bibliotek. Spesiell vekt legges på emner som benyttes i stor grad i avhandlingen. Det anbefales sterkt å sette seg inn i referansematerriellet for en dypere forståelse av de enkelte begrepene, da dette kapittelet i stor grad bare må regnes som en introduksjon til terminologien som benyttes i avhandlingen.

De emnene som belyses her er:

- ***Digitale bibliotek***
- ***Brukergrensesnitt***
- ***Dynamiske grensesnitt og dynamisk informasjon***
- ***Forholdet mellom arkitektur, modell og prototype***
- ***Metadata***
- ***Samlinger, datalager og databaser***
- ***Agenter og agentbaserte arkitekturer***

Det sees på bruken av disse begrepene i publisert litteratur, med vekt på å beskrive den forståelsen av begrepene som avhandlingen bygger på.

2.1 Begrepet digitale bibliotek

Digitale bibliotek (heretter ofte forkortet DL etter *Digital Libraries*[5]) er et forholdsvis vidt begrep, og benyttes i mange forskjellige sammenhenger, med til tider sprikende definisjon. En definisjon som passer alle sammenhenger hvor DL benyttes er derfor vanskelig å komme frem til.

Fuhr et. al. **[GCES01]** peker på at forskningsretningene og grupper av brukere av DL hver har sine egne måte å forholde seg til et DL på. Brukerene fokuserer i stor grad på de aspekter av DL som er i tråd med sitt fagfelt.

5. Vi unngår med dette sammenblanding med utrykket "database" (som ofte forkortes DB).

Vi tar her for oss to mulige definisjoner av begrepet digitale bibliotek. Ut i fra dissek definisjonene konkluderes det med hva som legges til grunn for bruken av utrykket i avhandlingen.

2.1.1 Digital Library Federation

En foreleser ved Simmons College sendte i 1999 sine studenter ut på leting etter definisjoner og satte opp en liste med 60 mulige definisjoner på "digitalt bibliotek" **[SCH99]**. En av de mest populære av disse definisjonene, blant de 10 elevene som deltok, var definisjonen medlemmer av Digital Library Federation **[W_DLF]** kom frem til i 1998 som basis for sin felles forståelse av digitale bibliotek:

> *"Digital libraries are organizations that provide the resources, including the specialized staff, to select, structure, offer intellectual access to, interpret, distribute, preserve the integrity of, and ensure the persistence over time of collections of digital works so that they are readily and economically available for use by a defined community or set of communities."* **[W_DLF]**

Denne definisjonen gjennomgås i detalj i en CLIR artikkel av Donald J. Waters **[WAT98]**[6]. Waters legger i sin artikkel spesielt vekt på bredden av begrepet, siden det griper om veldig mange mulige definisjoner av utrykket. Spesielt verdt å merke seg er det sentrale kravet om preservering og persistens over tid, funksjoner som kan være spesielt vanskelige å gjennomføre. Det samme gjelder kravet om at et digitalt bibliotek skal ha en eller flere definerte målgrupper.

2.1.2 D-Lib

I forbindelse med en workshop holdt av D-Lib om "Digital Library Metrics" **[SDL98]** kom det frem en annen mulig definisjon av digitale bibliotek.

Gruppen valgte å ta for seg begrepet som et sett av særtrekk:

> *"Samlingen av tjenester [1] og samlingen av informasjonsobjekter [2] som understøtter brukernes interaksjon med informasjonsobjekter [3] og organiseringen og presenteringen av disse objektene [4] tilgjengelig, enten direkte eller indirekte [5] via elektroniske/digitale metoder. [6]."* ***[SDL98]***[7]

Samlingen av tjenester [1]
Et digitalt bibliotek er mye mer enn bare en samling av data. Det skal tilby et spekter av tjenester til alle sine brukere (både mennesker og maskiner, og de som produserer, forbruker og administrerer informasjonen). En samling av tjenester finnes som en essensiell del av et digitalt bibliotek, i form av f.eks. administrering av samlinger, tjenester som står for stabil og kvalitetssikret lagring og tjenester som tar for seg navnhåndtering og lokalisering.

6. Artikkelen anbefales for en grundigere gjennomgang av definisjonen.
7. Fritt oversatt fra engelsk: "The collection of services and the collection of information objects that support users in dealing with information objects and the organization and presentation of those objects available directly or indirectly via electronic/digital means."

Samlingen av informasjonsobjekter [2]
Basisen for et digitalt bibliotek er informasjonsobjektene som danner grunnlaget for innholdet. Et elementært særtrekk for digitale bibliotek er at informasjonsobjekter finnes i samlinger med tilhørende administrative støttefunksjoner. Typen informasjon varierer fra tradisjonelle "dokumenter", multimedieobjekter (lyd, bilde), løpende oppdatert informasjon (f.eks. statistikk, måleresultater og annen informasjon som forandrer seg fortløpende) til dynamiske søkeresultater.

Understøtting av brukernes interaksjon med informasjonsobjekter [3]
Målet med digitale bibliotek er å assistere brukere. Dette kan oppnås ved å tilfredsstille brukernes behov og krav i forhold til administrering, aksessering, lagring og manipulering av materialet i biblioteket. Brukere kan være mennesker eller automatiserte prosesser som handler på vegne av eller er støttet av menneskelige behov. Brukere varierer også, fra å være endebrukere ("forbrukere"), bibliotekarer, til produsenter (f.eks. forfattere) som ønsker materiale gjort tilgjengelig via biblioteket.

Organisering og presentering av informasjonsobjekter [4]
For å få best og mest effektiv organisering i biblioteket er det viktig å presentere og sette opp gode strukturer som brukere forstår og enkelt kan dra nytte av. I tradisjonelle bibliotek blir bøker gjerne sortert ut ifra kriterier som tittel, forfatter, emne og dato - og gjort tilgjengelig ved skilter til rett rom, seksjon og hylle, helt ned til merke bak på ryggen til den aktuelle boken.

Tilgjengelighet, enten direkte eller indirekte [5]
Informasjonen i et digitalt bibliotek kan være presentert som rene digitale objekter. Presentasjonen kan også være ved hjelp av andre media, som for eksempel papir, men da representert i biblioteket på digital form (f.eks. metadata). Informasjonen kan videre være tilgjengelig enten direkte over nettverk (f.eks. et dokument som er direkte nedlastbart via biblioteket), eller indirekte (f.eks. ved at biblioteket gir informasjon om hvor dokumentet finnes eller hvordan det kan aksessenes, men at selve dokumentet er utenfor rammen for biblioteket).

Tilgjengelighet via digitale/elektroniske metoder [6]
For at et objekt skal være del av et digitalt bibliotek må det være en elektronisk representasjon av informasjonen på et eller annet vis. Dette kan være alt fra metadata om et objekt som ikke er tilgjengelig elektronisk, metadata om et elektronisk dokument utenfor rammen for biblioteket, dokumenter som er tilgjengelige både elektronisk og i for av andre medier, til rene digitale, direkte tilgjengelige dokumenter.

2.1.3 Avhandlingens definisjon av digitale bibliotek

Digital Library Federation setter ganske strenge krav i sin beskrivelse av hva et DL skal inkludere. Blant annet kan definisjonen tolkes til at DL er organisasjoner (eng: "organizations"), og på den måte at et DL kun er en "digital utgave" av et tradisjonelt bibliotek. Definisjonen til D-lib tar et mer teknisk utgangspunkt, og definerer i større grad de *egenskaper* et digitalt bibliotek skal ha.

Til tross for ulikhetene, danner de to definisjonene grunnlaget for det avhandlingen legger i digitale bibliotek. De gir rom for et stort spekter av mulige tjenester og bruksområder, samtidig som de setter en del viktige krav til tilgjengelighet, organisering, definerte brukergrupper og interaksjon.

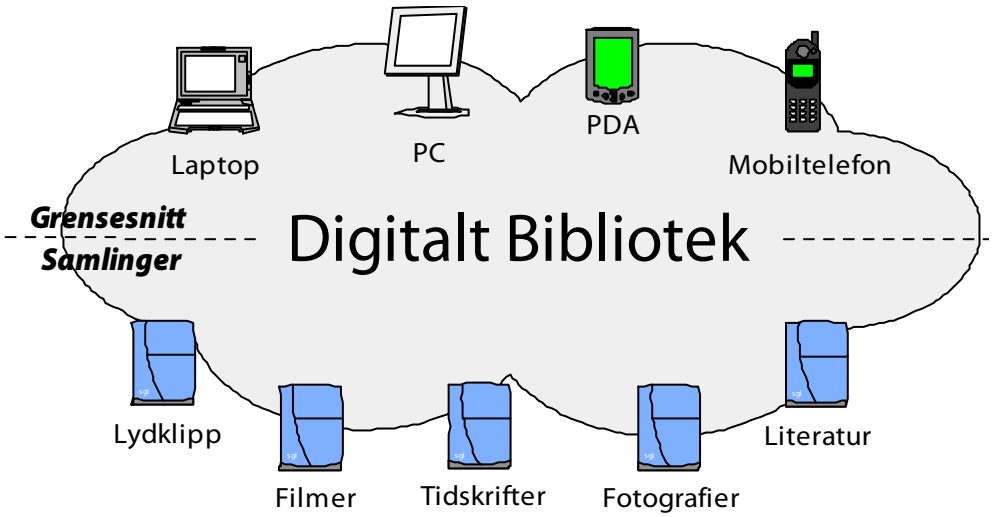

Fig 2-1 Noen mulige grensesnitt og samlinger i et digitalt bibliotek

Figuren **[Fig 2-1]** illustrerer både mangfoldet av samlinger og grensesnitt som vil kunne være del av et digitalt bibliotek innenfor avhandlingens definisjon av begrepet.

2.2 Brukergrensesnitt

Begrepet *brukergrensesnitt* (ofte kortet ned til *grensesnitt*) blir mye benyttet i avhandlingen. Jenny Preece definerer *user interface* slik:

> *"[...] the totality of surface aspects of a computer system, such as its input and output devices, the information presented to or elicited from the user, feedback presented to the user, the system's behaviour, its documentation and associated training programmes, and the user's actions with respect to these aspects"* ***[PRE94]***.

Brukergrensesnitt i en eller annen form finnes overalt hvor en person forholder seg til et eller annet datasystem. Spesielt i forhold til brukergrensesnitt på datamaskiner har man pr. dags dato kommet såpass langt at endel kunnskaper regnes som ganske grunnleggende innenfor forholdet mellom bruker og grensesnitt. Dette gjelder blant annet hvordan man beveger markøren ved hjelp av en mus, eller hvordan man starter applikasjoner i de mest kjente operativsystem. Det kan derfor være nyttig å kunne dra begrepet litt lenger, og definere forholdet mellom bruker og grensesnitt som en *pågående kommunikasjon mellom menneske og maskin*.

Når det kommer til nettbasert informasjonsutveksling (kommunikasjon) og distribusjon, er det i skrivende stund konseptet med en webleser (eng: *browser*), og tilgangen denne har til en lokal eller ekstern tjenesteleverandør (normalt en webserver) som dominerer området. Weblesere har dermed raskt blitt en av de i dag mest brukte og kanskje viktigste formene for interaktive brukergrensesnitt, både for kommunikasjon og i økende grad også grensesnitt til applikasjoner som tidligere lå på operativsystem-nivå.

Brukere av webgrensesnitt inkluderer dermed en stor gruppe mennesker. Disse kan man anta har et stort spekter av varierende krav og behov. Ved hjelp av en webbrowser som per i dag er forholdsvis standardisert, har brukerne prinsipielt mye av de samme mulighetene, uavhengig av maskinens software (eksempelvis Microsoft Windows, Mac OS, PalmOS og Linux) og hardware (f.eks. personlig datamaskin, TV, PDA eller mobiltelefon).

Avhandlingen fokuserer til tider mye på noen spesielt utvalgte brukergrensesnitt; det legges spesielt stor vekt på grensesnitt innenfor områdene *WEB* (PC med browser) og *PDA* (lomme-pc), og disse brukes i stor grad i flere eksempler, og i prototypen. Det er imidlertid viktig ikke å se seg blind på dette relativt snevre utvalget av brukergrensesnitt. Teorien til Preece **[PRE94]** omfatter alt fra tastaturet vi skriver på, markøren på skjermen, kontrollpanelet på skriveren til telefonsamtaler med automatiserte tjenester og betalingsterminalen man finner i butikker.

2.3 Forholdet mellom arkitektur, modell og prototype

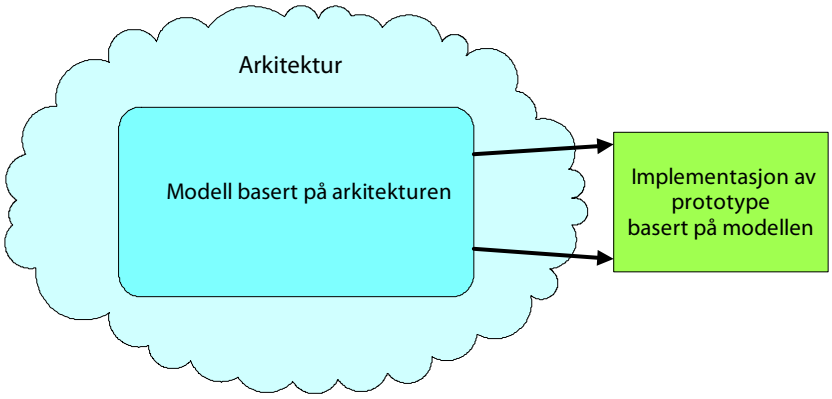

Fig 2-2 Forholdet mellom arkitektur, modell og prototype

Arkitektur, modell og **prototype** (eventuelt også "implementert system") er tre benevnelser som brukes i stor grad i avhandlingen, og som det er viktig å holde fra hverandre.

Figuren **[Fig 2-2]** illustrerer forholdet mellom disse begrepene. Begrepene kan gi ulike assosiasjoner, alt etter hvilken bakgrunn man har. Det redegjøres derfor her kort for bruken av disse begrepene.

Arkitekturen beskriver alt som er relatert til avhandlingens forslag til oppbygging av et digitalt bibliotek. Dette er ment å være en litt idealistisk og generell beskrivelse av hva som inngår i et digitalt bibliotek.

Modellen refererer til en spesifikk modell/figur (i hovedsak henviser dette til **[Fig 4-2]** som er beskrevet i *kapittel 4*), som sees på som *en mulig* representasjon av arkitekturen. Det legges dermed opp til at det vil kunne finnes flere mulige måter å modellere arkitekturen på, men at det i de sammenhenger der det refereres til "modellen", kun er denne representasjonen vi forholder oss til.

Prototypen (også løst referert til som "systemet") benyttes for å beskrive en *eksisterende* eller en *mulig* implementert løsning, basert på arkitekturen og den aktuelle modellen.

Bruken av disse utrykkene er spesielt knyttet til *kapittel 4* og *kapittel 5* i avhandlingen. Unntak som avviker fra disse beskrivelsene kan forekomme, men det er i de aller fleste tilfeller, ut ifra sammenhengen, mulig å skille ut disse.

2.4 Dynamisk informasjon og dynamiske grensesnitt

Begrepene *dynamisk informasjon* og *dynamiske grensesnitt* er en viktig del av denne avhandlingen, da mye av grunnlaget for problemstillingen går på hvordan man kan utnytte dynamikken i informasjon for å forbedre grensesnitt.

2.4.1 Informasjon og data

I sin enkleste form kan et eksempel på dynamisk informasjon, være informasjonen man kan avlese fra en klokke. Den kan vise 18:45:01 et øyeblikk, og neste sekund 18:45:02. På samme vis kan man si at meningen med begrepene *nå* og *i dag*, i mange situasjoner, er helt dynamisk, og at meningen forandrer seg etter sammenhengen utrykket benyttes i.

I følge Bulterman **[BUL98]** er en av de store fordelene - kanskje den eneste store fordelen - med online elektronisk publisering og presentasjon, at innholdet ikke er statisk av natur. Data kan manipuleres, endres og utgis mye enklere og rimeligere enn tradisjonelle "statiske" publiseringsmetoder som bøker og tidsskrifter, og det er mye lettere å hoppe mellom de enkelte delene av informasjonen. Man kan også introdusere metoder på informasjonen (søking, sammenlikning, oversetting og liknende).

Bulterman presiserer videre at online informasjon gjerne er preget av en tidsmessig presishet[8]. Informasjonen kan hentes direkte ved behov, og kan fra informasjonsleverandøren sin side oppdateres når leverandøren finner det fornuftig eller nødvendig. Et godt eksempel her er forskjellen mellom *nettaviser* og tradisjonell presse. Hver gang en nyhet dukker opp i løpet av døgnets 24 timer, blir nettavisen oppdatert kontinuerlig og øyeblikkelig. Papiravisen, på sin side, må nøye seg med nyhetsbildet slik det sto akkurat da avisen gikk i trykken. Hvis det oppdages en trykkfeil kan dette opplyses om først i neste utgave av papiravisen. For nettavisen sin del kan den feilaktige artikkelen endres eller erstattes for godt i løpet av minutter.

2.4.2 Grensesnitt

Det som også preger online informasjon er *selektivitet*. Brukeren kan velge seg det subset av informasjonen som dekker det behovet brukeren har, og informasjonsleverandøren kan bestemme seg for slags klasser av informasjon de ønsker å tilby til de enkelte klientene. I senere år har slike personifiserte (eng: *personalized*) og adaptive grensesnitt blitt mer og mer vanlige. Et personifisert grensesnitt tar hånd om bruke-

8. Bulterman benytter uttrykket *timeliness* (eng.)

rens ønsker, mens adaptive grensesnitt lærer fra brukeren. **[CHA99]**. Begge disse metodene går under det avhandlingen klassifiserer som dynamiske grensesnitt, men det legges større vekt på den personifiserte delen av begrepet.

2.4.3 Avhandlingens bruk av utrykket

Avhandlingen behandler det meste som dreier seg om data og informasjon som om det har det i sin natur å være dynamisk. Gitt lang nok tid, er de fleste typer informasjon gjenstand for en eller annen form for revisjon. Tar man utgangspunktet i dette, er målet at man vil kunne bedre levetiden og øke den generelle kvaliteten på informasjonssystemer.

2.5 Metadata

Ordet metadata dukket opp i internett-samfunnet rundt 1995, og har spredt seg raskt, uten at man har hatt en felles forståelse for betydningen av ordet **[AAH00]**. Det er derfor nødvendig med en kort redegjørelse av hva avhandlingen legger i begrepet.

Metadata er i flere tekster kun kort definert som ***data om data*** (se **[OPP00]**, **[AAH00]**). En mulig utvidet forklaring på dette er at man benytter et sett med elementer til å beskrive et dokument eller objekt, med den hensikt at man dermed lett skal finne tilbake til objektet **[OPP00]**. Andre benytter lange og relativt forskjellige forklaringer på det samme emnet, alt etter hvilken sammenheng og i hvilken ramme begrepet benyttes.

Det finnes en mengde publisert materiale som tar for seg begrepet metadata. I og med at metadata ikke regnes som et hovedemne i denne avhandlingen, nøyer vi oss her med å referere til noen få relevante definisjoner, og definerer avhandlingens bruk av ordet.

2.5.1 To definisjoner på metadata

Aalberg/Hegna **[AAH00]** (side 41.) definerer metadata slik:

> *"[...] en formell beskrivelse av indre og ytre karakteristika ved tradisjonelle og digitale dokumenter og objekter som understøtter formidlingen av dem (dokumenter og objekter) til personer"*

Cathro **[CAT97]** definerer et metadataelement som:

> *"[...] noe som beskriver en informasjonsressurs, eller som forenkler tilgangen til en informasjonsressurs" (fritt fra eng.).*

2.5.2 Metadata om samlinger av data

Cathro **[CAT97]** utdyper sin definisjon ved å peke på følgende:

> *"Metadata can be an information resource in its own right. For example, a review of a film, which on one level is a piece of metadata related to the film, is,*

on another level, a literary work with its own author and perhaps its own intellectual property constraints." ***[CAT97]***

I motsetning til definisjonen til Aalberg et. al., ønsker vi her ikke å begrense metadatabegrepet til kun å innbefatte objekter og dokumenter i samlinger. Begrepet vil i steden betegnes som beskrivende informasjon om ethvert informasjonsobjekt, eller samling av informasjonsobjekt, i tråd med definisjonen til Cathro.

2.6 Samlinger, databaser og datalager

Samlingsbegrepet står sentralt i de fleste publikasjoner som omhandler både tradisjonelle og digitale bibliotek.

På samme måte som ved definisjonen av et *digitalt bibliotek*, tilordnes også en *samling* ulik definisjon blant annet ut ifra hvilket fagfelt man tar utgangspunkt i **[GCES01]**. Samtidig er det lett å oppnå en stor grad av begrepsforvirring når man i tillegg tar inn de mer tekniske begrepene *database* og *datalager*.

I sin enkleste form kan en **samling** bestå av et sett indekserte dokumenter lagret i en **database** i et **datalager**.

Avhandlingen benytter følgende grunndefinisjon:

- ***Samling***
  Et sett med indekserte dokumenter som har et felles tema eller en egenskap som gjør at de hører "naturlig" sammen.
- ***Database***
  Lagringsmetoden som benyttes for samlingen (eksempelvis MySQL ***[W_SQL]****). En samling kan befinne seg over flere databaser.*
- ***Datalager***
  Samlingsbetegnelse på all datalagring innenfor et digitalt bibliotek. Et datalager kan bestå av flere databaser og samlinger.

Lagoze/Fielding **[CLDF98]** tar for seg begrepet samling (eng: collection) innenfor distribuerte digitale bibliotek. Figuren **[Fig 2-3]** er hentet fra den aktuelle artikkelen og definerer en arkitektur for distribuerte digitale bibliotek.

Arkitekturen legger grunnlag for én måte å definere samlinger på, uavhengig av hvor selve dataene lagres fysisk. En samling er i arkitekturen til Lagoze/Fielding uavhengig av datalager (eng: *repository*).

Definisjonen utvides derfor - i de tilfeller det er snakk om distribuerte systemer, til at en samling også kan være distribuert over flere ulike datalager, og at et digitalt bibliotek kan inneholde datalager som biblioteket har kontroll over selv (eksterne datalager)

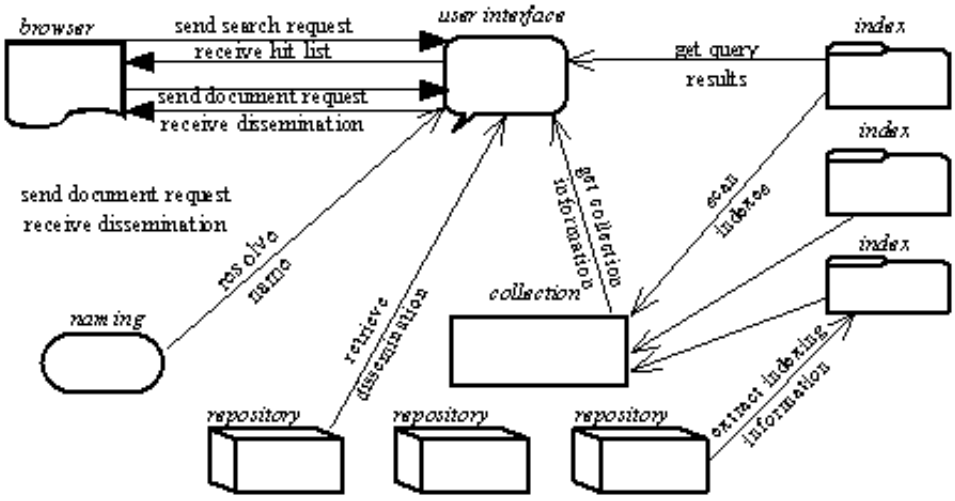

Fig 2-3 Samlinger innenfor distribuerte digitale bibliotek **[CLDF98]**

2.7 Agenter og agentbaserte arkitekturer

Agenter står sentralt i flere av arkitekturene som avhandlingen tar for seg. Agentbegrepet benyttes i korte trekk for å beskrive innkapslede og uavhengige prosesser innenfor et system. Vi vil her se på definisjoner av *agenter* og *agentbaserte arkitekturer*.

2.7.1 Agenter

Birmingham **[BIR95]** definerer agenter ut ifra to viktige egenskaper som skiller dem fra andre prosesser i et system, **autonomi** og **kommunikasjon**. Wooldridge/Jennings **[WOO95]** utvider dette med to andre egenskaper, **reaksjonsevne** og **initiativ**. Fuhr et. al. **[FUHR00]** definerer utover de overnevnte egenskapene to ytterligere egenskaper, som inngår i agenter innenfor en agentbasert arkitektur. Disse er **intelligens** og **adaptivitet**.

Autonomi (*autonomy*)
Agenten representerer både egenskaper og oversikt over hvordan disse kan benyttes. Agenter kan derfor selv vurdere om egenskapene skal benyttes for et gitt tilfelle. En agent behøver derfor ikke nødvendigvis gjøre alt den blir bedt om, men bare det den har egenskaper til å utføre. Denne vurderingsevnen er noe tradisjonelle systemer mangler **[BIR95]**.

Kommunikasjon (*negotiation*)
Siden agenten er autonom, må den selv kunne kommunisere med andre agenter for å få tilgang til andre ressurser og muligheter.

Reaksjonsevne (*reactiveness)*
En agents reaksjonsevne går på dens mulighet til å reagere når den blir kontaktet av andre agenter.

Initiativ (*proactiveness*)
En agent har mulighet til å ta initiativ selv, f.eks. når den oppdager forandringer i sitt miljø som krever en reaksjon.

Intelligens (*intelligence*)
En agent må sitte på kunnskap som den kan benytte til å trekke konklusjoner og ta beslutninger.

Adaptivitet (*adaptiveness*)
En agent kan endre oppførsel i forhold til hva slags miljø den befinner seg i.

2.7.2 Agentbaserte arkitekturer

Fuhr et. al. **[FUHR00]** definerer utrykket *agentbaserte arkitekturer* (ABA[9]) i forbindelse med arkitekturen gruppen benytter i sitt forskningsarbeid. Arkitekturen fokuserer på IR[10], og benytter seg i stor grad av agenter for å utføre nødvendige oppgaver.

Det refereres forøvrig til kapittel 3.2, *"Daffodil"* for en detaljert beskrivelse av denne arkitekturen.

2.7.3 Avhandlingens definisjon

For de enkelte arkitekturene som blir gjennomgått i kapittel 3, refereres det til definisjonen hver av disse kildene selv setter om agenter. Utover dette blir både begrepet *agenter* og *agentbaserte arkitekturer* benyttet om henholdsvis *prosesser* og *arkitekturer* som antas å ha egenskapene Fuhr et. al. **[FUHR00]** definerer.

9. *"Agent Based Architectures"* (eng.), *Wooldridge/Jennings* **[WOO95]**
10. "Information Retrival" (eng.)

3 Arkitekturer for digitale bibliotek

Tradisjonelt har bibliotek vært konsentrert rundt utvalg og lagring av samlinger med bøker og journaler **[MOD99]**. Dette spesielt fordi bøker og journaler i en årrekke var den primære manifisteringen av data, informasjon og kunnskap. Som en følge av dette brukte bibliotekarene mesteparten av sin tid på administrasjon av bok og journalsamlinger.

Nå, med oppblomstringen av et globalt nettverk av datamaskiner, er informasjonen oftere og oftere tilgjengelig også i digital form. I økende grad er informasjonen også *born digital* **[RAU01]**, det vil si at det ikke stammer fra et annet, gjerne "fysisk", medium. Informasjonen er derfor ikke nødvendigvis tilgjengelig i noen annen form, og kan derfor være ganske sårbar for faktisk å forsvinne helt om ingen tar hånd om dem på samme måte som de trykte mediene er blitt bevart i alle år.

Bibliotekene fortsetter selvsagt støtten av papirbaserte samlinger, men må på samme tid også tilpasse seg til de nye mediene og teknologiene, for å være i stand til å oppbevare og bearbeide også digitale samlinger. Det er heller ikke bare biblioteker som sitter med behov for å ta hånd om informasjon. Andre offentlige og private enheter vil også ha behov for å kunne ta vare på digitale dokumenter og samlinger for fremtiden.

Dette kapittelet tar for seg noen utvalgte arkitekturer for digitale bibliotek. De individuelle målene til disse kan være sprikende, men felles for dem alle er at de prøver å bøte på noen av problemene og utfordringene som digitale bibliotek fører med seg. I tråd med avhandlingens mål, er arkitekturene valgt ut ifra at de enten muliggjør medieuavhengige samlinger, brukertilpassede grensesnitt eller begge deler.

Som basis for studiet er fire aktuelle arkitekturer for digitale bibliotek valgt ut:

- ***SmartPush***
  - en arkitektur for nyhetspublisering med personifisert innhold
- ***DAFFODIL***
  - en agentbasert arkitektur for heterogene samlinger
- ***UMDL***
  - en agentbasert arkitektur fra University of Michigan
- ***MyLibrary***
  - en brukersentrert arkitektur

Det redegjøres for mål og virkemåte for hver av disse, og om arkitekturen har en fungerende prototype beskrives denne. Det vil oppsummeres til slutt rundt likhetene og ulikhetene mellom arkitekturene.

3.1 SmartPush

Internet og online publisering har over årene endret reglene for profesjonell publisering **[KUR99]**. Tidligere var den største flaskehalsen for nyhetspublisering det man kunne få plass til på et gitt antall sider i en avis eller tidskrift. Slike fysiske begrensninger i medieformatet er ikke lenger nøkkelfaktorer i publiseringsprosessen. Mulighetene i dag for å lage mer innhold er derfor i mye større grad tilstede, men utgiverne trenger metoder for personifisert distribusjon av dette innholdet.

Fig 3-1 SmartPush: Agenter i innholdsleveransen.

Kurki et. al. **[KUR99]** peker på det fundamentale spørsmålet innenfor innholdsfiltrering: *om en skal tilby et gitt dataelement til den aktuelle brukeren eller ikke.*

SmartPush prosjektet **[KUR99]** tar for seg en mulig løsning på dette problemet. Prosjektet definerer en arkitektur for personifisert innholdslevering, ved hjelp av agenter og semantisk metadata **[Fig 3-1]**.

I korte trekk fungerer systemet slik at etter innholdet som skal publiseres er laget, legges det til, av innholdsleverandøren, utfyllende informasjon om innholdet (i form av metadata). Disse metadataene brukes i sin tur for å filtrere informasjonen slik at endebrukere får en personifisert informasjonsflyt.

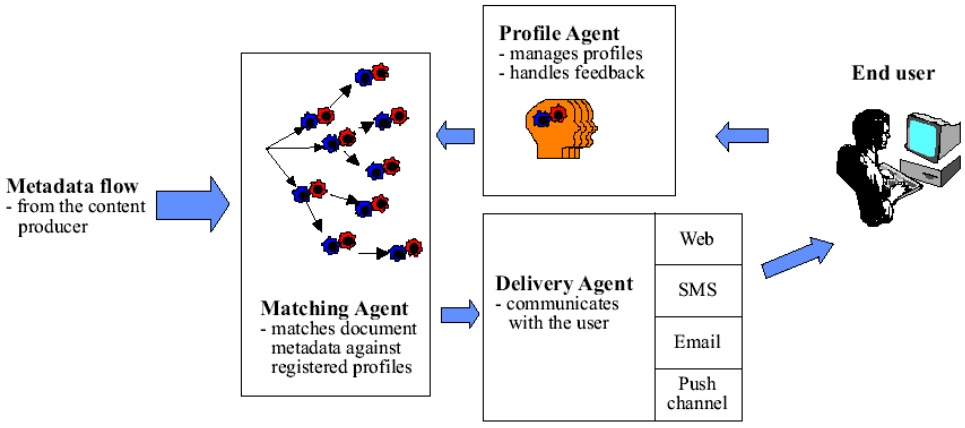

Fig 3-2 SmartPush: Oversikt over arkitekturen

Hovedkomponentene i arkitekturen **[Fig 3-2]** er:

- ***Profile agent***
  Lagrer profiler og brukerinformasjon
- ***Matching agent***
  Kobler dokument-metadata med brukerprofiler
- **Delivery agent**
  Kommuniserer med brukeren (end user)

Profile agent
Profilagentens primære oppgave er å lagre profiler og brukerinformasjon. Når en bruker tar i bruk systemet opprettes en profil tilhørende denne brukeren som inneholder informasjon om hva brukeren er interessert i av informasjon og hvordan denne informasjonen skal formidles tilbake til brukeren.

En slik profil kan lages på flere mulige måter. En mulighet er å få brukeren til å evaluere et sett med dokumenter, og generere en profil ut i fra dette. Et annet alternativ er å enten la brukeren velge blant generelle profiler, eller selv beskrive en profil ved hjelp av et sett med stikkord.

Hvordan informasjonen skal leveres brukeren må også defineres. Profilen inneholder derfor informasjon om hva som skal skje når interessant materiale dukker opp. Dette kan være å gi beskjed umiddelbart via e-post, eller å samle et sett med interessante informasjonsobjekter, som brukeren har tilgang til via et web-grensesnitt.

Matching agent
En "koblingsagent" tar for seg informasjonen som kommer inn, og kobler denne opp med de enkelte profilene. Metoden som benyttes defineres av Kurki et. al. **[KUR99]** som en asynkron avstandsmåling mellom dokumentet og profilen. Profilen definerer brukerens interesser i et hierarki - og en rangering av relevans vil dermed begrense seg til de deler av profilen som er relevante for dokumentet.

Delivery Agent
Denne agentens oppgave er å formidle ønsket informasjon til brukeren basert på hva slags leveringform som er definert i brukerens profil. Agenten åpner også for en begrenset kommunikasjon med brukeren. Eksempelvis kan dette for brukeren være å sende en mobil-melding (SMS) til agenten og få tilbake en kort emneliste, som videre kan utdypes ved nye meldinger.

3.2 Daffodil

Daffodil[11] er et digitalt biblioteksystem utviklet av forskere ved University of Dortmund i Tyskland, med forskningsstøtte fra German Science fundation.

Arkitekturen defineres av forskerene som:

> *"[...] a digital library system targeting at strategic support during the information search process"* ***[FUHR02]***

*11. "**D**istributed **A**gents **f**or User-**F**riendly Access **o**f **Di**gital **L**ibraries", **[FUHR02]***

Daffodil baserer seg på tidligere forskning av Fuhr et. al. **[FUHR00]**, som kom frem til en arkitektur for digitale bibliotek basert på agenter. Fokuset for arkitekturen er integrering av heterogene samlinger. Prototypen har som mål å tilby et rikt sett med funksjoner, over en heterogen mengde digitale bibliotek. I sin nåværende utgave integreres 10 ulike digitale bibliotek i prototypen. **[FUHR02]**

3.2.1 Arkitekturen

De fleste av dagens digitale bibliotek er frittstående informasjonssystemer som med dokumenter i digital form **[FUHR00]**. Til tross for et økende antall digitale bibliotek, vil de innenfor mange emner bare dekke en liten fraksjon av informasjonen som er tilgjengelig, så brukere blir nødt til å kombinere informasjon fra både digitale og tradisjonelle bibliotek.

Innenfor andre emner vil ulike digitale bibliotek tilsammen ha en så stor dekningsgrad, at man ikke lenger vil behøve å involvere tradisjonelle bibliotek. I en slik situasjon vil søk i mange ulike digitale bibliotek etterhvert bli en uakseptabel løsning for brukere. Søk må foretas på hver av tjenestene; utvidet søk vil kun være mulig innenfor de samlingene som det aktuelle bibliotek har tilgjengelig og data om de samme dokumentene kan være spredt utover flere digitale bibliotek.

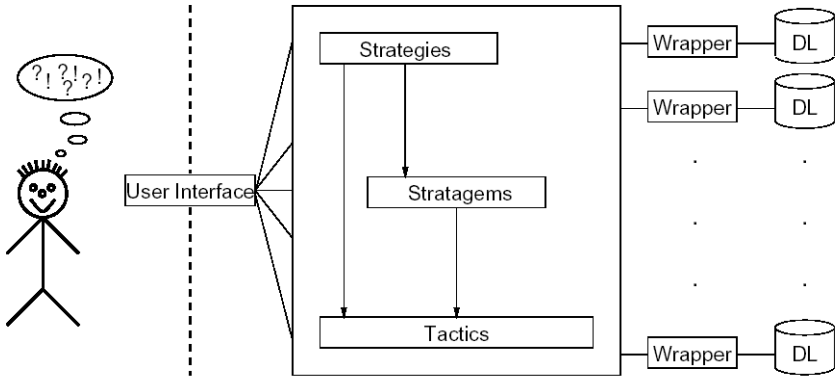

Fig 3-3 Daffodil: Arkitektur **[FUHR00]**

Løsningen Fuhr et. al. **[FUHR00]** foreslår er en agentbasert arkitektur for søk i det som betegnes som "forenede digitale bibliotek"[12]. Arkitekturen baserer seg på bruk av *høynivå* søkeaktiviteter.

Fuhr et. al. definerer fire klasser av søkeaktiviteter:

- ***Handling*** *(move)*
  Definerer en enkelt handling (f.eks. skrive et ord i en boks eller trykke på en link i en web-browser).
- ***Taktikk*** *(tactic)*
  En eller flere handlinger som foretas for å gjøre et søk.

12. Eng: *"Federated Digital Libraries"*

- ***"Knep"*** *(strategem)*
  Et komplekst sett med aktiviteter, basert på flere taktikker innenfor et gitt område. Ofte vil et informasjonsbehov bli tilfredstillt ved hjelp av et "knep".
- ***Strategi*** *(strategy)*
  En strategi består av en komplett plan for å tilfredstille et informasjonsbehov. Det består derfor ofte av flere "knep".

Fuhr et. al. **[FUHR02]** deler disse søkeaktivitetene inn i lavnivå (handling) og høynivå (taktikk, knep og strategi). Mens typiske informasjonssystemer kun støtter handlinger, legger Daffodil opp til å støtte høynivå-søkeaktiviteter, i hovedsak på "knep" (strategem) nivå. Figuren **[Fig 3-3]** viser den overordnede arkitekturen i Daffodil.

3.2.2 Prototype

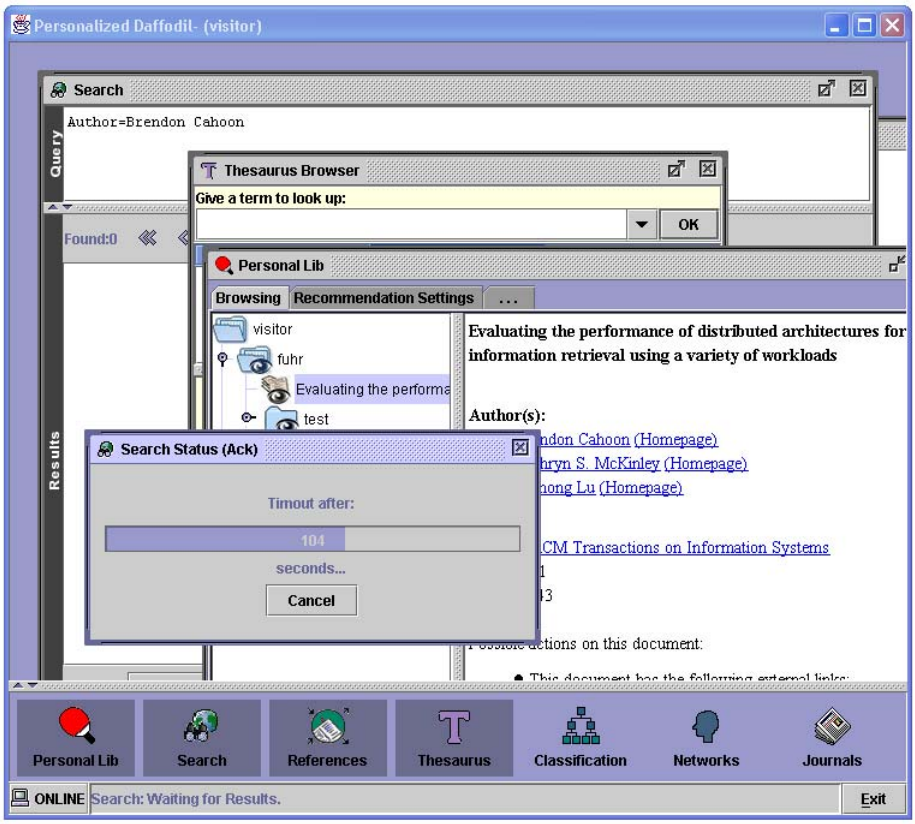

Fig 3-4 Daffodil: Prototype (Java)

Prototypen av Daffodil består av et grensesnitt som benytter seg av høynivå søk i mot 10 digitale bibliotek. Prototypen er utviklet som en applikasjon, i motsetning til i et web-grensesnitt som er det mest vanlige når det kommer denne type systemer. Skjermbildet **[Fig 3-4]** viser grensesnittet til prototypen under et søk.

3.3 University of Michigan Digital Library (UMDL)

The University of Michigan benytter seg av en agentbasert arkitektur **[W_MDL]** i sitt digitale bibliotek, UMDL. Basis for denne arkitekturen ligger blant annet i en artikkel publisert av Birmingham **[BIR95]** som tar for seg et forslag til en agentbasert arkitektur. Det er også publisert andre artikler om arkitekturen, (se Dunfee et. al. **[DUR97]** og Weinstein et. al. **[WEI99]**) og den er tidligere omtalt i hovedfagsavhandlingen til Holmslet **[HOL98]**.

Det vil her redegjøres for arkitekturens hovedprinsipper, i hovedsak basert på Birmingham **[BIR95]** og Weinstein et. al. **[WEI99]**.

3.3.1 Retningslinjer og mål for arkitekturen

Birmingham definerer i sin artikkel **[BIR95]** et sett med retningslinjer for UMDL:

- *For å gjøre UMDL så attraktivt som mulig, både for brukere såvel som innholds- og tjenesteleverandører, vil arkitekturen bygge på et konservativt antall og minst mulig restriktivt sett med standarder.*
- *For å sikre rettigheter og interesse for å lage og utvide samlinger legges det opp til betalingsløsninger som en integrert del av arkitekturen.*[13]
- *Alle tjenester og samlinger i biblioteket skal kunne gjøre sine egne beslutninger basert på sitt eget perspektiv. Med andre ord vil alle tjenestene i systemet være sidestilt og all interaksjon foregå som en dialog mellom tjenestene.*

Målene med arkitekturen er i hovedsak å tilby tjenester som faller under følgende kategorier:

- ***Registrering -*** *en komplett liste over alle agenter (samlinger, brukerinterface og liknende) må vedlikeholdes.*
- ***Megling og agent-team -*** *finne potensielle informasjonskilder og støttetjenester for å tilfredstille brukerens informasjonsbehov.*
- ***Kommersiell støtte -*** *tilby mekanismer som støtter kommersiell utnyttelse av innholdet og samtidig beskytte de intellektuelle rettighetene til eierene av informasjonen og privatlivet til brukerne.*

Utover dette settes det krav om at arkitekturen skal være modulær, skalerbar, utvidbar. Dette for at nye elementer lett skal kunne legges til eller fjernes med enkelhet, uten at det affekterer andre elementer, samt for å understøtte stor vekst i kompleksitet og størrelse.

13. Det pressiseres at dette ikke betyr at tjenesten nødvendigvis skal bli en betal-tjeneste, bare at mulighetene skal være tilstede i arkitekturen.

3.3.2 Agentklasser

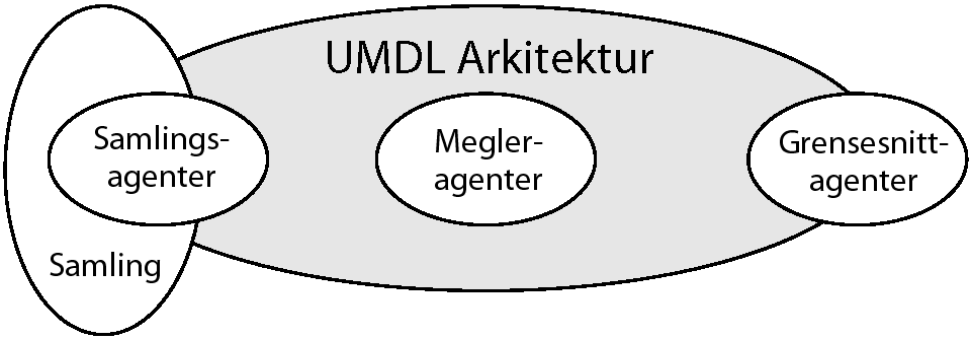

Fig 3-5 UMDL Arkitekturen **[BIR95]** (oversatt fra eng.)

Agentarkitekturen til UMDL består av tre agentklasser **[Fig 3-5]**:

- ***Brukergrensesnitt-agenter***
- ***Megleragenter***
- ***Samlingsagenter***

Brukergrensesnitt-agenter (User Interface Agents)
Disse agentene opptrer som kommunikasjonsvei mellom brukergrensesnitt og resten av systemet. Denne har to formål. Omgjøring av søk fra grensesnitt til UMDL sitt format, og en publisering av brukerprofil til andre egnede agenter. Denne informasjonen vil videre benyttes av megleragenter for videre å understøtte søkeprosessen.

Megleragenter *(Mediators)*
Det finnes et utvalg megleragenter som utfører varierende typer oppgaver. Disse tar blant annet for seg de nødvendige steg for å koble sammen brukergrensesnitt-agenter og samlingsagenter.

Samlingsagenter *(Collection Interface Agents)*
fungerer på sin side litt på samme måte som brukergrensesnitt-agenten, ved at den står for kommunikasjon mellom en samling av informasjon og megleragentene. Agenten publiserer også informasjon om innholdet og mulighetene til samlingen til megleragenter.

Etterhvert som agentarkitekturen blir videre raffinert, vil nye spesialiserte agenter legges til systemet. Eksempelvis vil det kunne lages brukergrensesnitt som er spesialtilpasset en gruppe brukere. I tillegg til dette legges det opp til at agenter skal kunne settes sammen i "team" for lettere å lage nye tjenester, basert på nye agenter. Dette i stor grad fordi Birmingham venter at agent-populasjonen vil forandre seg over tid.

3.3.3 Samarbeid og innkapsling

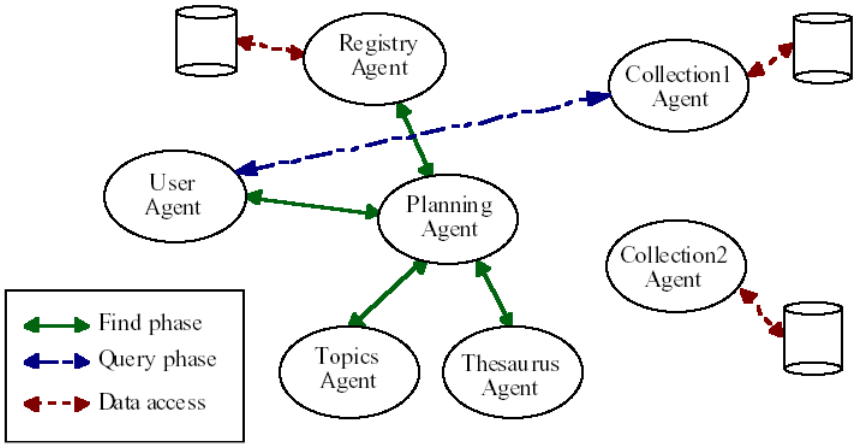

Fig 3-6 UNDL: to-stegs prosessering av søk **[WEI99]**

Figuren **[Fig 3-6]** tar for seg agentenes samarbeid for å tilfredstille et søk i en tidlig versjon av UMDL. **[WEI99]** Brukeren formulerer et søk med et grensesnitt laget av bruker-agenten (user agent). Brukeragenten kontakter deretter planleggings-agenten (planning agent) slik at et team av agenter kan settes sammen og tilfredstille søket.

Planleggingsagenten bruker tre agenter til å hjelpe seg med å fremskaffe denne informasjonen. Emne-agenten (topic agent) inneholder et sett med emner som samlingsagentene (Collection agents) benytter for å beskrive innholdet i samlingene sine. Thesaurus-agenten hjelper til med å binde disse emnene sammen med det aktuelle søket. Beskrivelsene av de enkelte samlingene, og hvilke samlings-agenter som har kontroll over disse samlingene, har registry agenten oversikt over.

Planleggings-agenten greier, ved hjelp av disse tre agentene, å omforme søket til ett sett med grensesnitt-agenter, som kan hjelpe bruker-agenten med å tilfredstille søket.

3.4 MyLibrary@NCState

Her beskrives en arkitektur for digitale bibliotek, MyLibrary@NCState (heretter forkortet til MyLibrary), som er utviklet ved North Carolina State University **[W_NCS]**. Gjennomgangen tar for seg arkitekturen ved å se på modellen som beskriver MyLibrary, og selve implementasjonen av systemet.

3.4.1 Modell

Arbeidet med modellen har foregått ved NCState **[MOD99]** og definerer en tredelt modell for brukersentrerte og dynamiske grensesnitt til bibliotekstjenester **[Fig 3-7]**.

NCState beskriver modellen slik:

> *"[...] an extensible model for implementing a user-centered, customizable interface to a library's collection of information resources. This model, called MyLibrary, integrates principles of librarianship (collection, organziation, dissemination, and evaluation) with globably networked computing resources creating a dynamic, customer-driven front-end to any library's set of materials"*

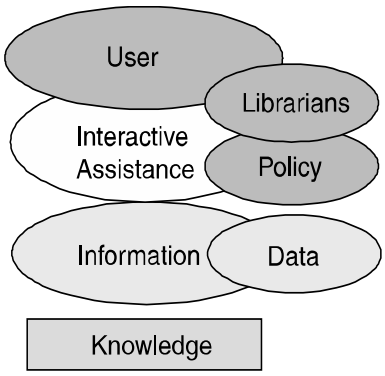

Fig 3-7 "NCSU MyLibrary Model"

Datalager
Datalageret består av både rene dataelementer og informasjon ("kunnskap"). Modellen setter som utgangspunkt at kunnskap eksisterer og er tilgjengelig, og at kunnskap er resultat av prosessering og internalisering (eng: internalization) av data **[MOD99]**. Informasjonen må også være organisert og presentert på en slik måte at den er forståelig for de som skal motta og benytte informasjonen.

Et digitalt bibliotek, i følge denne modellen, er ikke komplett uten at også to andre behov i tillegg til selve datalageret fylles. Bibliotekarer som praktiserer, utformer og implementerer biblioteksprosesser, og interaktiv assistanse. Med disse essensielle komponentene, går modellen over fra kun å være et digitalt lager til å være et "ekte" bibliotek.

Biblioteksprosesser
Man kan si at data er ikke informasjon før de er blitt organisert og gitt en verdi **[MOD99]**. Prosessen med å organisere data med mål om å arkivere kunnskap er bare en av mange prosesser som inngår i ethvert bibliotek. Andre prosesser inkluderer bl.a. det å samle, lagre, evaluere og dele opp data, informasjon og kunnskap.

For effektivt å skape informasjon ut av de dataene man har til rådighet, må man komme frem til endel mål for biblioteket. Først av alt er selve målsettingen for biblioteket viktig å klargjøre, samt målgruppen man ønsker å henvende seg til. Man må også definere omfang og begrensninger for biblioteket. Bibliotekaren har ut i fra dette et bedre utgangspunkt for å velge riktig teknologi til implementasjonen av disse målene.

Interaktiv assistanse
Uansett hvor bra et informasjons system er, vil det fortsatt finnes personer som ikke greier å få tak i informasjonen de søker etter, selv om informasjonen eksisterer i syste-

met. Interaktiv assistanse spesifiserer metoder for interaktiv tolkning og tilrettelegging av innhold. Målet er å redusere vanskeligheter i søkeprosessen, ved å tilby "specialized help for specialized situations" **[MOD99]** og muligheter for å tilrettelegge samlingene til de individuelle brukernes behov.

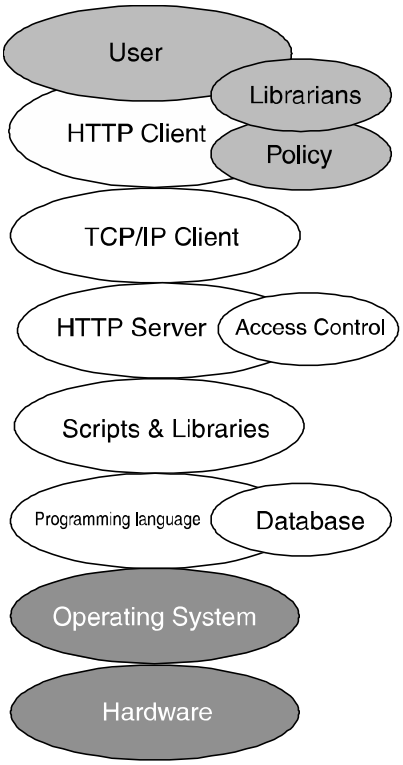

Fig 3-8 "NCSU MyLibrary Technical Infrastucture"

3.4.2 Oversikt over implementasjonen

En implementasjon/prototype av MyLibrary modellen er også utviklet ved NCSU. **[Fig 3-8]** gir en modellert oversikt over den tekniske infrastrukturen i implementasjonen.

Systembeskrivelse
NCSU beskriver implementasjonen slik:

> *"[...] a user-driven, customizable interface to collections of Internet resources -- a portal. Primarily designed for libraries, the system's purpose is to reduce information overload by allowing patrons to select as little or as much information as desired for their personal pages."**[MOD99]***

Systemet baserer seg på brukerprofiler, som gir brukeren mulighet til for eksempel å lagre sine egne lenker til generelle websteder av interesse, til journaler og til abstracts. Systemet setter også brukeren opp imot en eller flere bibliotekarer som kan kontaktes for assistanse.

Når det kommer til selve grensesnittet, kan farger og utseende settes opp slik brukeren ønsker. Brukeren blir også bedt om å oppgi endel personlig informasjon (navn, bosted, e-post adresse og liknende) som kan benyttes av biblioteket.

NCSU benytter seg av agenter i sin "Current Awareness Manager". Her setter man opp visse faglige interesseområder, og blir varslet (via e-post) når mye publikasjoner innenfor disse områdene publiseres.

Funksjonalitet

På implementasjonens webside **[W_MYL]** fremheves det endel viktige punkter som fremhever MyLibrary fremfor andre systemer:

- ***Rik på innhold***
  Databasen som ligger i bunn av MyLibrary inneholder lenker til informasjonsrik informasjon. Den representerer innhold som er nødvendig for research i et akademisk miljø. Typer av informasjon inkluderer full-tekst databaser, datasett, bibliografiske databaser, elektronisk tekst av alle slag, referansemateriale og det de ser på som viktigst - direkte tilgang for bibliotekarer.
- ***Portabel***
  Siden du lager deg et brukernavn og passord på en åpen webside, kan du benytte MyLibrary stort sett hvor som helst.
- ***Tilpasset***
  Du kan selv velge hva du vil skal vises, og fremfor alt hva som ikke skal vises.
- ***Dynamisk***
  MyLibrary skal være lett å administrere og vedlikeholde, og holdes dermed lett kontinuerlig oppdatert.
- ***Proaktiv***
  Ved å iverksette funksjoner i systemet, vil du kunne motta informasjon fra bibliotekarer (f.eks. om nye bøker) basert på dine interesseområder.
- ***Fokusert***
  I hovedsak lister MyLibrary opp bare den informasjonen du vil at skal vises. Som en følge av dette vil du slippe unna endel "støy". Dette setter derimot ingen begrensninger på hvor mye informasjon du ønsker at skal vises.
- ***Plattform-uavhengig***
  For å bruke MyLibrary kreves kun en web-browser med støtte for cookies, samt SSL for sikker kommunikasjon.
- ***Privat***
  All konfigurering innenfor systemet holdes konfidensielt.

Grensesnittet

[Fig 3-9] viser hovedsiden til MyLibrary. Ut i fra denne fremkommer 8 forskjellige emner som grensesnittet er delt inn i.

1. ***My Librarian(s)***
   - her fremgår en liste over bibliotekarer som kan hjelpe deg.
2. ***Library Links***
   - lenker til generell og spesifikk biblioteksinformasjon. Eksempler på slike lenker er åpningstider for bibliotekene, din lånehistorikk og tilbakemeldings-skjemaer (manglende bøker, kommentarer etc.)

Fig 3-9 MyLibrary@NCState: Grensesnitt

3. ***University Links***
   - både på generelt grunnlag, og spesifikt for dine interesseområder foreslås her lenker til universitetsressurser som kan være av interesse.
4. ***New Titles***
   - her fremgår resultatet av bibliotekets "Current Awareness Manager". Profilen til den enkelte bruker settes opp med en eller flere emner (basert på emnekatalogen til Library Of Congress ***[W_LOC]****).*
5. ***Quick Searches***
   - direkte søk i de mest brukte samlingene i biblioteket.
6. ***Reference Shelf***
   - her kan brukeren velge å legge til eller fjerne interessante ressurser/referanser.
7. ***Indexes and Abstracts***
   - herifra kan søk i bibliografiske databaser (f.eks. ACM, Census) settes opp og benyttes direkte.
8. ***Electronic Journals***
   - NCSU har lisensiert tilgang til over 1000 full-tekst journaler. De som er av interesse kan man sette opp her.

Hver av disse informasjons-blokkene kan konfigureres individuelt så lenge man er logget inn på biblioteket. Brukere autentiseres ved hjelp av et felles brukernavn og passord som benyttes til resten av datasystemet til NCSU.

Punktene NCSU setter for systemet er realistiske og ser ut til å bli fulgt i implementasjonen.[14]

3.5 Sammendrag

Dette kapittelet har tatt for seg fire ulike arkitekturer for digitale bibliotek. Vi tar her for oss en kort oppsummering av arkitekturene:

SmartPush
Arkitekturen består av tre agenter, en som tar for seg profilen til brukeren (profile agent), en som kobler dokument metadata og brukerprofiler sammen, og en som leverer innhold til endebrukere.

SmartPush skiller seg ut i sammenhengen ved at den ikke direkte omhandler seg selv som noen en arkitektur for digitalt bibliotek. Den er likevel med her fordi den går under avhandlingens definisjon av et digitalt bibliotek, og samtidig illustrerer områder som ikke i så stor grad omtales i de tre andre arkitekturene. Systemet skiller seg også ut vet at det er det mest "kommersielle" og pragmatiske systemet av de fire, ved at det løser et veldig spesifikt problem for blant annet utgivere av aviser.

Daffodil
Hovedfunksjonaliteten til Daffodil går på integrering av digitale bibliotek ved hjelp av høynivå-søk. Arkitekturen benytter seg av "wrappers" som oversetter de enkelte digitale bibliotekene til et felles system og i et felles grensesnitt.

UMDL
Agentarkitekturen til University of Michigan beskriver i stor grad hvordan agenter kan jobbe sammen for å løse problemer innenfor digitale bibliotek. Arkitekturen definerer også "wrappers" både for grensesnitt og samlinger.

MyLibrary
En komplett løsning på et digitalt bibliotek utviklet ved North Carolina State University. Det legges stor vekt på utforming og personifisering av grensesnitt i digitale bibliotek.

14. Mulighetene til virkelig å få testet systemet inngående er begrenset til brukere innenfor universitetet, så det er vanskelig å si eksakt hvor vellykket implementeringen er.

4 Forslag til arkitektur for digitale bibliotek

Dette kapittelet tar for seg et forslag til arkitektur for digitale bibliotek. Forslaget er utarbeidet av forfatteren, og bygger i stor grad på forskning og utvikling forfatteren har gjort innenfor dynamiske grensesnitt og samlinger.

Forslaget bygger også i stor grad på terminologi og teori innenfor området digitale bibliotek (blant annet **[FUHR00]**, **[KW95]** og **[MYL99]**). Terminologien som er benyttet, og da spesielt teorier rundt agenter, er både inspirert av, og baserer seg på, flere av de arkitekturene som er gjennomgått i *kapittel 3*.

I tråd med de målene som skisseres i innledningen (ref. *kapittel 1.1.4*) vil arkitekturen som fremlegges være underlagt noen primære mål. For å ta fatt på disse målene, er det nødvendig å først definere basiskomponentene som arkitekturen vil bestå av. En mulig oppdeling er å se på arkitekturen i form av tre logiske deler:

- ***Grensesnitt***
  - interaksjon med brukeren
- ***Datalager***
  - inneholder samlinger og andre informasjonsobjekter (for eksempel brukerprofiler)
- ***Struktur***
  - er det som måtte finnes mellom grensesnitt og datalager

Strukturen i arkitekturen vil beskrives ved hjelp av agenter. Kapittel 3 bleskriver flere arkitekturer som baserer seg på en eller annen form for agentbasert arkitektur (se blant annet **[FUHR00]** og**[BIR95]**).

Arkitekturen tar i stor grad for seg målene ut ifra et teknisk perspektiv, og det vil legges vekt på problemer og løsninger som er av teknisk art. Det legges dermed vekt på hvordan man skal bygge opp et digitalt bibliotek rent teknisk og infrastruktur-messig fremfor eksempelvis samfunnsmessige eller psykologiske innfallsvinkler. Det refereres videre til Fuhr et. al. **[GCES01]** og kapittel 1.1.6, *"Tilnærming"* for en utdypet beskrivelse av avhandlingens perspektiv.

Kapittelet starter med en gjennomgang av hovedkomponentene i arkitekturen. En modell av arkitekturen skisseres, som danner grunnlaget for en mer gjennomgående beskrivelse av arkitekturens enkelte komponenter og hvordan disse arbeider i forhold til hverandre.

En gjennomgang av informasjonsflyten i modellen avslutter kapittelet. Det vil i denne gjennomgangen sett på et tenkt case og hvordan informasjonen beveger seg mellom de forskjellige agentene i modellen.

4.1 Arkitekturens hovedkomponenter

Arkitekturen består av tre hovedkomponenter.

- ***Grensesnitt***
  Her inngår alle typer grensesnitt som går under definisjonen som er skissert tidligere (ref. kapittel 2.2). Eksempler kan være alt fra WEB-klienter (browsere), mobiltelefoner, PDA-enheter til TV og elektroniske oppslagstavler.
- ***Datalager***
  Her lagres all data og metadata innenfor det aktuelle DL. Datalageret inneholder tre kategorier data: metadata om samlinger, brukerinformasjon og de reelle samlingene.
- ***Agenter***
  Her inngår de funksjoner som sammen med datalager og grensesnitt danner et komplett digitalt bibliotek. Disse prosessene illustreres som agenter, hver med sine spesifikke oppgaver - som tilsammen dekker det samlede behovet til det aktuelle DL.

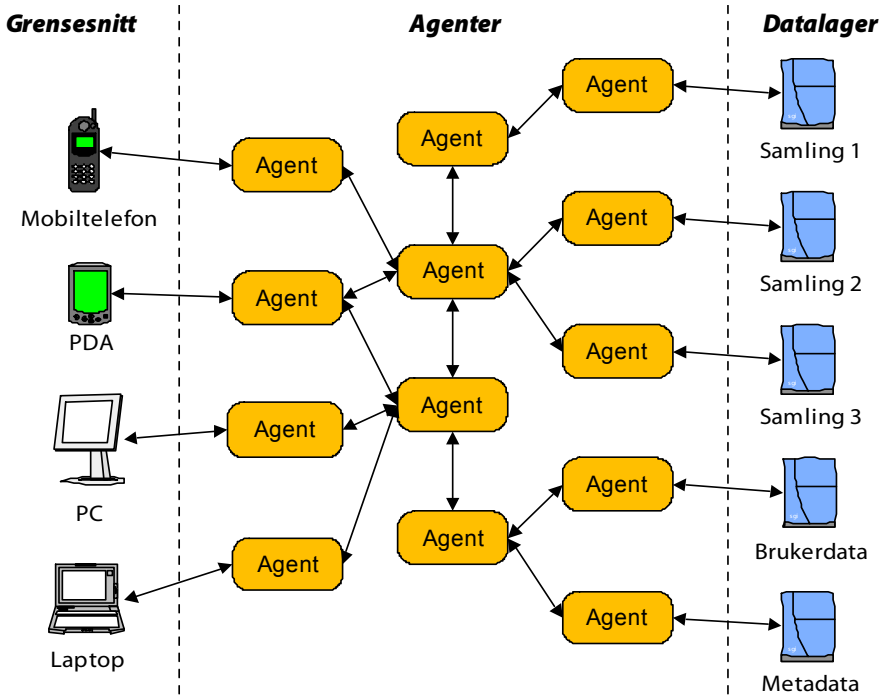

Fig 4-1 Arkitektur, Basiskomponenter

Figuren **[Fig 4-1]** tar for seg disse tre komponentene, og illustrerer løst forholdet komponentene har til hverandre. Figuren vil senere bli utdypet i en modell basert på arkitekturen (ref. side 43).

4.1.1 Datalager

Datalagerets hovedoppgave er å holde på informasjon (data) og kunnskap. Arkitekturen setter krav om at minst ett datalager er tilgjengelig for lagring av informasjon.

Det settes derimot ingen direkte krav til hvordan dataene lagres eller hvor de ligger lagret. Det er derfor ikke være noe problem i en implementasjon av arkitekturen å ha ett eller flere datalager, eller samlinger som ikke er i direkte kontakt med resten av systemet.

Et typisk datalager eller sett med datalager i arkitekturen inneholder tilsammen; **informasjon om brukerne** (personlige profiler), **informasjon om samlingene** (samlingsmetadata), og **selve samlingene** (informasjon og kunnskap).

4.1.2 Agenter

Arkitekturen består også av et sett med agenter som tar for seg all innhenting, evaluering og bearbeiding av informasjon. Den overordnede oppgaven til agenter i denne arkitekturen, er sammen å utføre alle de oppgaver som er nødvendig for at informasjon flyttes mellom datalager og grensesnitt.

Arkitekturen setter krav om at agentene sammen må kunne understøtte følgende oppgaver i et digitalt bibliotek:

- *Tilgang og administrasjon av samlinger.*
- *Tilgang og administrasjon av metadata om samlinger.*
- *Bruk, opprettelse og oppdatering av brukerprofiler.*
- *Aksessering av informasjon på grunnlag av brukerprofil og metadata om samlinger.*

Det er flere grunner til at agentbegrepet er i bruk i arkitekturen. En årsak er at det forhåpentligvis kan identifisere og beskrive de funksjonelle delene av et gitt digitalt bibliotek. Disse er i hovedsak ment å kunne overføres direkte til prosedyrer og objekter i en implementasjon av arkitekturen. I denne arkitekturen fungerer de også som en måte å dra inn og promotere en objektorientert tankegang.

Grensesnittagenter

Grensesnitt og datalager kan, i motsetning til agenter, regnes som "passive" aktører i et digitalt bibliotek. Agenter må utføre de oppgaver som ikke ligger i grensesnittets eller datalagerets natur[15].

Personlige profiler

En viktig mulighet som modellen ønsker å tilrettelegge for, er at brukere kan sette opp personlige profiler og spesifisere rutiner de ønsker utført. Denne informasjonen benyttes, i sin tur, av en eller flere agenter slik at brukeren for utført sine ønsker.

Asynkrone agenter

Arkitekturen i seg selv setter ingen begrensninger til når eller hvordan agentene skal nå sitt mål. Agenter kan dermed teoretisk også involvere ikke-maskinelle metoder, og på den måten fungere asynkront. Et eksempel på dette er at en person som er involvert i det aktuelle digitale biblioteket får tilsendt en e-post, der han blir bedt om å utføre en oppgave. Mange ganger er det behov for slike rutiner, f.eks. om det er nødvendig å godkjenne en bruker som prøver å registrere seg i DL manuelt.

15. Med et grensesnitt eller datalagers "natur" her menes for eksempel et SQL-interface til en database, eller den underliggende HTML dekodingen som foregår i et web-grensesnitt for å få frem det endelige resultatet.

Automatiske rutiner
Detfinnes også agenter som utfører automatiske rutiner som foregår til gitte tider på døgnet (f.eks. nedlasting av data hver natt fra en sentral database). Dette kan være nødvendig i de tilfeller det benyttes en lokal kopi av en database, eller et subset av en større database. Automatiske rutiner vil også være en løsning for backup-bruk og kvalitetssikring (for eksempel slette utdaterte data).

Events
En annen mulig gruppe av agenter, kan reagere på, og sette i gang prosesser, når en gitt situasjon er oppfylt. Et mulig eksempel på dette er å gi beskjed når en ny publikasjon innenfor interesseområdet til en spesifikk bruker er tilgjengelig.

En utdypet og mer spesifikk diskusjon om agenter omtales i detalj i forbindelse med gjennomgangen av arkitektur-modellen (ref. **[Fig 4-2]** *i kapittel 4.2*).

4.1.3 Grensesnitt

Det må finnes minst ett, og gjerne flere grensesnitt mot brukeren med tilhørende grensesnitt-agent til hvert enkelt.

Grensesnitt	Fordeler	Ulemper
Web grensesnitt på personlig datamaskin	Høy grad av lesbarhet, store mengder data kan gjøres tilgjengelig, stor grad av fleksibilitet, få tilleggskrav til informasjonsleverandør utenom en webserver på nett	Krever tilgang til internett på datamaskinen, ikke tilgjengelig overalt
Telefon (tale, DTMF)	Stor grad av tilgjengelighet, de fleste hjem har en telefon.	Fordi dette stort sett begrenser seg til tall mellom 0-9 og tale er det begrenset datamengde som kan gå mellom bruker og system. For informasjonsleverandør kreves spesialsystemer
Mobiltelefon (SMS, tale)	Mobilitet. Tilgjengelighet	Relativt små datamengder kan utveksles. For informasjonsleverandør kreves spesialsystemer
WebTV (toveis, via kabelselskap)	Relativt store datamengder kan overføres	Vanskelig å lese tekst
Portable løsninger, eksempelvis avanserte mobiltelefoner (WAP/MMS/DoCoMo) og PDA	Mobilitet, tilgjengelighet og benytter i stor grad internett. Dermed mindre krav til spesialsystemer for informasjonsleverandør (teleselskaper og liknende)	Bare moderate datamengder kan fremvises og overføres innenfor rimelig tid.

Tabell 4-1 Oversikt over noen mulige interaktive grensesnitt

Arkitekturen setter få begrensninger på hva slags grensesnitt som kan benyttes, men setter krav om en viss grad av interaktivitet mellom bruker og grensesnitt (toveis kommunikasjon, eller andre metoder for å gjøre brukeren unik identifiserbar) enten direkte (f.eks. web) eller indirekte ved en kombinasjon av kommunikasjonsmetoder (f.eks. SMS og direktesendt TV).

[Tabell 4-1] gir en oversikt over forslag til mulige grensesnitt som tilfredstiller kravet om interaktivitet, og belyser noen av de fordeler og ulemper som løsningen innebærer. Tabellen er på ingen måte en komplett oversikt, men gir en pekepinn på state-of-the-art innenfor mulige brukergrensesnitt pr. juni 2002.

4.1.4 Dynamiske grensesnitt og samlinger

Arkitekturen baseres på et grunnprinsipp om at data er dynamisk av natur **[BUL98]** og at få eller ingen informasjonsobjekter i et digitalt bibliotek kan regnes å være statiske. Dette omfatter alt fra selve innholdet i samlingene, til brukegrensesnittene, all brukerinformasjon, mediene informasjonen lagres på, samt de distribusjonsmetoder som benyttes.

Som et ledd i denne dynamiske tankegangen foreslås det for eksempel å benytte agenter som holder informasjonen i samlinger oppdatert og korrekt i den grad det er mulig. Dette innebærer eksempelvis alltid hente ned siste utgave av en artikkel, siste revisjon av en bok eller oppdaterte lottoresultater. Avhengig av mål og bruksområde for systemet, er det også naturlig å ta stilling til om historisk informasjon (revisjonshistorie) for de enkelte informasjonsobjektene skal lagres[16].

En viktig del av arkitekturen er videreføring av denne dynamiske tankegangen til å inkludere selve samlingene. Tanken bak dette er at også hvilke samlinger som finnes i det digitale biblioteket kan variere, og at det ikke legges noen barrierer for hverken at eksisterende samlinger fjernes eller nye legges til. Ideelt fanger et system basert på arkitekturen, så langt som praktisk mulig, selv opp hvilke samlinger som til en hver tid er tilgjengelige.

Grensesnittene på sin side drar også stor fordel av å oppfattes som dynamiske. Dette går både på hva som presenteres i grensesnittet, hvordan det vises og hvilket grensesnitt som benyttes.

Den største fordelen med denne dynamiske tankegangen er, fremfor alt, at man ideelt sett oppnår en mer stabil kodebase på et implementert system. Dette oppnås ved at man legger opp til at minimalt må endres i et ferdigimplementert DL for å legge til og fjerne både samlinger og grensesnitt. Det er også sansynligvis lettere å delegere ansvarsforhold mellom de enkelte delene i et DL basert på denne arkitekturen, da de enkelte delene skal være så selvstendige som mulig.

16. Arkitekturen vil ikke ta spesielle hensyn til akkurat disse aspektene, men det er ikke noe som hindrer å legge til denne type funksjonalitet.

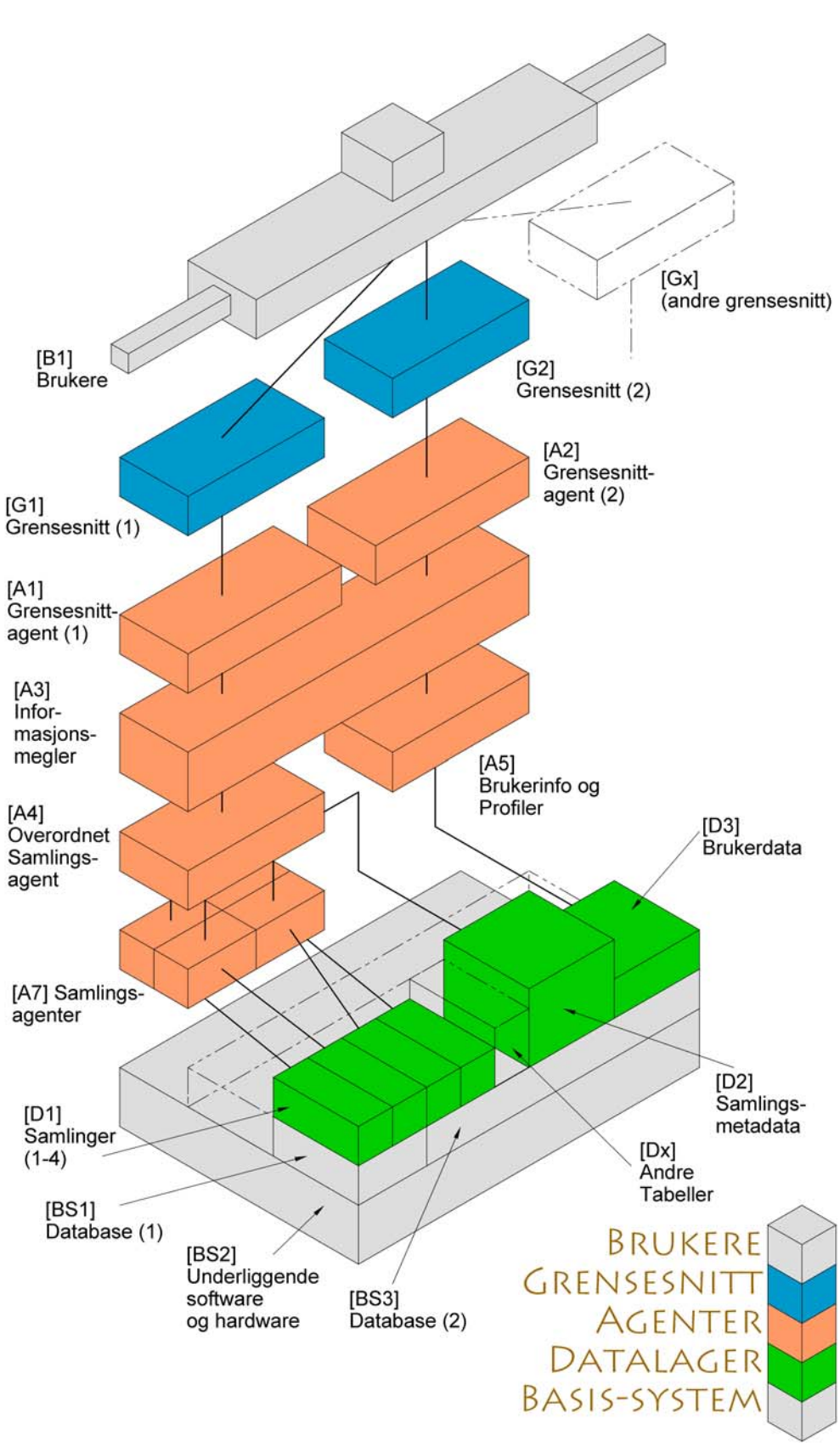

Fig 4-2 Modell av arkitekturen

4.2 Inngående beskrivelse av komponentene i arkitekturen

Modellen i **[Fig 4-2]** fremstiller en mulig tolkning av arkitekturen skissert tidligere i dette kapittelet. Målet med modellen er å få frem et mer detaljert og konkretisert bilde av prinsippene i arkitekturen, for igjen å kunne peke på enkelte viktige detaljer. Modellen blir her gjennomgått, og hver enkelt element beskrives i detalj.

Modellen **[Fig 4-2]** over arkitekturen kan deles inn i fem kategorier:

- ***Brukere [Bx]***
  Brukere eller grupper av brukere som er i kontakt med systemet.
- ***Grensesnitt [Gx]***
  Ett eller flere grensesnitt som blir brukernes mulige inngangsportaler til systemet.
- ***Agenter [Ax]***
  Et sett med agenter som til sammen utfører alle nødvendige oppgaver innenfor DL.
- ***Datalager [Dx]***
  Alle samlingene som inngår i det aktuelle DL.
- ***Basis system [BSx]***
  Teknologiplattform (hardware), operativsystem, database-servere, basis-verktøy og andre støttesystemer.

Gjennomgangen av modellen foregår nedenfra og oppover i modellen, og dermed beskrive hva som skjer med dataene underveis fra datalageret i bunn til brukeren på toppen. Referanser i formen **[Bx]**, **[Gx]**, **[Ax]**, **[Dx]** og **[BSx]** vil heretter bli benyttet for å referere direkte til de enkelte delene av **[Fig 4-2]**.

4.2.1 Basis-system

Starter vi fra bunnen av modellen **[Fig 4-2]** finner vi underliggende hardware og software (farget grått). Innholdet i disse regnes utenfor rekkevidden til modellen, men representeres her for å sette den ellers "logiske" modellen inn i en større sammenheng.

Arkitekturen ønsker ikke å sette noen spesifikke krav til hva slags teknologi (hardware og software) som benyttes for implementasjon. Det henvises til kapittel 4.4, "Teknologi" for en mer generell diskusjon om tekniske krav og muligheter innenfor digitale bibliotek.

4.2.2 Datalager

Beveger vi oss oppover i **[Fig 4-2]** finner vi modellens datalager **[Dx]**. Datalageret tar for seg lagringen av dataelementene som benyttes i modellen.

Datalageret kan inndeles i to hovedkategorier:

- ***System-støttende data***
  Består av samlingsmetadata og brukerdata. Disse har til felles at de understøtte behovene en implementasjon har for løpende lagring av informasjon om brukere og samlinger.
- ***Samlinger***
  En fellesnevner for alle kolleksjoner av data og informasjon som representeres i biblioteket.

Samlingsmetadata
Samlingsmetadata er en viktig del av arkitekturen. Her er hver av samlingene tilknyttet systemet, beskrevet i detalj. Her beskrives informasjon skal så lang det er mulig som hvordan dataene kan aksesseres, hva samlingen inneholder, hvor den befinner seg, hvilke typer informasjon som finnes og liknende, etter en felles mal.

Brukerdata
Her beskrives brukerne av systemet. Dette avgrenses ikke nødvendigvis bare til endebrukere. De som vedlikeholder og utvikler systemet vil også kunne representeres her. Hver bruker vil også kunne være tilknyttet sin egen profil som inneholder brukerens egne instillinger i forhold til systemet og eventuell informasjon systemet har bygget opp om personen.

Samlinger
Samlingene i systemet[17] danner selve informasjonssenteret i arkitekturen. Disse kan bestå av alt fra metadata, tekstelementer og bilder til lydfiler og videoelementer. Arkitekturen setter ingen kav til at samlingene befinner seg innenfor systemet eller at systemet har full kontroll over samlingen, selv om modellen tar utgangspunkt i at dette er tilfelle. Det er derimot et krav at både interne og eksterne samlinger beskrives i datalageret "samlingsmetadata".

4.2.3 Spesifikke samlingsagenter

Modellen definerer videre samlingsagenter **[A7]** for hver av samlingene **[D1]**. Kommunikasjon mellom datalagerets forskjellige samlinger går via disse agentene. Et av formålene med disse agentene er å skape en uavhengighet mellom samlingens implementasjon og resten av systemet. Dette betyr at om en samling er representert i en relasjonsdatabase, og en annen kun består av en tekstfil, vil hver enkelt samlingsagent ta hånd om dette, og skape en enhetlig kommunikasjon videre. En samling vil dermed også kunne bytte lagringsmåte eller plattform uten å skape ringvirkninger for resten av systemet.

4.2.4 Overordnet samlingsagent

Denne agenten tar hånd om kommunikasjonen med de spesifikke samlingsagentene **[A7].** Når denne agenten får en forespørsel fra informasjonsmegleren **[A3]** benytte den informasjon om de enkelte samlingene, beskrevet i datalageret samlingsmetadata **[D2]** for både å finne ut hva slags samlinger som er tilgjengelige, hvordan disse skal aksesseres, og deretter sende en forespørsel til en eller flere spesifikke samlingsagenter **[A7]**.

4.2.5 Brukerinfo og profiler

Denne agentens hovedoppgave er å ta hånd om all informasjonsutveksling som omhandler brukere og brukerprofiler, og benytter datalageret brukerinfo **[D3]** til lagring. Agenten fungerer ellers på samme måte som de spesifikke samlingsagentene.

17. Modellen viser tre enkeltstående samlinger.

4.2.6 Informasjonsmegler

Hovedoppgaven til informasjonsmegleren er å stå for interaksjonen mellom grensesnittagentene **[A1,A2]**, overordnet samlings-agent **[A4]** og brukerdata-agent **[A5].**

Eksempler på data denne agenten vil ha tilgang til og prosesserer er **oppgitt informasjon** (f.eks. et søkeord motatt fra et grensesnitt), **tilgjengelige samlinger**, brukerens **tilgangsrettigheter** og **personlige profil**. Agenten benytter data som finnes om brukeren (eksempelvis tilgangsrettigheter til data og interesse profil) og kombinerer det med den informasjonen som er tilgjengelig.

Eks. 4-1 ***Eksempel, B2C/B2B virksomhet***

For å forklare noen av de ansvarsområdene informasjonsmegleren har, kan man eksempelvis ta for seg en bedrift som både driver med B2B (business to business) og B2C (business to customer) virksomhet. Eksempelet tar utgangspunkt i et firma som både er importør og driver detaljsalg til endebrukere.

Til B2C kunder ønsker man å gi endebruker-priser, og et gitt sortement. Til B2B kunder ønsker man derimot å formidle forhandlerpriser til, og i visse tilfeller også et annet sortement av varer. Informasjonsbehovet på de enkelte varer er også forskjellig for endebrukere og bedriftskunder. Mens forbrukerne er veldig interessert i egenskaper ved et produkt, vil leveringstid og kvantiumsrabatter mer interessant for bedriftskunden.

*En mulighet er å definere to samlinger - en som inneholder alle endebruker-produkter **[D1.1]** og en som inneholder alle produkter myntet på bedriftskunder **[D1.2]**. I tillegg må man definere brukerdata **[D3]** for å identifisere brukere og bedrifter og spesielle preferanser de enkelte av disse har. Samlingsmetadata **[D2]** behøver man også for å forklare detaljene i **[D1.1]** og **[D1.2]**.*

*Gitt et ønske om å vise varesortimentet til en definert bruker, tar informasjonsmegleren først informasjon om brukeren fra Brukerinfo/Profil-agenten **[A5]** og kombinere denne informasjonen med informasjon som er forhandlet frem med den overordnede samlingsagenten **[A4]**.*

*Om **[A5]** definerer en bruker som "endebruker" og brukeren har spesialparameteren "uten mva", hentes sortementet for endebrukere hentes frem, pris regnes ut fra at brukeren ikke ønsker mva kalkulert inn, og resultatet sendes til ønsket grensesnitt-agent.*

Agenten blir også gjennomgått i detalj i forbindelse med prototypen (kapittel 5).

4.2.7 Agenter med domenekunskap om grensesnitt

Agentene i arkitekturen som tar for seg kommunikasjonen med grensesnittene (**[A1]** og **[A2]**), lages spesifikt for hvert enkelt grensesnitt. Dette har to viktige fordeler. Først og fremst bidrar dette til at grensesnittet er så optimalt som mulig både når det gjelder funksjonalitet og utseende. Den andre fordelen er at dette også gjør grensesnittene uavhengige av hverandre slik at nye grensesnitt kan legges til eller erstatte gamle uten at det har noen innvirkning på resten av systemet.

Det er viktig at man er klar over at selv om disse agentene skal ha en god del domenekunnskap om grensesnittene, benytter de seg i størst mulig grad av informasjon fra andre agenter når det kommer til **hva** som skal vises og **hvordan**. Eksempelvis vet agenten ideelt sett hvordan man på best måte viser et bilde, en tabell, eller stiller et spørsmål til brukeren, uten at agenten sitter på selve spørsmålene eller selve bildet.

Eks. 4-2 ***Eksempel, To grensesnitt mot samme kilde***

Eksempelvis kan man her ta for seg et grensesnitt mot en mobiltelefon og et grensesnitt mot en PDA. Mobiltelefoner i markedet pr. juni 2002 har et relativt begrenset grensesnitt mot brukeren. Små display og numeriske tastatur, i tillegg til lave oppkoblingshastigheter karakteriserer også mediet. Tar man WAP som et eksempel begrenses man i hovedsak til mellom 60 og 80 tegn samtidig på skjermen. En PDA derimot kan ha alfanumerisk tastatur og relativt stor skjerm, gjerne med farger.

Vi definerer to funksjoner som må være tilstede i våre grensesnitt-agenter:

- visBilde("bilde");
- skrivArtikkel("overskrift", "ingress", "brødtekst");

For mobiltelefon-agenten defineres disse to funksjonene som at det er en WAP side som sendes til grensesnittet (mobiltelefonen). ***visBilde()*** *funksjonen skalerer ned bildet til en størrelse som kan vises på den aktuelle mobiltelefonen og legger det aktuelle bildet til wap siden.* ***skrivArtikkel()*** *lager en tredelt tekstblokk med bold tekst på overskrift, ingress med kursiv og brødtekst i normal skrifttype.*

PDA grensesnittet har litt flere muligheter, men viser fortsatt ikke all type grafikk. Så også her er det sannsynligvis smart å legge inn en rutine for å sjekke dimensjonene til bildet og eventuelt skalere det etter behov. Ellers kan de fleste PDAer lese av HTML sider, så ut-dataene fra agenten til grensesnittet er her formatert som enkel HTML. skrivArtikkel() funksjonen lager her en tabell i HTML med bruk av fonter og skriftstørrelser, i tillegg til attributter (bold, italic etc.) på teksten.

Eksempelet over illustrerer i enkle trekk hvordan man effektivt kan bygge opp et system basert på disse prinsippene som setter fokus mer på *hva* man ønsker å vise enn hvordan.

4.2.8 Brukeren og grensesnittene

På toppen av modellen finner vi brukeren **[B1]**. Kravene som stilles til denne er at den kan aksisere en av de tilgjengelige grensesnittene **[Gx]** i modellen. Modellen velger å konsentrere seg rundt to grensesnitt. Disse er ikke navngitt spesifikt i håp om at det gjør at modellen lett kan settes inn i forskjellige sammenhenger og dekke de behov som det aktuelle digitale biblioteket ønsker å tilfredstille.

Felles for grensesnittene i vår modell er at de er den fysiske manifestasjonen av grensesnittet for brukeren, og er i vår modell "dumme" og uten egen funksjonalitet, men i steden avhengig av underliggende funksjonalitet.

Refererer igjen til **[Tabell 4-1]** som tar for seg noen grensesnitt som kan være aktuelle. En tidlig versjon av modellen tok for seg grensesnittene WEB og PDA som to mulige grensesnitt (**[G1]** og **[G2]**). Andre muligheter som kan nevnes og som gjennomgås i tabellen, er TV (WebTV) og telefon (talesentraler, SMS, WAP etc.). Grensesnitt-agentene **[Ax]** spesialtilpasses nødvendigvis hver av disse grensesnittene.

4.3 Informasjonsflyt

Arkitekturens vekt på selvstendige agenter setter store krav til informasjonsflyt gjennom hele modellen. Informasjonsutvekslingen binder modellen sammen, og er derfor et område som man må legge spesielt stor vekt på ved utvikling av systemer basert på arkitekturen.

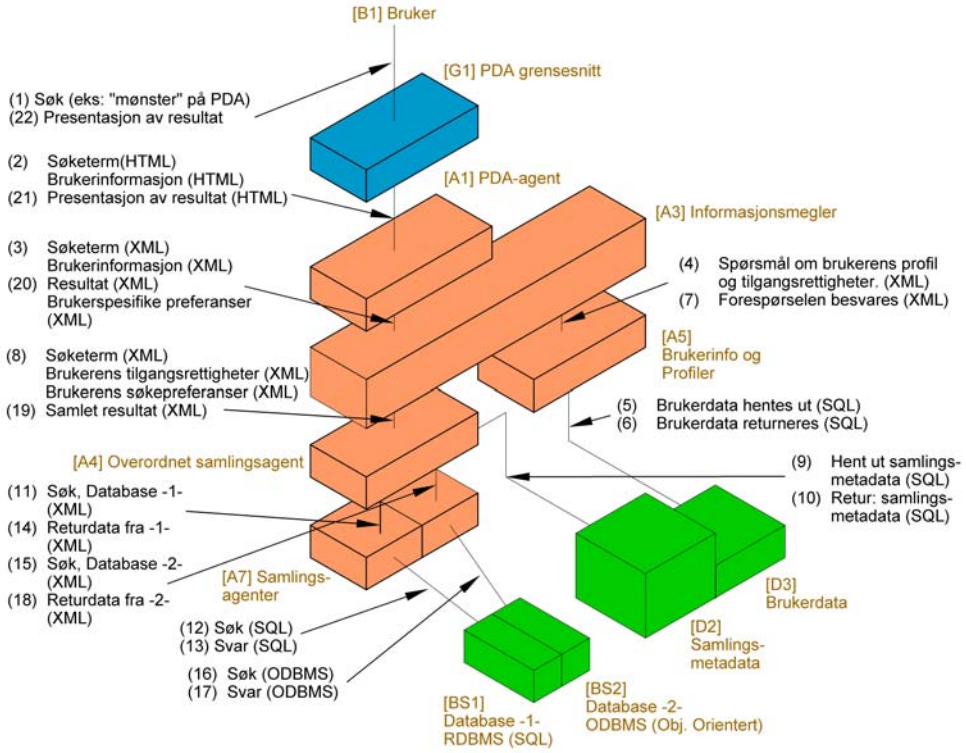

Fig 4-3 Eksempel på informasjonsflyt mellom agentene

Et enkelt scenario/eksempel blir her gjennomgått for å illustrere noen mulige relasjoner mellom de enkelte delene i **[Fig 4-2]**. Informasjonsflyten gjennom modellen blir først beskrevet med en rask gjennomgang av de enkelte stegene i figuren.

Eksempelet blir deretter videre konkretisert og beskrevet steg for steg, og i stor grad eksemplifisert ved bruk av kodefragmenter. Formålet med eksempelet er at det beskriver hvordan de enkelte delene av modellen kan tenkes å kommunisere og fungere sammen.

Det er verdt å merke seg at fragmentene med kode og data som fremgår i scenariet (bl.a. Web-interface, HTML, XML og SQL) velges ut for å illustrere dataflyten på en gjenkjennelig form. Det setter på ingen måte noen begrensninger eller krav til reelle metoder for datautveksling i en implementasjon av arkitekturen.

Strukturert gjennomgang av informasjonsflyten i [Eks. 4-3]

1. *Data (søketerm) fra bruker mottas av det "fysiske" grensesnittet til brukeren* ***[G1].***
2. *Søketermen overføres til grensesnittagenten* ***[A1].***
3. *Termen sendes til informasjonsmegler* ***[A3].***
4. *Informasjonsmegler ber om informasjon om brukeren fra agenten "brukerinfo og profiler"* ***[A5].***
5. *Agenten ber om informasjon i datalageret* ***[D3].***
6. *"Brukerinfo/profiler" får tilbake alle data vedrørende aktuell bruker (tilgangsrettigheter og profiler som er registrert) fra datalageret.*
7. *Relevant informasjon fra datamengden i (6) sendes videre til informasjonsmegler* ***[A3].***
8. *Søketermen, tilgangsrettighetene og søkepreferansene til brukeren sendes til overordnet samlingsagent* ***[A4].***
9. *Informasjon om samlingsmetadata etterspørres fra datalageret* ***[D2].***
10. *Oversikt over alle tilgjengelige databaser returneres til overordnet samlingsagent* ***[A4].***
11. *Søkeordet og hvilke felter i databasen som det søkes i, sendes til samlingsagent* ***[A7.1].***
12. *Agenten søker i databasen* ***[BS1]*** *ved hjelp av domenekunnskapen agenten har om databasen.*
13. *Resultatet mottas av samlingsagenten* ***[A7.1].***
14. *Samlingsagenten sender resultatet til overordnet samlingsagent* ***[A4]****.*
15. *Søkeordet og hvilke felter i databasen som det søkes i, sendes til samlingsagent* ***[A7.2].***
16. *Agenten søker i databasen* ***[BS2]*** *ved hjelp av domenekunnskapen agenten har om databasen.*
17. *Resultatet mottas av samlingsagenten* ***[A7.1].***
18. *Søkeordet og hvilke felter i databasen som skal søkes i sendes til samlingsagent* ***[A7.2].***
19. *Resultatet fra de to søkene integreres og sendes til informasjonsmegler* ***[A3].***
20. *Informasjonen vurderes videre ut ifra brukerens profil og hvorvidt det er i samsvar med hva brukeren ønsker - og videresendes grensesnitt-agenten* ***[A1].***
21. *Grensesnitt-agenten bruker domenekunnskapen den har om grensesnittet til å lage en egnet fremstilling (f.eks. i HTML). Denne sendes grensesnittet* ***[G1]*** *som presenterer dataene.*
22. *Brukeren* ***[B1]*** *kan lese av resultatet.*

Eks. 4-3 ***Eksempel, "Søk: Mønster"***

Det tas utgangspunkt i en bruker som foretar et enkelt søk på en PDA-enhet tilknyttet et digitalt bibliotek bygget på modellen i ***[Fig 4-2]****. Dataflyten er lagt til denne modellen og gitt nummer fra 1 til 22 i figuren* ***[Fig 4-3]*** *over. Brukeren er innlogget i systemet, og har en personlig profil tilhørende sitt brukernavn som allerede ligger lagret i systemet.*

Brukeren ***[B1]*** *velger å skrive "Mønster" i søkeboksen* ***[Eks. 4-4]*** *på PDA-grensesnittet* ***[G1]****. Det antas at PDAen er tilkoblet direkte til tjenesteleverandør, slik at søket prosesseres umiddelbart etter at "Utfør søk" knappen er blitt trykket.*

Eksempel 4-4 Søkedialog i et PDA grensesnitt

```
POST /cgi-bin/post-query HTTP/1.0
Accept: www/source
Accept: text/html
Accept: image/jpeg
Accept: image/gif
User-Agent:  Lynx/2.2  libwww/2.14
From:  user@host.ntnu.no
Content-type: application/x-www-form-urlencoded
Content-length: 150

type=phrase
&q=M%F8nster
&advanced=1
&cs=iso-8859-1
&type=hele%20uttrykket
&username=karinord
&submit=S%F8k
```

Eksempel 4-5 POST query mellom grensesnitt **[G1]** og grensesnitt-agent **[A1]** (noe simplifisert)

*Søket sendes til, og behandles av, grensesnitt-agenten **[A1]** som et POST query (HTML) **[Eks. 4-5]**. Fra grensesnittet får dermed grensesnitt-agenten informasjon om hva brukeren søkte på ("Mønster") og vi tar også utgangspunkt at brukernavnet til brukeren ("karinord") overføres samtidig.*

*Grensesnitt-agenten har ikke til oppgave å svare direkte på et slikt søk selv, men behandler dataene og sender forespørselen videre til informasjonsmegleren **[A3]**, ved hjelp av et kommunikasjons-format **[Eks. 4-6]** (her representert som XML).*

I informasjonsmegleren sjekkes normalt sett brukerinformasjon og profiler tilnyttet denne. I tillegg prosesseres det som måtte finnes av utfyllende informasjon om søket (for eksempel en forespørsel om å søke innenfor en spesiell kategori). I vårt eksempel er

```
<?xml version='1.0'?>
<search field="title">Mønster</search>
<username>karinor</username>
```

Eksempel 4-6 Dataflyt mellom grensesnitt-agent **[A1]** og informasjonsmegler **[A3]**

*det kun brukernavn som er interessant for denne agenten i første omgang. En forespørsel sendes derfor fra informasjonsmegleren til "brukerinfo og profiler"-agenten **[A5]**, **[Eks. 4-7]** for å hente ut nødvendige data om brukeren.*

```
<?xml version='1.0'?>
<function>Hent Brukerprofil Og Rettigheter</function>
<username>karinor</username>
```

Eksempel 4-7 Dataflyt mellom informasjonsmegler **[A3]** og brukerinfo/profiler **[A5]**

*Ønsket brukerinformasjon hentes av brukerinfo/profiler agenten **[A5]** ut fra databasen (f.eks. ved hjelp av et SQL query[18]) og sendes tilbake til informasjonsmegler (igjen som en XML fil) **[A3]**, **[Eks. 4-8]**.*

```
<?xml version='1.0'?>
<function>Hent Brukerprofil Og Rettigheter</function>
<username>karinor</username>
<result>
   <realname>Kari Nordmann</realname>
   <rights>
      <database title="forskdb">read+write</database>
      <database title="imgdb">read</database>
   </rights>
   <profile>
      <areas_of_interest>
         <item>Mønstergjenkjenning</item>
         <item>Bibliotek</item>
      </areas_of_interest>
   </profile>
</result>
```

Eksempel 4-8 Dataflyt mellom brukerinfo/profiler **[A5]** og informasjonsmegler **[A3]**

*Basert på søket som ble gjennomført ("Mønster"), brukerinformasjon (her: tilgangsrettigheter) og profilen til den aktuelle brukeren (her: interesseområder og fagfelt), dannes et eller flere egnede søk som sendes til overordnet samlingsagent **[A4]** **[Eks. 4-9]**.*

*Den overordnede samlingsagenten har, per definisjon, full oversikt over alle samlingene, og foretar søk i hver av samlingene **[D1]**. Hvordan og hva den skal søke i er beskrevet i metadata om samlinger **[D2]**. Den benytter også informasjon fra det pågående søket (her: søketermen og rettigheter til databasene). Overordnet samlingsagent mottar derfor søkeinformasjonen, og benytter samlingsinformasjonen **[D2]** for å mappe et*

18. Detaljene fra denne informasjonsutvekslingen beskrives ikke her, da en liknende informasjonsutveksling vil belyses senere i forbindelse med samlingsagentene og i punkt (9) og (10) i oversikten over informasjonsflyten.

```
<?xml version='1.0'?>
<search field="title">Mønster</search>
<origin>Web Interface, ID 1023</origin>
<profile>
   <rights>
      <database title="newsdb">read</database>
      <database title="forskdb">read/write</database>
      <database title="imgdb">read</database>
   </rights>
</profile>
```

Eksempel 4-9 Dataflyt mellom informasjonsmegler **[A3]** og overordnet samlingsagent **[A4]**

*sett med generelle felt-koder til database-spesifikke, for deretter å foreta søk i hver av de to tilgjengelige databasene ved hjelp av samlingsagentene **[A7.1]** og **[A7.2]** (henholdsvis **[Eks. 4-10]** og **[Eks. 4-11]**).*

```
<?xml version='1.0'?>
<query db_field="imgdb.image_title">Mønster</query>
```

Eksempel 4-10 Dataflyt mellom overordnet samlingsagent **[A4]** og en samlingsagent (1) **[A7.1]**

```
<?xml version='1.0'?>
<query db_field="forskdb.prosjekt_tittel">Mønster</query>
```

Eksempel 4-11 Dataflyt mellom overordnet samlingsagent **[A4]** og en samlingsagent (2) **[A7.2]**

*Hver av samlingsagentene **[A7.x]** er spesialskrevet for samlingen de representerer. Samlingsagent (1) går mot en SQL database, imens (2) går mot en ODBMS (objektorientert) database. Samlingsagent **[A7.1]** og **[A7.2]** sender derfor et spesifikt query til sine respektive databaser basert på domenekunnskap om databasen (**[Eks. 4-12]**, **[Eks. 4-13]**).*

```
select * from imagedb
       where image_title like '%Mønster%';
```

Eksempel 4-12 Dataflyt mellom samlingsagent (1) **[A7.1]** og tilhørende datalager **[D1.1]**

```
@forskdb.report{title="Mønster"};
```

Eksempel 4-13 Dataflyt mellom samlingsagent (2) **[A7.2]** og tilhørende datalager **[D1.2]**

*Det samlede resultatet den overordnede samlingsagenten **[A4]** får fra hver av samlingsagentene **[A7.x]** samles **[Eks. 4-13][Eks. 4-14]**, og sendes tilbake til informasjonsmegleren **[A3]** **[Eks. 4-15]**. Dataene sendes derifra umodifisert til grensesnittagenten som sendte forespørselen **[A1]** **[Eks. 4-16]**. Grensesnitt-agenten gjør om dataene til en egnet fremvisning på PDA (HTML) og sender disse dataene til grensesnittet **[Eks. 4-17]**.*

```
<?xml version='1.0'?>
<result>
     <object>
          <title>
               Mønsterdybde i vinterdekk -
               Illustrasjon
          </title>
          <binary_element mime="image/gif">
               SGVpIQoKRHUgaGFyIGZha3Rpc2sgbuUgZnVub
               mV0IGVuIGlra2UgYWtrdXJhdCBrcnlwdGVydC
               B0ZWtzdC4uLiA6KSBTZW5kIG1lZyBnamVybmU
               gZW4gbWFpbCEKCk9sZS1NYXJ0aW4sCm9sZUBt
               YXJ0aW4uYXMK
          </binary_element>
     </object>
</result>
```

Eksempel 4-14 Dataflyt mellom samlingsagent (1) **[A7.1]** og overordnet samlingsagent **[A4]**

```
<?xml version='1.0'?>
<result>
     <object>
          <title>
               Mønstergjenkjenning i praksis
          </title>
          <binary_element mime="application/pdf">
               VPhmdCBhdCBkdSBmYW50IGRlbm5lIG1lbGRpb
               mdlbiEgU2VuZCBtZWcgZW4gbWFpbCEgb2xlQG
               1hcnRpbi5hcyEgSGF2ZSBmdW4h
          </binary_element>
     </object>
</result>
<result>
     <object>
          <title>
               Anvendt Bioinformatikk, Gruppering av
               kontingar (contigs) i
               sekvenseringsprosjekt.
          </title>
          <binary_element mime="text/plain">
               I typisk sekvensering blir genomet
               oppdelt i små fragment, som blir
               sekvensert kvar for seg. Desse blir så
               sett saman att med bruk av ein
               assemblar. Resultatet frå ei slik
               assemblering er fleire kontingar
               (contigs). Eit grupperingsprogram
               prøver så å finna orientering og
               rekkefølge av desse kontingane, basert
               på sekvens og anna informasjon samla
               under sekvenseringsarbeidet.
          </binary_element>
     </object>
</result>
```

Eksempel 4-15 Dataflyt mellom samlingsagent (2) **[A7.2]** og overordnet samlingsagent **[A4]**

```
<?xml version='1.0'?>
<search field="title">Mønster</search>
<origin>Web Interface, ID 1023</origin>
<result>
     <object>
          <title>
               Mønsterdybde i vinterdekk -
               Illustrasjon
          </title>
          <binary_element mime="image/gif">
               [...]
          </binary_element>
     </object>
     <object>
          <title>
               Mønstergjenkjenning i praksis
          </title>
          <binary_element mime="application/pdf">
               [...]
          </binary_element>
     </object>47335896
</result>
<result>
     <object>
          <title>
               Anvendt Bioinformatikk, Gruppering av
               kontingar (contigs) i
               sekvenseringsprosjekt.
          </title>
          <binary_element mime="text/plain">
               [...]
          </binary_element>
     </object>
</result>
```

Eksempel 4-16 Dataflyt fra overordnet samlingsagent **[A4]** via inf. megler **[A3]** til gr.snitt-agent **[A1]** (fork.)

Eksempel 4-17 Resultat på PDA

4.4 Teknologi

I denne delen omtales kort noen aspekter som kan være nyttig å ta hensyn til når det skal velges teknologi for implementering av et digitalt bibliotek basert på arkitekturen.

Det er viktig å merke seg at de spesifikke software- eller hardware-løsninger som kommer frem her er ment å gi en pekepinn fremfor å diktere noen ideell teknologi for implementasjon av arkitekturen. Det vektlegges derfor generelle egenskaper den underliggende teknologien bør ta hensyn til og understøtte.

4.4.1 Databaseteknologi

Arkitekturen setter ingen krav til hva slags lagringsmetode som benyttes for samlingene. En samling skal uten problem kunne bestå av et sett med PDF **[W_PDF]** filer i en filstruktur, en tabell i en relasjonsdatabase, eller en XML-fil **[W_XML]**. I mange tilfeller er man derimot, av rent pragmatiske årsaker, ikke istand til selv å velge teknologi. En databaseteknologi kan allerede være i bruk, eller at samlingene har egenskaper som krever en spesiell teknologi.

I de tilfellene der man har mulighet til å velge fritt blant teknologi, er det viktig å vurdere samlingens egenskaper, for å på best mulig vis finne en egnet teknologiplatform. Når det kommer til ren databaseteknologi legges det like mye til rette for å benytte objektorienterte databaser (ODBMS), objekt-relasjons-databaser (ORDBMS) som relasjonsdatabaser (RDBMS).

4.4.2 Integrering

Integrering mellom de forskjellige delene av systemet vil kunne foregå både som backend-prosesser (eksempelvis ved hjelp av run-time script som Perl **[W_PRL]** eller kompilerte applikasjoner), integrering med f.eks. Corba **[W_COR]** eller Microsoft .NET **[W_NET]** eller som en direkte del av grensesnitt-teknologien (PHP **[W_PHP]**, Microsoft ASP **[W_ASP]**).

En av grunntankene i arkitekturen er at integrering skal foregå på samlingsnivå, og at hver samling behandles separat, men med et felles grensesnitt til resten av systemet. Visse typer integreringsteknologi kan gjøre uavhengighet mellom samlinger vanskeligere, og det er derfor viktig å evaluere dette aspektet ved valg av integreringsteknologi.

4.4.3 Grensesnitt

Hvert enkelt grensesnitt skal være, i forhold til kravene arkitekturen setter, i stor grad uavhengig av hverandre. Dette betyr ikke på noen måte at de må benytte forskjellig teknologi, så lenge dette ikke setter begrensninger eller skaper avhengigheter i forhold til andre grensesnitt.

4.5 Oppsummering

Det har i dette kapittelet blitt sett på et forslag til arkitektur for digitale bibliotek.

Arkitekturen baserer seg på en tredelt modell med følgende oppbygning:

- ***Grensesnitt***
  - tar for seg forholdet til brukeren.
- ***Datalager***
  - tar for seg lagring av brukerdata, samlingsmetadata og samlinger.
- ***Agenter***
  - tar for seg alle informasjonsflyt i arkitekturen.

Det er blitt sett på hvordan de enkelte delene av et digitalt bibliotek kan samarbeide for å oppnå mål om dynamiske, og personifiserte grensesnitt presentert på ulike grensesnitt med et dynamisk sett med samlinger.

Det refereres videre til *Kapittel 5*, som tar for seg prototyping av arkitekturen, *kapittel 7* som tar for seg evaluering av arkitekturen.

5 Prototype

En prototype er utviklet i forbindelse med arbeidet med avhandlingen **[Fig 5-1]**. Prototypen har i to spesifikke formål. For det første danner utviklingen av den mye av erfaringen som ligger til grunn for arkitekturen (ref. kapittel 4). Prototypen vil også benyttes til å forklare sentrale egenskaper og grunnprinsipper ved arkitekturen (ref. kapittel 1.1.5, *"Mål med prototypen"*).

Vi tar her for oss hovedtrekk ved prototypen. Gjennomgangen begrenses i stor grad til kun å se på relevante deler i forhold til arkitekturen som gjennomgås i kapittel 4.

Gjennomgangen starter med en kort systembeskrivelse som tar for seg de målene som prototypen tar utgangspunkt i, og en definisjon av prototypens målgruppe. Det vil deretter redegjøres for implementeringen av hovedkomponentene i fra arkitekturen (ref. *kapittel 4.1*) **agenter**, **grensesnitt** og **datalager**.

Det pekes videre på punkter rundt skalerbarhet i denne type systemer, spesielt relatert til prototypen, og det vil kort redegjøres for teknologien som er brukt.

Kapittelet avsluttes med et casestudie som redegjør både for brukerens forhold til systemet, og systemets funksjonalitet på grunnlag av de valgene brukeren gjør.

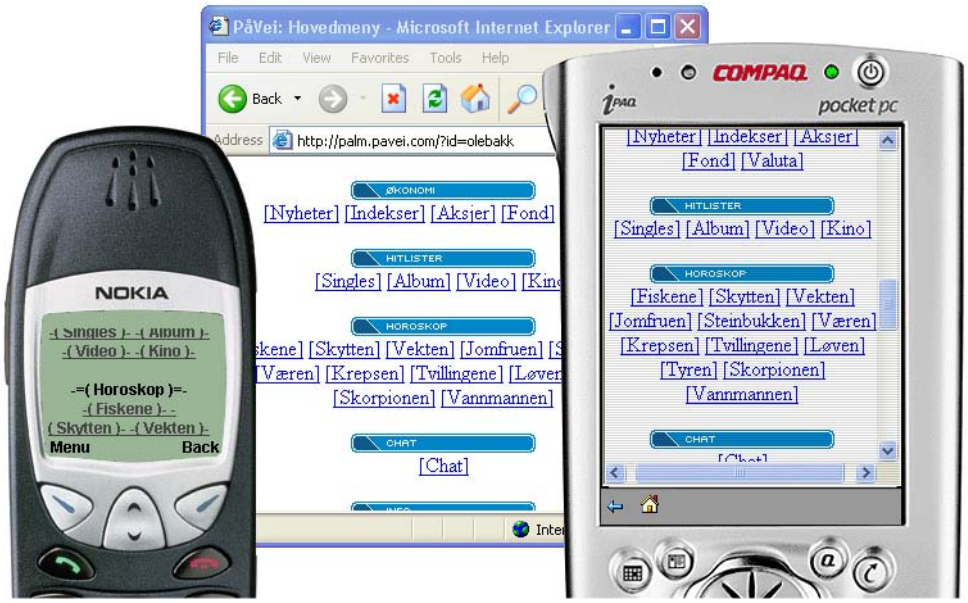

Fig 5-1 PåVei Underholdning med tre ulike grensesnitt

5.1 Systembeskrivelse

Hovedmålsettingen med prototypen, rent bortsett fra å være en mulig implementasjon av arkitekturen, kan beskrives slik:

> *"Systemet skal være et tilbud for underholdningsrelaterte tjenester på PDA. I hovedsak dagsaktuell informasjon innenfor emner som TV, kino, musikk og økonomi."*

Prototypen (heretter også løst referert til som *"systemet"* eller "*PåVei Underholdning*") er i bunn og grunn tiltenkt som et komplett tilbud til en bestemt brukergruppe (eiere av en PDA), med ulike samlinger presentert i et grensesnitt på en PDA.

Det er en forutsetning at systemet er så brukervennlig og dynamisk som mulig. Hver bruker bestemmer derfor selv så mye av sitt eget grensesnitt som praktisk mulig. Konfigurering av tjenester skjer i hovedsak via en PC med en webbrowser, og samlingene kan også aksesseres direkte derifra. Fra grunnen av legges det dessuten, så godt det er mulig, opp til at fremtidige grensesnitt skal kunne legges til på en enkel måte.

5.2 Bakgrunn og historie

En pragmatisk og statisk prototype startet implementasjonsfasen, på grunnlag av en tidlig målsetning om å legge tv-programmet over på PDA, og ved hjelp av et pragmatisk sett med tilgjengelige verktøy. I denne versjonen foregikk alt arbeid på PDAen og man hadde hverken kontroll over utseendet til grensesnittet, eller over hva slags informasjon som skulle vises.

Det dukket etterhvert opp flere informasjonsobjekter som kunne egne seg til dette formålet (kinoguide, aksjeresultater, valutakurser, værmeldinger etc.), og som også ble lagt inn i systemet. PDAens teknologiske begrensninger (i hovedsak lagringsminne, størrelse på skjerm og hastighet på mobil kommunikasjon) gjorde det derimot vanskelig å få presentert all denne informasjonen på en fornuftig måte.

Fig 5-2 Skjermdump fra tidlig utgave av prototypen (PalmOS)

En ny prototype ble derfor utarbeidet som en løsning på disse problemene. En egen side på PDAen ble satt opp med en del dialogbokser som gjorde det mulig for brukeren og velge de TV-kanalene og kinobyene brukeren var interessert i. Disse tjenestene ville da bli aktivisert neste gang PDAen ble oppdatert (synkronisert) mot systemet.

Denne prototypen ble testet ut med 5 brukere og ble godt mottatt, men ble likevel lagt litt på is til fordel for andre mulige innfallsvinkler til avhandlingen. Prosjektet gikk først videre etter at en søkerobot tilfeldigvis hadde funnet prototypen, og en ganske stor mengde mennesker helt uventet tok kontakt med forfatteren, i håp om at prosjektet ikke var dødt.

Prosjektet fortsatte derfor, motivert av entusiasmen til endebrukerne. Denne gangen ble oppgaven vanskeligere, i og med at det ble foreslått tjenester som ville gjøre det umulig å konfigurere systemet kun via en PDA. Dette blant annet fordi bare størrelsen på informasjonen som var nødvendig for å konfigurere tjenesten, ville overgå kapasiteten til mange PDA enheter i markedet.

Tredje prototype ble derfor utviklet fra bunnen av, basert på en modell hvor alle aspektene av tjenesten ble separert ut ifra en objektorientert (og senere agentbasert) tankegang, og fokus ble lagt på å dele opp all informasjon inn i "datapakker" som brukeren kunne plukke fra, for å sette opp sitt egnede grensesnitt.

Det refereres til *Appendix A* (side 93) for en utdypet forklaring på systemets historie og nåværende status, og *kapittel 5* for en grundig gjennomgang av prototypen.

5.3 Målgruppe for prototypen

Systemet er laget med et innhold som antas å lokke en stor del av de som er "early adopters" og per i dag eier en PDA. På grunn av en fortsatt relativ høy kostnad, og en applikasjonsbase myntet på et voksent publikum, ventes de fleste brukerne å være i aldersgruppen 18-40 år. Et flertall av brukerne regnes å være menn, men det eksisterer ikke noe statistisk grunnlag for dette, utenom en andel av tilbakemeldinger forfatteren har motatt fra mannlige brukere, som utgjør omlag 90% av total kommunikasjon med brukere, ut ifra et statistisk grunnlag på omlag 6% av brukermassen (350 mail).

Fordelt på tjenester kan TV-programmet regnes for å dekke hele spekteret av brukere. Tjenester som chat, horoskop, kino og hitlister antas være pådrivere for den yngste delen av denne gruppen, mens værvarsel, aksje, fond og valutakurser tar det eldste delen av gruppen.

Som en indikasjon på dette, hadde tjenesten i juni 2002 ca. 5500 aktive brukere. 660 av disse abonnerte på en kinooversikt, 1600 benyttet en av økonomi-modulene, 1480 benyttet hitlister, 1980 hadde værmeldingen i sin profil, 3850 abonnerte på tv-programmet og 1250 abonnerte på horoskopet.[19]

19. Tallene sier ikke nødvendigvis noe om at tjenestene virkelig benyttes/leses av endebruker, men kan fungere som en indikasjon på populariteten til de enkelte tjenestene.

5.4 Agenter

Informasjonsagentene står til sammen for all kommunikasjon mellom grensesnittene til systemet og datalageret. Det finnes dermed en mengde mindre agenter i prototypen.

Det er her imidlertid viktig å presisere at ikke agentene i modellen fra kapittel 4 representeres på samme måte i prototypen. Grunnen til dette er at arkitekturen er blitt laget i stor grad som et resultat av et parallelt arbeid med prototypen, og at det er blitt gjort arbeid med å raffinere arkitekturen som ikke er representert i prototypen (ref. kapittel 7.1.2, *"Arkitekturens forhold til prototypen"* for en utdypet diskusjon om dette forholdet)

Det kan skilles mellom tre klasser av agenter i prototypen:

- ***Backend agenter***
- ***Grensesnittagenter***
- ***Samlingsagenter***

Sett i forhold til arkitekturens modell **[Fig 4-2]**, tilsvarer backend agenter elementene **[A3]** og **[A4]**, grensesnittagenter elementene **[G1]** og **[G2]** og samlingsagenter elementene **[A5]** og **[A7]**.

5.4.1 Backend agenter

Backend agenter er i stor grad implementert i Perl **[W_PRL]** og tar for seg alt arbeid innenfor systemet som ikke involverer et tidskritisk forhold til en bruker.

Henting av data fra eksterne kilder og leverandører
En stor del av informasjonen som systemet tar seg av er kontinuerlig oppdatert informasjon (f.eks. tv-program, værmelding og aksjekurser) og det er derfor viktig at denne informasjonen hentes til rette tider (for tv-program er det gjerne nok om det oppdateres hver dag, aksjekurser bør ikke være mer enn 15 minutter gamle).

Kvalitetssikring
Ofte er det ikke nok å bare hente informasjonen. Man må også være sikker på at den informasjonen man henter er det man venter å få. Denne formen for kvalitetssikring er i dette systemet implementert ved at noen ansvarlige får beskjed (f.eks. via e-post) om at f.eks. en XML fil ikke var tilgjengelig for prosessering. En manuell oppdatering eller endring i rutiner kan dermed foretas.

Prosessering av data
Noen ganger vil man bearbeide data på grunnlag av to eller flere samlinger for å lage en helt ny samling. Et eksempel på dette i prototypen, er listen over kinofilmer som er mest populære. Selve listen hentes fra HitCompany, og kinoinfo (kinoprogram og omtaler av disse) hentes fra FilmWeb. En agent sjekker om det finnes noen omtale for hver av filmene i listen, og man ender opp med et resultat hvor man kan se på omtalen til filmer direkte fra denne hitlisten.

Prosessering av brukerspesifik informasjon
Disse henter enten kontinuerlig eller ved behov ned i informasjon som brukeren har spesifisert at er innenfor hans interessedomene. En funksjon som gjør akkurat dette i prototypen er "programpeileren" som finner tak i tv-programmer som matcher en gitt profil og gir tilbakemeldinger om disse ut ifra hvilken dag det er, tidspunkt på dagen og hvilke tv-kanaler brukeren har tilgang på.

5.4.2 Grensesnittagenter

Grensesnittagenter er utviklet spesifikt for et grensesnitt, og sitter med domenekunnskaper om hvordan dette grensesnittet på best måte kan utnyttes. De to grensesnittene som eksisterer i systemet er implementert i PHP.

Det refereres til *Appendix B* for implementasjon av WEB-grensesnittet og *Appendix C* for implementasjonen av PDA-grensesnittet.

5.4.3 Samlingsagenter

Det finnes ikke implementert noen spesifikk samlingsagent i prototypen, men funksjonaliteten til denne er isteden definert ut ifra en API[20] for hvordan hvert av grensesnittene selv vil finne ut hvilke samlinger som er tilgjengelige (samlingsmetadata og brukerdata) samt hvordan disse aksesseres, via et datalager. Mye av denne "agentens" funksjonalitet ligger derfor i datalageret (MySQL **[W_SQL]**), samt at det eksisterer en sentral konfigureringsfil som alle grensesnittagentene skal benytte seg av, for finne tak i datalageret.

Det refereres videre til *Appendix B, C, D* og E for kildekoden til prototypen.

5.5 Grensesnitt

Prototypen har to grensesnitt tilgjengelige for brukeren; et utviklet for å vises på en **PDA** og et for å vises på **WEB**.

WEB-grensesnittet har som målsetning å være brukerens inngangsportal til systemet. Her foretas det meste av brukerinteraksjon, blant annet administrering av brukerkonto (opprette og slette bruker), samt konfigurasjon (endre på hvilke tjenester som ønskes).

I motsetning til web-grensesnittet som det ventes at brukeren benytter kun sporadisk, er *PDA-grensesnittet* det mer "daglige" grensesnittet brukeren har til systemet. Her finnes det ideelt sett alltid oppdatert informasjon som brukeren har interesse av.

Som, noe forenklet, illustrert i **[Fig 5-3]**, føres informasjonen fra brukeren via WEB-grensesnittet inn til en informasjonsmegler, som igjen er ansvarlig for å få informasjonen ut til brukeren via PDA-grensesnittet.

20. API står for "*application program interface*" og er, litt simplifisert, et sett med rutiner, protokoller og vektøy for å bygge uavhengig softwarekomponenter som skal fungere sammen.

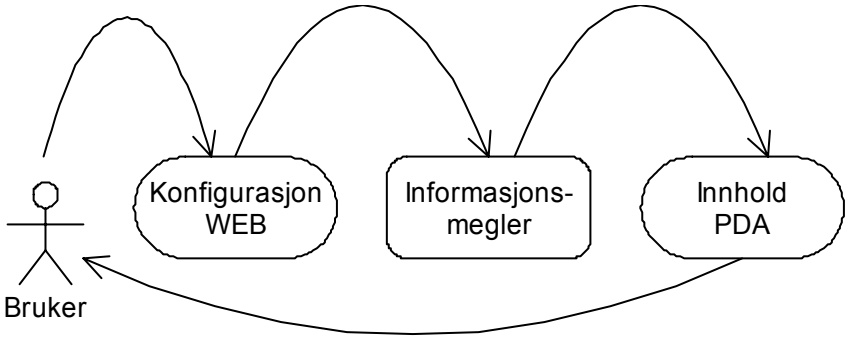

Fig 5-3 Informasjonsflyt mellom bruker, WEB- og PDA-grensesnitt

Det er verdt å merke seg at WEB-grensesnittet, om nødvendig, kan brukes til å lese av informasjonen. Det finnes blant annet en preview-funksjon som gjør det mulig å se hvordan resultatet blir seende ut på en PDA. På samme måte kan PDA grensesnittet påvirke systemet i mindre grad. Dataflyten, fra brukerens side, foregår likevel i hovedsak som illustrert.

For en mer grundig gjennomgang av informasjonsflyten i prototypen og i arkitekturen generelt, refereres det til kapittel 4.3, *"Informasjonsflyt"* og kapittel 5.7, *"Casestudie av prototypen"*.

5.5.1 Web-grensesnitt

I WEB-grensesnittet **[Fig 5-4]** foregår det meste av brukerinteraksjon i forhold til oppsett av brukerprofil. Man må først registrere seg som bruker, og deretter kan man sette opp de tjenestene og informasjonsagentene man ønsker å aktivere. Til sammen utgjør denne informasjonen brukerens *personlige profil*.

Brukeren kan i dette grensesnittet velge blant annet:

- *Hvor mange dagers tv-program man ønsker å ha tilgjengelig*
- *Hvilke tv-kanaler man har*
- *Hvilke tv-programmer man ønsker å bli påminnet om*
- *Hvilke kinoer man ønsker å motta kinoprogram for*
- *Hvilke aksjer, fond, opsjoner man har og eventuelt verdi/opsjonskurs*

Denne informasjonen lagres i databasen, og vil benyttes senere for å gi relevant informasjon tilbake til brukeren.

5.5.2 PDA-grensesnitt

PDA grensesnittet blir til ved at informasjonen brukeren har definert, prosesseres og lagres som et sett med sider på brukerens PDA, ved hjelp av å synkronisere den med en PC. En hovedside presenterer tjenestene du abonnerer på og har linker til undersider med informasjon.

Informasjonen presenteres med et forholdsvis enkelt grensesnitt **[Fig 5-5]**. Målet med grensesnittet er at det, av flere grunner, skal være så enkelt som mulig. Først og fremst har en PDA gjerne relativt *begrenset lagringskapasitet* og det er derfor viktig å holde

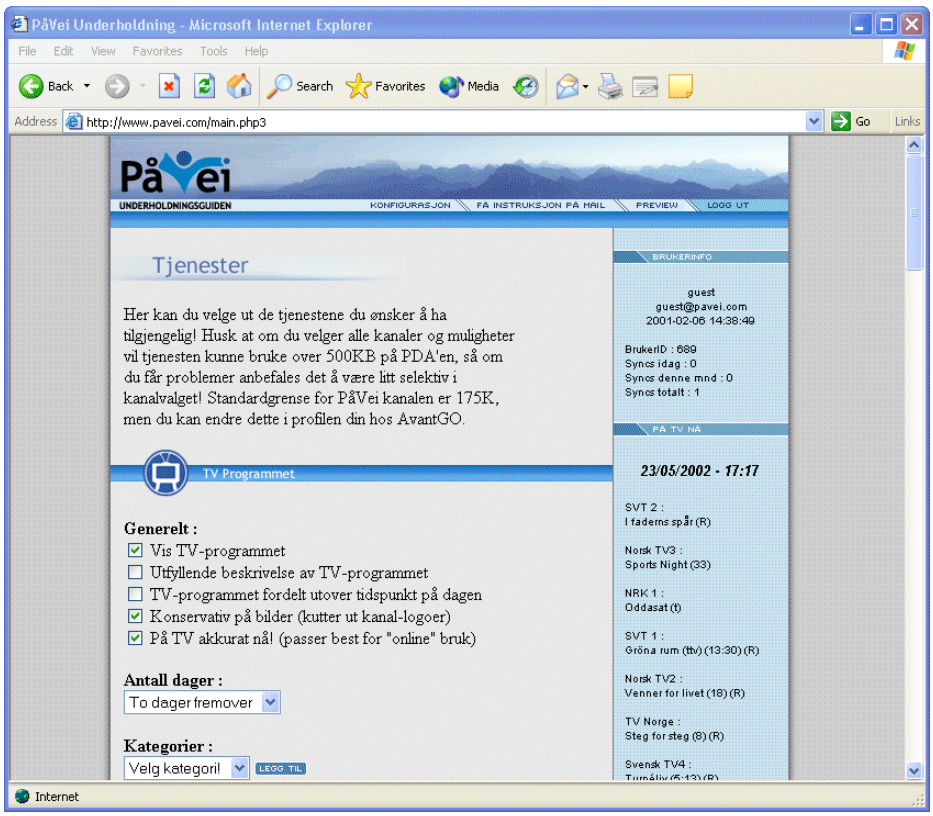

Fig 5-4 WEB grensesnitt

Fig 5-5 PDA Grensesnitt (PocketPC)

datamengden nede. På samme måte er det fornuftig med små datamengder siden det da tar *kortere tid å overføre dataene*. En tredje grunn er at en PDA gjerne har en veldig *begrenset oppløsning* på skjermen, samt ofte har *begrenset med farger* noe som også setter krav til et relativt beskjedent grensesnitt.

Også gjennom PDA grensesnittet finnes det interaktivitet med resten av systemet, selv om den er veldig begrenset. Brukeren kan blant annet forandre sitt profilnavn, og bytte om på utseendet til grensesnittet ut ifra noen ferdige profiler.

5.5.3 Forholdet mellom WEB og PDA

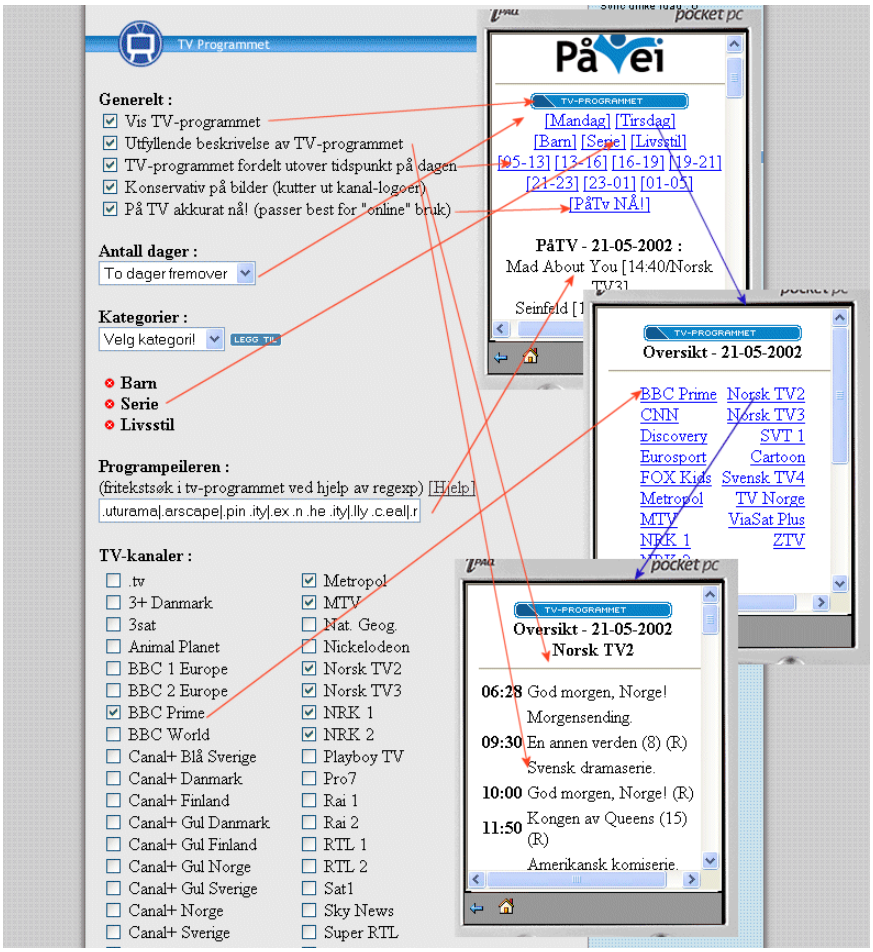

Fig 5-6 WEB til PDA - blå piler og haker er brukervalg - røde piler er system-valg

Figuren **[Fig 5-6]** viser forholdet de to grensesnittene har til hverandre. De røde pilene viser hvordan prototypen visualiserer brukerens valg, som representeres med blå piler og haker.

5.6 Datalager

Datalageret er i praksis databasen(e) **[BS2]** fra den abstrakte modellen **[Fig 4-2]** gjennomgått tidligere. Her ligger all informasjon for systemet lagret. Dette innebærer blant annet:

- ***Understøttende datalager***
  - **Brukerdatabase**
  - **Tjenestedatabase (personlige profiler)**
  - **Samlingsmetadata**
  - **Systemlog (løpende informasjon om bruken av tjenesten)**
- ***Samlinger***
  - **TV-program oversikt**
  - **Økonomi**
  - **Vær**
  - **Horoskop**
  - **Hitlister**
  - **Chat**

I tillegg til dette har datalageret et grensesnitt som gjør dataene tilgjengelig for resten av systemet. Prototypen benytter MySQL **[W_SQL]** til all datalagring.

Det gjennomgås her to av de understøttende datalagerene, **brukerdatabase** og **tjenestedatabase**, og kort to av samlingene, **tv-program** og **økonomi** som er benyttet i systemet. Det blir sett på hvordan datalagerene er bygget opp, hvordan de benyttes, hvilke agenter som er knyttet opp til dem, og hvordan disse fungerer i forhold til grensesnittet og brukeren.

Det refereres til *Appendix B* for ytterligere informasjon om de enkelte database-elementene i implementasjonen.

5.6.1 Brukerdatabase

Brukerdatabasen **[Tabell 5-1]** er et enkelt datalager hvor informasjon om brukere tilknyttet systemet lagres.

Hver bruker av systemet identifiseres ved hjelp av en brukernavn/passord kombinasjon. Brukeren bestemmer selv både brukernavn og passord ved første registrering inn i systemet. Brukeren bes også om å oppgi en gyldig e-post-adresse, da deler av registreringen foregår over e-post.

Utover disse tre feltene benyttes et enkelt identifikasjons-nummer (BrukerID) som unik identifisering av brukere internt i systemet og i de ulike datalagrene.

Det er flere grunner til at ID-nummer (BrukerID) benyttes isteden for brukernavnet eller e-post-adressen som primærnøkkel i databasen. Tekstfelt tar for det første uforholdsmessig mye plass i forhold til en enkel integer-verdi noe som kan føre til at systemet skalerer dårligere enn nødvendig.

DatabaseID	Beskrivelse
BrukerID	ID-nummer (tall) som benyttes internt i systemet for unik identifisering.
Brukernavn	Kort brukernavn uten separatorer (ikke ekte navn) som identifiserer brukeren - også dette unikt på samme måte som ID feltet.
Passord	Et passord valgt av brukeren selv og verifisert via e-post.
E-post	Autentisering av brukeren foregår via e-post-adressen

Tabell 5-1 Brukerdatabasen

Bruk av ID-nummer som identifikator gir også muligheter til enkelt å benytte en annen intern identifikator senere uten at endebrukeren blir påvirket eller merker noe til endringen - eller for en bruker å endre sitt brukernavn uten at systemet påvirkes i stor grad.

Passord koblet opp mot brukernavn benyttes for å autentisere brukeren.

E-post er en alternativ form å nå brukeren på. Benyttes primært i autentiserings- og oppstartsfasen (første gangen brukeren tar systemet i bruk), men kan få også andre bruksområder senere.

BrukerID	Brukernavn	Passord	E-post
3714	olebakk	hemmelig[a]	omb@pdapath.com[b]
3715	RogerHermansen	trustNo1	RH@dfh.no
3716	trondbre	4567	gnan@hotmail.com
3717	supermann	qwerty	trezz_77@online.no

Tabell 5-2 Utdrag fra brukerdatabasen

a. Passord vil i databasen opptre i kryptert form
b. E-post adressene er fiktive

5.6.2 Tjenestedatabase

I sammenheng med brukerdatabasen finnes det også en tjenestedatabase **[Tabell 5-3]** som inneholder alle variabler tilknyttet en bruker. Finnes det ingen data tilknyttet en bruker, mottar brukeren heller ikke noe data på sin PDA. De elementene som beskrives i denne databasen er derfor hvilke tjenester og funksjonalitet brukeren ønsker.

DatabaseID	Beskrivelse
BrukerID	ID koblet opp mot brukerdatabasen for å identifisere brukeren.
TjenesteID	ID-nummer (tall) som benyttes internt i systemet for unik identifisering av elementet.
Tjeneste	Hovedkategorien tjenesten hører under. (fond, kino, økonomi osv.)
Type	Utdypende beskrivelse av type (en bestemt aksjeticker, fondsleverandør, en tv-kanal osv.)
Parameter	Valgfri underspesifikasjon av tjeneste (f.eks. antall aksjer)

Tabell 5-3 Tjenestedatabasen

I steden for å lage en unik database for hver av tjenestene i systemet, brukes en generell modell. Dette gjøres av to grunner. Først og fremst for å slippe å oppdatere basis databasestruktur når nye tjenester blir innført i systemet. Dernest for å gjøre det enkelt å skrive generelle agenter som skal forholde seg til dataene.

Databasen inneholder en tredelt domenespesifikk beskrivelse av tjenesten. **Tjeneste**-feltet inneholder det første "laget" i denne strukturen som beskriver første nivå. I dette tilfellet spesifiserer den konkret hvilken agent som tar for seg forespørselen, men det er ingenting i veien for at det kan være flere agenter som tar for seg samme element og benytter det i forskjellige sammenhenger, eller at en agent tar for seg flere typer. Neste felt, **type**, inneholder en utdypende beskrivelse av tjenesten, og **parameter** spesifiserer videre (om nødvendig) hva slags informasjon det er brukeren vil ha tak i.

TjenesteID	BrukerID	Tjeneste	Type	Parameter
31759813	3714	Hovedside	Horoskop	Væren
31759814	3714	Hovedside	Kino	
31759815	3714	Hovedside	Tv	
31759816	3714	Kino	Oslo	
31759817	3714	Kino	Trondheim	
31759818	3714	Tv	Dager	2
31759819	3714	Tv	Kanal	NRK
31759820	3714	Tv	Kanal	TV2

Eksempel 5-1 Utdrag fra en tjenesteoversikt for en gitt bruker

5.6.3 TV-Program

Denne samlingen består av tv-programmet til i alt rundt 85 TV-kanaler. Informasjonen hentes fra flere kilder, både innholdsleverandører som NTB Pluss **[W_NTB]**, og i noen tilfeller direkte fra websidene til TV-kanalen. **[Tabell 5-4]** viser utformingen av samlingen og en kort beskrivelse av de forskjellige feltene.

DatabaseID	Beskrivelse
ID	ID-nummer (tall) som benyttes internt i systemet for unik identifisering av elementet.
Kanal	Tv-kanal (NRK1, TV2, TV3 etc.)
Tittel	Programtittel ("Frasier", "Livets Lyse Side", "Friends", etc.)
Kategori	Klassifisering av programmet ("Serie", "Film", "Sport" etc.)
Beskrivelse	Kort omtale eller beskrivelse av programmet
Tid	Tidspunkt på dagen det aktuelle programmet starter
Dato	Dato for sending

Tabell 5-4 Databasefelter for "TV Program"

Agenter som benytter denne samlingen:

- ***Dagsoversikt***
  Denne agenten samler sammen det brukeren ønsker av daglige tv-oversikter, basert på kriterier som; hvor mange dager i forveien brukeren ønsker informasjon, hvilke kanaler brukeren vil ha tilgang til, om grafikk ønskes på sidene, og om omtaler av programmene ønskes.
- ***Tidsoversikt***
  Gir en oversikt som ovenfor, men som her presentes sortert på tidspunkt i steden for kanal, og kun dagens program.
- ***På TV nå***
  Samler sammen en oversikt over hva som går på TV akkurat nå på de forskjellige tv-kanalene brukeren har tilgang til.
- ***Programpeiler***
  Lager en liste over "dagens høydepunkter" på tv basert på endel stikkord som brukeren har oppgitt, samt selvsagt hvilke tv-kanaler brukeren har tilgang til.
- ***TV***
  En overordnet agent samler sammen og gir fra seg all informasjonen fra aktuelle agenter (basert på brukerens valg, f.eks. fra dagsoversikt, tidsoversikt og programpeiler) på etterspørsel fra agenten som jobber direkte med brukergrensesnittet.
- ***Datainnhøster***
  Henter inn data fra aktuelle kilder og oppdaterer databasen på daglig basis.

5.6.4 Økonomi

Økonomidelen består av aksjekurser, fondkurser og valutakurser. Aksjekursene er oppdatert hvert 15. minutt, mens fond og valuta oppdateres 1-2 ganger pr. dag. For aksjer og fond setter man inn hvor mange andeler man har, og dermed holder man rede på den totale verdien av sine investeringer. For opsjoner setter man inn antall og opsjonskurs, og får ut hvor mye man vil tjene ved å innløse dem ved dagens kurs.

[Tabell 5-5] viser utformingen av fond-samlingen. Det refereres til *Appendix B* for komplett oversikt over databasene.

DatabaseID	Beskrivelse
ID	ID-nummer (tall) som benyttes internt i systemet for unik identifisering av elementet.
Fond	Fondsleverandør og navn på fond
Dato	Sist oppdatert dato
Kurs	Dagens kurs (kr. pr. andel)
Endring	Endring siden 31.12 i fjor
SisteÅr	Endring de siste 12 mnd
Siste2År	Endring de siste 24 mnd

Tabell 5-5 Databasefelter for "Fond"

5.7 Casestudie av prototypen

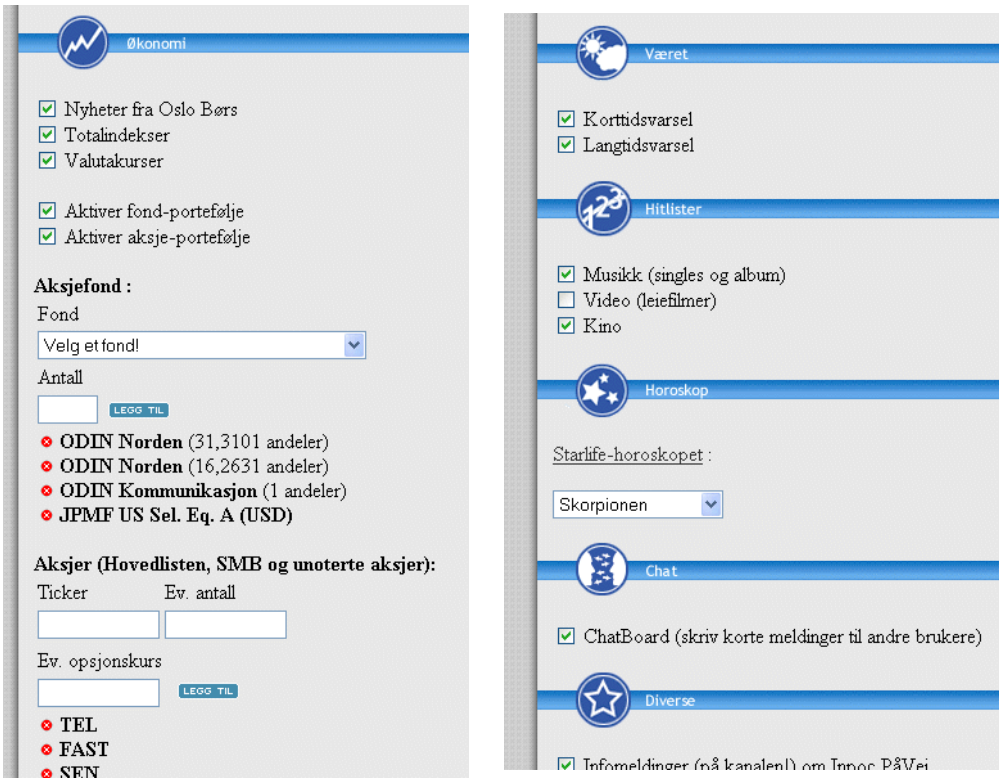

Fig 5-7 Konfigurering av tjenester i WEB grensesnitt.

[Fig 5-7] viser noen av mulighetene brukeren har for å konfigurere innholdet på sin PDA. For å illustrere hvordan systemet vil kunne brukes, vil en tenkt brukersituasjon beskrives i detalj, ut i fra to aspekter.

- ***Brukervinkel***
  - tar for seg brukerens situasjon. Et tenkt scenario illustreres punktvis hvor en bruker som er ukjent med dette spesifikke systemet, tar det i bruk for første gang.
- ***Systemvinkel***
  - ser på hvordan systemet tolker og bearbeider brukerens reaksjoner.

5.7.1 Brukervinkel

I en tenkt brukersituasjon har vi en kvinne, Sissel, på 35 år som jobber i IT-bransjen. Hennes hverdag er stresset, og krever derfor nøye planlegning både privat og på jobb. Der hun arbeider benytter både hun og de hun jobber sammen med en PC-basert tidsplanlegger for å holde rede på egne og andres møter. Hun har i den forbindelse også en PDA som synkroniseres med denne informasjonen, slik at alle avtalene både kan leses og noteres på denne.

Samtidig som disse dataene synkroniseres, lagres også en del informasjon på hennes PDA gjennom informasjonsformidleren AvantGO **[W_AVA]**. Det hun fra før har valgt å motta er siste nyheter fra VG og Aftenposten. Disse nyhetene oppdateres dermed hver gang hun synkroniserer, og kan leses ved behov uten å være tilkoblet internett.

Sissel ønsker nå å abonnere på PåVei Underholdning. Hun finner tjenesten PåVei hos AvantGo, legger denne til tjenesteoversikten sin, og synkroniserer sin PDA.

Fig 5-8 Bruksanvisning, PåVei / PDA

Sissel møtes på sin PDA av en bruksanvisning som forklarer hvilke steg som står igjen før hun kan benytte tjenesten **[Fig 5-8]**. Grunnen til at hun ikke kan benytte tjenesten umiddelbart, er at PåVei krever at hun registrerer seg, og setter opp en profil, for å benytte tjenesten.

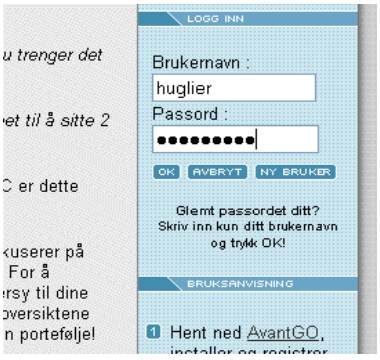

Fig 5-9 Login dialog

Vel inne på systemets hovedside (ref. **[Fig 5-4]**, side 59) finner hun henholdsvis en beskrivelse av tjenestene som tilbys, en kort bruksanvisning, og en boks hvor man kan fylle inn sitt brukernavn og passord **[Fig 5-9]**.

På de samme sidene kan man også velge å opprette en ny bruker - noe Sissel velger å gjøre, i tråd med bruksanvisningen. Fire dialogbokser dukker da opp, og ber henne fylle inn ønsket brukernavn, passord (to ganger), samt e-post-adresse. Hun prøver

først å registrere *sissel* som brukernavn, men det viser seg å være opptatt og hun får tilbakemelding om å prøve igjen med et annet brukernavn. Hun velger derfor *sanilsen* isteden, noe systemet aksepterer og gir tilbakemelding om at er godtatt.

Noen sekunder senere kommer det en bruksanvisning til Sissel pr. e-post.**[Eks. 5-2]**.

```
To: sissel@testbruker.com
From: brukerregistrering@pavei.com
Subject: Registrering for "sanilsen"

Velkommen som PåVei bruker!

Bruksanvisning:

1. Gå til http://www.pavei.com/go/?id=sanilsen
2. Du vil bli videresendt til AvantGO - logg deg inn
   om nødvendig.
3. Gå tilbake til http://www.pavei.com/, logg deg inn,
   og sett opp det du trenger av tjenester!

        Brukernavn : sanilsen
        Passord : mittPaSS

4. Sykroniser PDA og skriv inn ditt brukernavn på PDA.
5. Synkroniser igjen, og alt skal være i orden!

   Lykke til!

PåVei Underholdning
mail@pavei.com
```

Eksempel 5-2 Bruksanvisning fra systemet (e-post)

Denne forklarer i korte trekk hva man skal foreta seg for å få aktivert tjenesten, enten man har gjort som Sissel og lagt til kanalen hos AvantGO før man startet, eller har gått direkte til websidene.

Sissel gjør som hun blir instruert, og trykker på en weblink i mailen som sender henne til AvantGO, hvor hun logger seg på (hun har allerede en brukerkonto hos AvantGO) og følges igjennom en ledet prosess hos AvantGO. Siden hun allerede har PåVei i sin profil behøver hun gjøre minimalt her, og går derfor tilbake til websidene til tjenesten, hvor hun nå logger seg inn med sitt nye brukernavn.

Vel inne møtes hun med en rekke spørsmål (ref. **[Fig 5-7]**, side 65). Ut ifra informasjonen som samles inn her blir en spesifikk og målrettet informasjonsmengde utvalgt og distribuert til henne hver gang hun synkroniserer sin PDA.

Siden hun ikke har kabel-tv, ønsker Sissel kun å ha tv-programmet for NRK1, NRK2, TV-Norge og TV2 tilgjengelig på sin PDA. Hun er også veldig interessert i naturprogrammer og nyheter, og setter derfor opp disse to som interessefelt. Systemet gir også mulighet for å i fri tekst[21] skrive programtitler man er interessert i. Her noterer Sissel "60 Minutes".

21. For brukeren vil det kunne utarte seg som et fritekst-søk, men I praksis er det her snakk om et RegExp søk i databasen.

Hun velger videre at hun bor i Bergen under kino-alternativet, for også å motta et oppdatert kinoprogram. Hva som er aktuelt på musikk og filmfronten er hun derimot ikke interessert i å følge med på, og det samme gjelder horoskopet. Disse funksjonene velger hun dermed bort.

Sissel er en aktiv fond-sparer og har dessuten 100 opsjoner i firmaet hun jobber i. Denne informasjonen legger hun derfor inn i systemet. Hun setter også opp at hun ønsker å følge med på nyheter fra Oslo Børs og at hun vil ha oppdaterte valutakurser tilgjengelig, og ser seg ferdig med konfigurasjonen og lagrer sine innstillinger.

Sissel tar igjen frem sin PDA, noterer sitt brukernavn på den, og synkroniserer igjen med datamaskinen. PåVei er da tilgjengelig på hennes PDA med oppdatert informasjon fra hennes interesseområder. Hver gang "60 Minutes" går på en kanal som Sissel abonnerer på, lager systemet en notis om dette på hovedsiden.

Sidene kan om ønskelig også benyttes direkte uten å gå via AvantGo. Så om hun ikke har mulighet til å synkronisere PDAen med en PC en dag, har hun mulighet til å koble seg til internett via sin mobiltelefon, og hente ned akkurat de dataene hun abonnerer på, uten å bruke tid på å hente ned informasjon som ikke er av interesse.

5.7.2 Systemvinkel

Konfigurering
Dette gjøres via web interface. Alle brukere må registrere seg (med e-post-adresse) før de kan ta i bruk systemet. En e-post sendes til brukeren etter registrering med informasjon om hvordan systemet fungerer, og hva hvilke steg som må utføres for å få ting på plass.

Noe av det første brukeren må gjøre er å fylle ut et skjema med informasjon om hvilke tjenester som ønskes. Informasjonen lagres i en "tjenestedatabase" med en referanse til brukernavnet til brukeren.

Bruken av e-post-adresse i dette steget er relativt overflødig fra et brukerstandpunkt, da all informasjon som meddeles via e-post potensielt kan vises via web-grensesnittet. E-post benyttes allikevel som et ledd i en autoriseringsprosess, som prøver å sikre at brukerne er seriøse og reelle brukere, samtidig som det er en fornuftig måte å kommunisere med brukerne direkte på (for eksempel ved masseutsendelse av e-post).

Synkronisering
Brukerdatabasen og tjenestedatabasen aksesseres og kombineres med det som finnes av informasjon om samlinger som er tilgjengelige, og et subset returneres i form av en mengde HTML sider som tilsamen inneholder alt brukeren er interessert i å motta. Dette sendes via AvantGo **[W_AVA]** og mottas av brukerens PDA.

Avlesning av resultat
Når som helst kan brukeren åpne sin PDA nettleser og lese av det aktuelle subset av skreddersydd informasjon. Dette kan foregå både via AvantGo som er hovedintensjonen, og direkte på enhver web-leser, som innebærer alt fra stasjonære PCer, bærebare maskiner, PDA, mobiltelefon (helst med innebygd browser, men noen har også hatt suksess gjennom wap-gateways)

5.8 Skalerbarhet

Et problemområde som før eller siden dukker opp i de fleste systemer av den typen vi snakker om her, er skalerbarhet. Det sees her på noen av de tiltakene som er blitt gjort for å øke ytelsen og skalerbarheten til prototypen underveis.

Indekser
Flere tiltak er gjort for å bedre skalerbarheten til systemet. Den aller største flaskehalsen for systemet, er databasetilgangen. Løsningen her ble å legge til indekser i databasen for å øke effektiviteten til databasen. Dette resulterte i en stor forbedring i ytelse uten merkbare bivirkninger.

Optimaliseringer av tabeller
I forbindelse med indekseringen av databasen er det også blitt gjort endringer av feltstørrelser og fjerning av unødvendige felt basert på den tilgjengelige datamassen. Eksempelvis fantes det mange feil som var satt av til 200 eller flere tegn som i gjennomsnitt sjelden brukte mer enn 10-15% av denne kapasiteten.

Begrense database-aksess
For å øke hastigheten ble det også gjort forbedringer når det kom til hvordan og hvor ofte databasen ble bedt om data. Databasen bør kalles så få ganger som mulig fra grensesnittet. Dette kan gjøres ved at grensesnittet tar vare på informasjon (cache) og at denne interne informasjonen benyttes, isteden for å koble seg flere ganger til databasen.

Persistent kobling til databasen
Man kan også benytte seg av persistente koblinger til databasen for å øke hastigheten. Det er da bare nødvendig for et grensesnitt å koble seg opp til databasen en gang for deretter å ha en løpende kontakt med databasen. Prototypen benytter seg ikke av dette enda, men det er en fremtidig løsning om skalerbarhet blir et nytt problem.

Fordele data over flere fysiske databaser
En annen mulighet som også vil løse videre skaleringsproblemer er å benytte seg av flere ulike databaser på separat hardware. Dataene fordeles ut ifra en nøkkel mellom databasene og dataene i databasene kan speiles regelmessig. Dette har også sikkerhetsmessige fordeler, og regnes også som en mulig utvidelse av prototypen.

5.9 Teknologi

Valg av teknologi for prototypen er i stor grad gjort av pragmatiske årsaker. Forfatteren var fra før godt kjent med verktøyene som ble valgt, noe som gjorde at prototypingen ikke i så stor grad er blitt hindret av mangel på praktisk kunnskap. Verktøyene er også lett tilgjengelige, og fremfor alt gratis i bruk.

De valgte verktøyene har løst de fleste problemene som har dukket opp under implementeringen, selv om de har hatt sine begrensninger som til tider har gjort prototyping av enkelte typer funksjonalitet vanskelig.

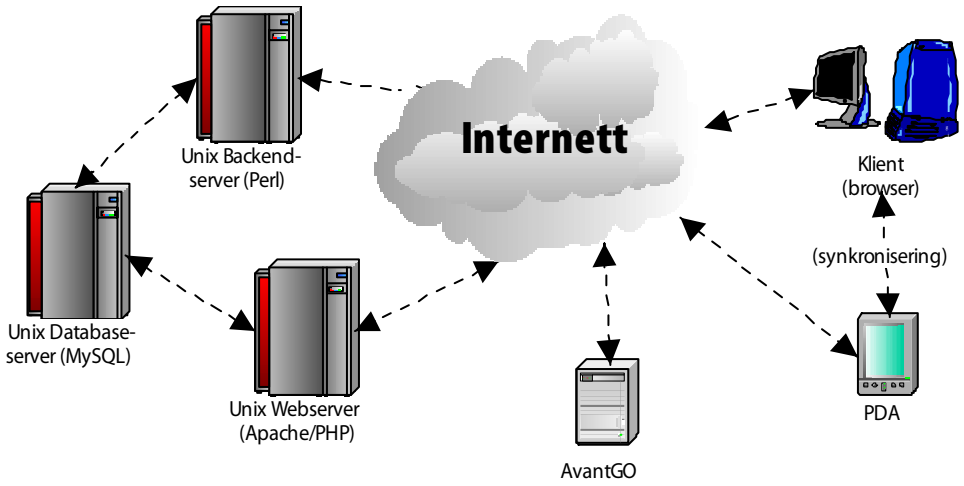

Fig 5-10 Teknologien som benyttes i prototypen

Modellen **[Fig 5-10]** illustrerer forholdet mellom de ulike teknologiplattformene som benyttes i prototypen.

5.9.1 Datalagring

MySQL

I bunnen av systemet finner vi en database, som benyttes til stort sett all lagring av data og informasjon i systemet. MySQL **[W_SQL]** er valgt, da denne er en relasjonsdatabase med et enkelt subset av SQL standarden, optimalisert for å være effektiv og enkel i bruk.

Filsystem

For enkelte typer informasjonsobjekter (spesielt binærobjenter) vil det kunne være mer hensiktsmessig å lagre innenfor filsystemer istedenfor i en database. Forfatterens erfaringer i forbindelse med lagring av store binærobjekter i forbindelse med DigLib-prosjektet ved IDI (ref. Lichtenberg **[LIC00]**) har vist både stabilitetsmessige og praktiske utfordringer tilknyttet dette.

Filsystemer er i prototypen brukt til å lagre større dataobjekter, bilder og kildekode til de enkelte agentene.

5.9.2 Informasjonsdistribusjon

Apache og PHP4

Som webserver ble Apache med server-side språket PHP4 **[W_PHP]** valgt. PHP4 benytes for dynamisk konstruering av HTML (brukergrensesnitt) og gir svært enkel tilgang til databasen.

AvantGO
AvantGO er en kommersiell tjenesteleverandør **[W_AVA]** som tar for seg distribusjonen av informasjonssider til den håndholdte maskinen. Firmaet leverer også selve browseren som benyttes, og kan distribuere informasjon til de største håndholdte operativsystemene på markedet (bl.a. PalmOS og Microsoft PocketPC).

Et praktisk problem som dukket opp underveis var at AvantGO, som med mange fine ord selv hevder de er gratis å benytte seg av, ikke tillater at noen lager en veldig populær tjeneste som distribueres gjennom dem uten å undertegne en kontrakt som blant annet gir dem rett på 25% av alle inntekter tjenesten genererer. Imidlertid var ikke dette noe stort problem, da prototypen ikke hadde noen direkte inntekter. Forfatteren har dessuten flere ganger i løpet av perioden vært i kontakt med AvantGO for å tilpasse tjenesten til tekniske krav de setter til tjenesteleverandører.

Kildekoden til WEB- og PDA grensesnittet av prototypen finnes i *Appendix C og D*.

5.9.3 Backend system

Backend systemet er i stor grad basert på runtime-kompilerte script som kjører ved visse intervaller (for eksempel nedlasting og prosessering av tv-programmet hver dag) eller når spesielle hendelser oppstår (databasen har et felt som er ugyldig eller at en bruker ber om en spesiell handling blir gjort).

Perl
Perl **[W_PRL]** benyttes i stor grad for backend-agenter. Perl har egenskaper som egner seg godt til parsing og bearbeiding av data og populering av databaser.

Unix-verktøy
Flere mindre verktøy blir også benyttet i prototypen. Dette inkluderer blant annet verktøyene **wget, lynx, bash-shell** og **cron.**

Det refereres til *Appendix E* for eksempel på backend-agenter.

6 Erfaringer og analyse

Det har i denne avhandlingen blitt vist noen eksisterende og forslag til nye mulige løsninger på problemområder innenfor digitale bibliotek, spesifikt innenfor problematikken rundt det å integrere flere grensesnitt og flere samlinger i digitale bibliotek. Dette kapittelet vil oppsummere noen hovedpunkter fra avhandlingen, og sette teoriene som er blitt gjennomgått opp mot hverandre.

I avhandlingen presenteres i alt seks[22] ulike vinklinger til digitale bibliotek:

- ***SmartPush*** *(kapittel 3.1)*
- ***MyLibrary*** *(kapittel 3.4)*
- ***UMDL*** *(kapittel 3.3)*
- ***Daffodil*** *(kapittel 3.2)*
- ***Et forslag til arkitektur*** *(kapittel 4)*
- ***En prototype basert på arkitekturen*** *(kapittel 5)*

Vi tar for oss de arkitekturene som er blitt gjennomgått, og evaluerer dem ut ifra tre forskjellige perspektiver:

- ***Forholdet mellom samling og grensesnitt***
- ***Grensesnitt basert på profiler / personifisering***
- ***Bruk av agenter i digitale bibliotek***

På basis av denne evalueringen vil de enkelte arkitekturenes mulige bruksområder identifiseres og kartlegges, for hvert av perspektivene.

22. Den foreslåtte arkitekturen og prototypen regnes i denne sammenhengen som to ulike vinklinger siden de har visse forskjeller (ref. kapittel 7.1.2, *"Arkitekturens forhold til prototypen"*)

6.1 En analyse av arkitekturene

Her belyses de viktigste prinsippene for den foreslåtte arkitekturen (*kapittel 4 og 5*), og settes i forhold til arkitekturer gjennomgått i *kapittel 3*. Ut ifra avhandlingens sentrale problemstilling (ref. *kapittel 1.1.3*) og egenskaper til de ulike arkitekturene kan det pekes på tre sentrale områder som arkitekturene kan sammenliknes rundt:

- ***Forholdet mellom samling og grensesnitt i digitale bibliotek***
- ***Grensesnitt basert på profiler / personifisering***
- ***Bruk av agenter i digitale bibliotek***

6.1.1 Grensesnitt

Mye arbeid er de siste år blitt nedlagt innenfor området digitale bibliotek, og det finnes i dag utallige digitale bibliotek, både i form av implementerte og tilgjengelige løsninger, og forslag til mulige arkitekturer (eksempelvis **[ARM97]**, **[KW95]** og **[FUHR00]**).

Et fenomen forfatteren har observert, er at informasjonsportaler generelt (deriblant mange digitale bibliotek) ofte fremstår som en eller flere samlinger som hver har sitt spesifikke og spesialtilpassede grensesnitt, med et overordnet paraply-grensesnitt som binder samlingene sammen. Dette gjelder ikke i så stor grad forskning, spesielt innenfor digitale bibliotek, hvor det finnes flere eksempel på arkitekturer som setter fokus på et enhetlig grensesnitt. MyLibrary, (ref. *kapittel 3.4*) og Daffodil (ref. *kapittel 3.2*) er gode eksempel på dette.

Dette fenomenet velger vi her å referere til som en *samlingsorientert* tilnærming. Man kan peke på flere mulige grunner til denne utpregede fokuseringen på samlingens innhold, fremfor grensesnittets enhetlighet.

Det kan spekuleres i om ikke et naturlig utgangspunkt for noen som er i ferd med å utvikle et digitalt bibliotek, er først å ta for seg hvilke samlinger som er tilgjengelige. Er man først på dette stadiet, er sannsynligvis et naturlig videre steg for eksempel å se på hvordan samlingene best kan lagres, eller allerede er lagret.

På gunnlag av denne kunnskapen kommer man deretter frem til et passende, gjerne spesialtilpasset grensesnitt. I visse tilfeller blir grensesnittet noe nedprioritert, i og med at dette relativt logisk sett ofte er det siste man utvikler. Man har kanskje sett for seg et grensesnitt gjennom prosessen, kanskje også planlagt i detalj hvordan det skal se ut og hva det skal inneholde, eller utviklet det parallelt med resten av prosjektet. Grensesnittet har dermed hele tiden blitt basert på hvilke muligheter samlingene og tjenestene som er tilgjengelige legger til rette for.

Resultatet man oppnår med en slik tilnærming, er et grensesnitt som reflekterer og er optimalisert til alle mulighetene til de forskjellige elementene i biblioteket, og dermed et kraftig og spesialtilpasset grensesnitt *der og da*. Dette kan være eksakt hva man er ute etter. Problemet dukker gjerne først opp når man har noen *nye* samlinger eller tjenester som skal inn i biblioteket, som man tidligere ikke hadde kunnskap om.

Alternativene da er gjerne enten å **skrive om brukergrensesnittet** helt eller delvis, slik at det også passer det nye systemet, **anpasse det nye samlingen/tjenesten** slik at det har tilnærmet lik funksjonalitet som en av de eksisterende elementene i biblioteket, eller **skrive et helt nytt grensesnitt** og eventuelt lage et *paraply-grensesnitt* over de enkelte grensesnittene for å samle dem.

6.1.2 Forholdet mellom grensesnitt og samling

Ut ifra arbeidet med avhandlingen er det identifisert flere mulige forhold mellom samlinger og brukergrensesnitt i et digitalt biblioteksystem, og hvordan disse delene påvirker hverandre.

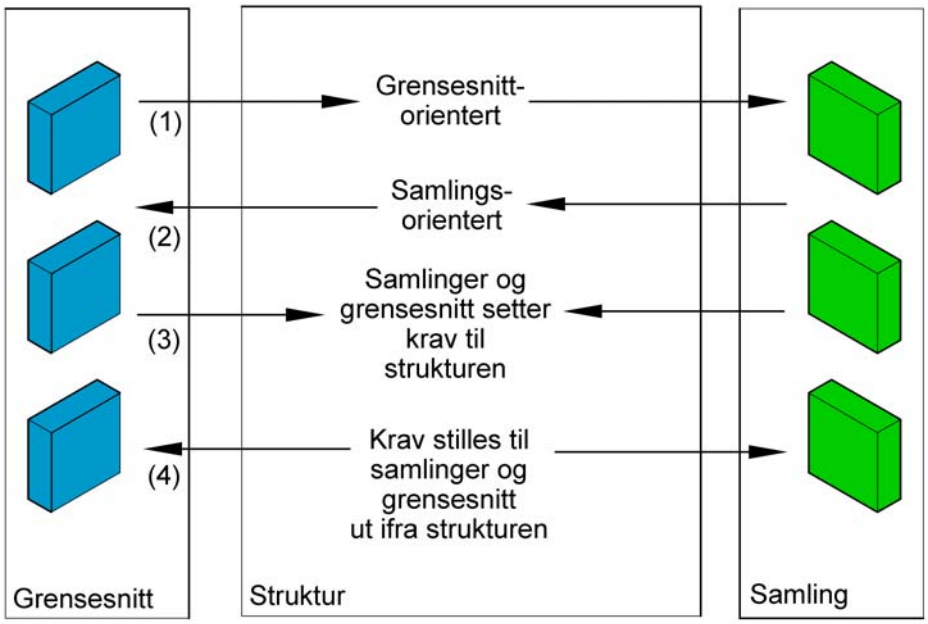

Fig 6-1 Forholdet mellom grensesnitt og samling

Vi tar her for oss fire mulige forhold som kan eksistere mellom et gitt grensesnitt og en gitt samling **[Fig 6-1]**.

Første alternativ (1) er en grensesnitt-orientert tilnærming. Her vil egenskapene til grensesnittet i stor grad legges til grunn både for hvordan resten av systemet og hvordan samlingene blir implementert. Det motsatte er tilfelle med det andre settet med piler i figuren (2). Samlingenes utforming setter presidens over resten av systemet som igjen påvirker hvordan grensesnittet vil se ut.

De to siste tilnærmingene (3 og 4) settes det henholdsvis enten er store krav til hvordan strukturen skal utformes (at dette dikterer et grensesnitt av en gitt type og en samling av et gitt format), eller at både grensesnitt og samling har et format som resten av strukturen i systemet må føye seg til.

Grensesnittorientert (1)
Et grensesnitt-orientert system gjenkjennes ved at både samlingen i bunn og systemet rundt er opprettet akkurat med tanke på det aktuelle grensesnittet. Et enkelt ek-

sempel på dette kan være en WAP side for mobiltelefoner, som kun er spesialtilpasset akkurat denne teknologien og dette grensesnittet.

Den største fordelen med dette er at man har full frihet til utforming av grensesnitt og at man slipper mye dobbeltarbeid om man ønsker et helt spesifikt utseende på grensesnittet, da alt bygger rundt akkurat dette grensesnittet fra begynnelsen av. Ulempene er derimot at man vil kunne risikere at det er mye jobb å utvikle nye grensesnitt og samlinger til systemet, og at eksisterende samlinger som skal inn i dette grensesnittet sansynligvis ikke vil få optimal støtte.

Sett i forhold til arkitekturene i kapittel 3, vil modellen til MyLibrary (*kapittel 3.4.1*) gå inn under denne definisjonen. Det legges stor vekt på at grensesnittet i MyLibrary skal ha et spesifikt utseende, og datalager spesialtilpasses dette grensesnittet.

Samlingsorientert (2)

Samlingsorienterte systemer er sansynligvis de mest vanlige. Her setter samlingen presedens for både grensesnitt og systemet rundt. Et av mange eksempel på dette vil være en samling med hovedfagsavhandlinger, med et tilhørende interface, hvor alle avhandlingene er tilgjengelige.

Den åpenbare fordelen med dette er at man gjerne oppnår et 1:1 forhold mellom samlingen og grensesnittet, hvor all funksjonalitet og alle dataelementer i samlingen også sansynligvis er blitt spesialtilpasset i grensesnittet.

SmartPush (ref. *kapittel 3.1*) vil under denne definisjonen. Det legges i SmartPush arkitekturen vekt dataene som kommer fra en informasjonskilde, og disse påvirker hvilken presentasjon som skal benyttes i grensesnittet.

Strukturorientert (3)

Applikasjonsservere og ferdigutviklede portaler er begge eksempel på strukturorienterte løsninger. Av pragmatiske årsaker vil et digitalt bibliotek kunne basere seg på ferdige løsninger. Dette vil kunne føre til at man er ganske bundet til spesifikke typer grensesnitt og databaser.

Den viktigste fordelen med denne type systemer, er gjerne at de kan settes i drift i løpet av veldig kort tid. De vil også være nødvendig i tilfeller hvor eksempelvis økonomi og kompetanse setter begrensninger for utviklingen av egne løsninger.

Den eneste av arkitekturene i kapittel 3 og 4 som kan relateres til dette aspektet, er MyLibrary i sin implementerte utgave. Tar man i bruk den implementerte versjonen (*kapittel 3.4.2*) i steden for å utvikle sitt eget bibliotek basert på teoriene rundt modellen (*kapittel 3.4.1*) vil man være bundet til de begrensninger på grensesnitt og samlinger som eksisterer i implementasjonen.

Grensesnitt- og samlingsorientert (4)

Det siste aspektet i figuren **[Fig 6-1]** definerer et system hvor både grensesnittet og samlingen setter spesifikke krav til strukturen. Systemer som har et utvalg samlinger som går mot et utvalg grensesnitt vil ofte falle under denne kategorien.

Et eksempel på et slikt system UMDL (*kapittel 3.3*). Den foreslåtte arkitekturen (*kapittel 4*) vil også gå under denne definisjonen.

6.1.3 Grensesnitt basert på profiler

En av nøkkeltankene bak arkitekturen, i tråd med målene presentert i innledningen, er at brukerne selv skal være med i prosessen om å bestemme hva slags grensesnitt som skal benyttes. De skal også, i så stor grad som mulig, selv bestemme hva de ønsker å ha tilgjengelig av informasjon og ressurser i dette grensesnittet.

En måte å løse dette på er å opprette personlige profiler for hver enkelt bruker av systemet. Dette krever at brukeren kan identifiseres unikt og ofte en viss velvilje fra brukere som dermed må oppgi en brukernavn/passord-kombinasjon, eller identifiseres på annet vis, for å få tilgang til ressursene.

Den foreslåtte modellen og MyLibrary deler her mye av den samme tankegangen. MyLibrary krever at brukeren inngår aktivt i informasjonsutvekslingen. Et mål begge arkitekturene er at profiler i begynnelsen vil føre til endel merarbeid for brukeren - men at dette utover tid vil spare brukeren for mye "støy" (uønsket informasjon).

Spesielt UMDL og Daffodil fokuserer på en tradisjonelt streng definisjon av et digitalt bibliotek, hvor det legges veldig stort fokus på statiske samlinger og informasjonsgjenfinning[23]. SmartPush tar i mye større grad for seg en informasjonsflyt - hvor agentene i arkitekturen fortløpende evaluerer innkommende materiale og leverer til brukeren ved behov. MyLibrary har også elementer av informasjonsflyt (blant annet gjennom sin "Current Awareness Manager") men ikke i like stor grad som SmartPush.

6.1.4 Agentbaserte arkitekturer

Alle arkitekturene som omtales er agentbaserte. Dette har ikke vært noe absolutt krav ved valg av arkitekturer, men alle arkitekturene som er vurdert til å være mest gjennomført og i tråd med avhandlingens fokus.

Måten de ulike arkitekturene behandler og bruker agenter på i sine respektive arkitekturer er derimot relativt ulikt, noe vi tar kort for oss her.

Tar vi for oss den foreslåtte arkitekturen, og setter denne opp imot UMDL kan det pekes på endel likheter og sentrale forskjeller. En av de mest åpenbare forskjellene er at agentene i UMDL i mye større grad snakker direkte med hverandre - mens den foreslåtte arkitekturen baserer seg på en form for informasjonskjeder - med en informasjonsmegler i sentrum.

Når det gjelder Daffodil likner agentstrukturen, spesielt for samlinger veldig på den som er foreslått i kapittel 5. "Wrappers"-agenter benyttes for å skille de enkelte samlingene i fra hverandre. MyLibrary har også store likhetstrekk til den foreslåtte arkitekturen når det kommer til implementeringer av agenter i grensesnittet.

6.2 Erfaringer

En stor del av utfordringen innenfor det å utvikle gode arkitekturer for digitale bibliotek, ligger i å finne en fin balansegang mellom det å spørre brukeren om informasjon eller å estimere hva brukeren ønsker. Dette refereres gjerne til som *invasive* og *non-in-*

23. "Information Retrieval"

vasive metoder **[CHA99]**. Brukeren spørres enten om informasjon, ved hjelp av en dialog mellom bruker og system (invasive) eller at en profil for brukeren estimeres (non-invasive), for eksempel på bakgrunn av et statistisk materiale. Systemet bestemmer på grunnlag av dette hva slags grensesnitt brukeren skal ha.

Ingen personer er helt like, og det er det er derfor urimelig å anta at alle brukere vil foretrekke et bestemt grensesnitt eller system. De kan ha også ha forskjellige mål de ønsker å oppnå, og selv om en bruker har et mål en dag, kan dette være forandret neste gang brukeren er i kontakt med systemet **[PER00]**.

Det er, på grunnlag av slike problemfelt, viktig at nyansene mellom brukere, selv innenfor en gitt brukergruppe, og forskjellene mellom større brukergrupper, identifiseres. Dette for dermed å sansynligvis være i stand til å utvikle gode arkitekturer, som utnytter disse forskjellene og små nyansene, med mål om at hver individuell bruker får et optimalt system å jobbe imot.

Noen av disse problemene er veldig avhengig av hva slags digitalt bibliotek man bygger og hvilke krav som stilles til dette, andre tar i stor grad for seg sosiale og psykologiske problemstillinger utenfor rekkevidden til denne avhandlingen.

Det er i avhandlingen blitt sett på en arkitektur for digitale bibliotek, som i stor grad tar for seg de tekniske aspektene av noen problemområder innenfor digitale bibliotek. Det sees her på erfaringer med den foreslåtte arkitekturen og prototypen, i form av noen fordeler og ulemper.

6.2.1 Arkitekturens fordeler

Vi velger her å se på prototypen (kapittel 5), kombinert med prinsippene som kommer frem i arkitekturen (kapittel 4), og se på hva som fungerte best med den foreslåtte arkitekturen.

Arkitekturen bygger på et veldig komplisert problem, som uten tvil har mange mulige løsninger. Det kan pekes på tre viktige grunner til at noen som vil utvikle et digitalt bibliotek bør velge en arkitektur av denne typen:

- ***To abstraksjonslag for samlinger***
  Både samlingenes underliggende teknologi (eksempelvis databaseserver) og hvilke samlinger som eksisterer i det digitale biblioteket abstraheres, slik at det hele blir en stor meta-samling av data.
- ***Abstraherte grensesnitt***
  Legger man til et nytt grensesnitt i det digitale biblioteket, vil dette kun måtte tilrettelegge et generelt sett med funksjonalitet til et spesifikt grensesnitt, dermed ikke bekymre seg om funksjonaliteten til et helt digitalt bibliotek for hvert av grensesnittene.
- ***Dynamiske grensesnitt***
  Brukerens preferanser legges til grunn for både hva slags data som skal vises og hvordan det skal vises.

Subjektivt sett er forfatteren veldig fornøyd med arkitektur, og spesielt prototypen, da den rent pragmatisk sett fungerer på en tilfredstillende måte. Metoden det hele er implementert på er trolig ikke den mest ryddige løsningen man kunne valgt, men den vil likevel kunne regnes som veldig fremtidsrettet, da det byr på lite problem å

legge til nye typer grensesnitt og/eller samlinger. Dette er både basert på de tilbakemeldinger forfatteren har fått av brukere, og på erfaringene som har kommet ut av utviklingen og administreringen av prototypen.

6.2.2 Ulemper med den foreslåtte arkitekturen

Noen av kravene arkitekturen setter kan føre til endel ulemper for både bruker og de som administrerer et digitalt bibliotek basert på den. Det må regnes som ganske usannsynlig å komme frem til en arkitektur som passer for alle DL.

VI tar her for oss en kort gjennomgang av to spesifikke ulemper som har dukket opp under arbeidet med arkitekturen.

Generalisering kan føre til unøyaktighet
Arkitekturen legger opp til at det ikke skal spesialskrives noe spesielt grensesnitt for hver samling. Dette fører med seg ulempen om at spesielle krav en samling har ikke alltid vil kunne bli tilfredstilt, og man mister det 1:1 forhold som gjerne kjennetegner spesialskrevne grensesnitt.

Kompleksitet
Arkitekturen introduserer flere datalager som må administreres (brukerdata, tjenestedata og samlingsmetadata). Disse må alle holdes oppdatert for at systemet skal fungere slik det skal. Sett i forhold til det å ha separate samlinger, med et spesifikt grensesnitt for hver av dem, kan potensielt arkitekturen øke kompleksiteten for den administrative delen av et digitale bibliotek.

7 Evaluering og videre arbeid

Det presenteres her en evaluering av avhandlingen og forslag til videre arbeid. Evalueringen tar utgangspunkt i arbeidet som er blitt gjort med avhandlingen og evaluerer ut ifra de mål og delmål som ble satt i kapittel 1.1, *"Problemstilling og avgrensning"*.

Kapittelet tar deretter for seg både tanker rundt, og konkrete forslag til, mulige utvidelser og forbedringer som kan gjøres innenfor avhandlingens omtalte emner.

7.1 Evaluering av arbeidet med avhandlingen

7.1.1 Prosjektarbeidet

Prosjektarbeidet rundt avhandlingen startet med et ganske annerledes utgangspunkt enn det som er det endelige resultatet. Dette kommer av flere ulike faktorer.

Prosjektet startet med et utgangspunkt om å sette "brukeren i fokus" og se på brukerens forhold til dynamiske grensesnitt - og i den første perioden med prototyping tok avhandlingen dette utgangspunktet. Det viste seg imidlertid fort at denne problematikken hadde så mange aspekter, som ville kreve både empiriske resultater og en meget bred fagkrets for å løses på en tilfredstillende måte. Det ble derfor skiftet fokus over på forholdet mellom data og grensesnitt og hvordan infrastrukturen i et system skulle kunne lette arbeidet med å lage dynamiske brukergrensesnitt.

Avhandlingens fokus over på begrepet arkitekturer, fremfor bare forholdet mellom grensesnitt og data, ble til etterhvert som en av de første utgavene av prototype/implementasjon begynte å ta form. Det kom frem flere aspekter ved prototypen i ettertid, som ville vært nyttig å tatt hensyn til fra starten av. Først og fremst gikk dette på å dele opp (abstrahere) det man ønsket å implementere i flere logiske deler. Dermed kunne man definere hvordan hver enkelt del av systemet skulle fungere i forhold til resten av systemet.

I første omgang ble dette basert på objektorientert tankegang, hvor arv og innkapsling er viktige faktorer. Men assosiasjonene til objektorientert programmering hadde sine ulemper, ved at det satte, i hvertfall i forfatterens øyne, begrensninger i forhold til hvilke typer teknologi som kunne brukes ved implementasjon. Valget falt derfor på agenter, som i stor grad var brukt for å illustrere intelligensen i digitale bibliotek.

7.1.2 Arkitekturens forhold til prototypen

Arbeidet med arkitekturen og prototypen har foregått side om side og med gjensidig inspirasjon. Ser man med kritiske øyne på forholdet prototypen har til arkitekturen, viser det seg fort at prototypen ikke er noen perfekt representasjon.

Det kan pekes på to konkrete grunner til at dette er tilfelle:

- ***Pragmatiske forutsetninger***
  Prototypen er tiden bygget rundt en del pragmatiske forutsetninger. Det er viktig å kunne teste ut nye muligheter og ideer, uten at verktøyene i bakgrunnen setter unødvendige begrensninger. Det er derfor valgt godt dokumenterte og lett tilgjengelige verktøy (ref. kapittel 5.9). Ikke alle konseptene i arkitekturen er derfor like ryddig implementert i prototypen.
- ***Prototypen har fungert som et verktøy for å utvikle arkitekturen***
  Ofte har inspirasjonen til å gjøre endringer i arkitekturen kommet først; deretter har implementasjon av disse endringene inspirert til videre arbeid med arkitekturen. Arkitekturen er derfor ofte et par steg forran prototypen. Ved et tidspunkt ble prototypen lagt til side, og konsentrasjonen fokusert på arkitekturen - den har dermed til dels fått "sitt eget liv".

De viktigste forskjellene mellom arkitekturen og prototypen er:

Agentstrukturen
Agentstrukturen er mye mer konkret og omfattende i arkitekturen enn det som er tilfelle i prototypen. Den samme hovedfunksjonalitet er til stede i både arkitektur og prototype, men prototypen har et mindre disiplinert forhold til agentbegrepet - hvor ideene er beholdt - men implementasjonen ofte vært pragmatisk tilrettelagt.

Ulikhetene i agentstrukturen viser at det kan finnes forskjeller mellom teori og praksis når det kommer til å utvikle digitale bibliotek. Det er imidlertid viktig og prøve å beholder grunntankene ved en arkitekturen til tross for slike forskjeller - noe prototypen i stor grad har gjort.

Informasjonsformidler vs. digitalt bibliotek
I arkitekturen legges det opp til at samlinger både kan være tilnærmet statiske, i det at de er et veldefinert sett med data, og at samlinger kan være lagret eksternt, men også at informasjonsmengder som varierer mye i størrelse og form (for eksempel en samling som alltid inneholder dagens nyheter).

Prototypen tar kun for seg en liten del av dette. Man kan ha rett i en definisjon av prototypen som en ren informasjonsformidler, hvor tradisjonelle "samlinger" ikke eksisterer i så stor grad. Det er likevel grunnlag for å definere prototypen som et digitalt bibliotek - sett i forhold til definisjonen som presenteres i kapittel 2.

7.1.3 Evaluering av måloppnåelse for arkitekturen

Innledningen (*kapittel 1.1.4*) definerer tre hovedmål for arkitekturen. Til felles for disse målene er at de krever en gjennomgående tankegang om at så mye som mulig i arkitekturen skal kunne bestemmes av brukere og de som vedlikeholder et digitalt bibliotek basert på arkitekturen, og ikke hardkodes inn en implementasjon.

> ***Mål 1: Grensesnittene***
> *Det skal være lett å legge til og fjerne grensesnitt til en implementasjon, og arkitekturen skal understøtte at de samme dataene forholdsvis enkelt skal kunne fremvises på forskjellige grensesnitt og teknologiplattformer.*

Mål 1 tas i stor grad hånd om av arkitekturen. Dette illustreres i modellen som beskriver arkitekturen ved hjelp av grensesnitt-agenter med domenekunskap som abstraherer forholdet mellom grensesnittet og resten av systemet.

> ***Mål 2: Samlingene***
> *Det skal være lett å legge til og fjerne samlinger, i den grad dette er mulig uten å endre store mengder kode i en ferdig implementasjon.*

Agenter (med tilgang til samlingsmetadata) benyttes for å abstrahere forholdet mellom de enkelte samlingene og den totale mengden samlinger. Dette støttes ved hjelp av et sentralt datalager med samlingsmetadata som beskriver i detalj hvilke samlinger som finnes til enhver tid. De enkeltstående samlingene er i tillegg abstrahert bort fra den fysiske datalagringen ved hjelp av agenter. Disse stegene oppfyller dermed målet om å gjøre det lettere å legge til og fjerne samlinger i et operativt digitalt bibliotek.

> ***Mål 3: Brukerne***
> *Det skal være mulig for brukeren å fritt velge et passende grensesnitt, og å være med på å bestemme innholdet og utseendet til dette grensesnittet.*

Mål 3 er også tilfredsstilt, både indirekte ved at mål 1 er løst, og ved at det finnes agenter i arkitekturen som tar for seg brukerprofiler.

7.1.4 Arbeidet med avhandlingen

Avhandlingen har bygget seg opp gradvis i løpet av tiden forfatteren har vært hovedfagsstudent, med store variasjoner underveis før avhandlingens endelige ramme ble satt. Arbeidet har det meste av perioden foregått på deltid, i kombinasjon med jobb, noe som fra et økonomisk og karrieremessig standpunkt har vært både positivt og en ren nødvendighet. Men det er er en arbeidsform som ikke alltid gjør at arbeidet med avhandlingen har fått den prioritet den har trengt. Inspirasjon kan komme når som helst. Ofte har det mest fruktbare arbeidet foregått på lange dager og gjerne imellom 02:00 og 06:00 om natten - det lar seg dårlig kombinere med hektiske arbeidsdager.

Det er veldig vanskelig å evaluere kvaliteten på informasjon, enten det gjelder forskningsarbeid eller generell informasjon på internett. I forbindelse med arbeidet med avhandlingen har forfatteren erfart hvor store sprik det kan finnes mellom kilder til informasjon, og hvor viktig det er å bruke referanser som, om ikke annet, dokumenterer andre kilder til informasjon innenfor det samme emnet slik at man i mye større grad er i stand til å evaluere informasjon.

7.2 Utvidelser og videre arbeid

Arbeidet med dette prosjektet har avdekket flere områder som kan være aktuelle for videre arbeid og forskning. Disse emnene presenteres her i form av noen mulige utvidelser av arbeidet, og tre konkrete prosjektforslag. Prosjektforslagene tar for seg både forslag til angrepsvinkler og aktuelle litterære utgangspunkt.

7.2.1 Utvidelser av emner avdekket i avhandlingen

- Forholdet mellom brukergrensesnittet og brukeren, spesielt innenfor personifiserte og adaptive grensesnitt, er et område som bør vies videre forskning. Emner som utmerker seg er blant annet hvordan interaksjonen med forskjellige brukergrupper (med variasjon på kunnskapsnivå, alder og liknende) fungerer, og kan gjøres bedre og mer effektiv.
- Hvordan et sett med ulike grensesnitt kan fremvise den samme informasjonen er i denne avhandlingen i hovedsak fremstilt ved å se på noen få spesifikke grensesnitt. Et emne for videre forskning er å se på generelle egenskaper ved ulike grensesnitt og ut ifra dette komme frem til generelle høynivå-metoder for informasjonsformidling på tvers av grensesnitt.

7.2.2 Digitalt Biblioteksprosjekt (DigLib)

Det digitale biblioteksprosjektet (DigLib) ved Institutt for Datateknikk og Informasjonsvitenskap (IDI) **[W_IDI]** er et pågående prosjekt som drives og utvikles av ansatte, studenter, hovedfagsstudenter og doktorstudenter ved informasjonsforvaltning (IF). DigLib har vært i utvikling siden 1999 med mål om å være en plattform og utviklingsmiljø for studenter og ansatte ved IDI.

Målet for prosjektet er å utvikle en komponentbasert arkitektur for interoperatible digitale bibliotek. Prosjektet skal tilby ansatte og studenter ved IDI muligheten til å forske og utvikle komponenter og arkitekturer etter eget behov og ønske. **[LIC00]**

Det er her flere muligheter til videre forskning. En mulig retning er å utvide arkitekturen til DigLib med elementer fra de agentbaserte arkitekturene gjennomgått her, for blant annet å understøtte gjenfinning på tvers av samlingene. En annen mulig retning er å se på muligheter for å utvide grensesnitt-mulighetene i DigLib, og de samlingene som DigLib har tilgjengelig, med personifiserte profiler og et eller flere samlede grensesnitt.

For en grundigere gjennomgang av DigLib, refereres det forøvrig til Aalbergs artikkel "*IDI DIGLIB: Komponentbasert arkitektur for forskning og utvikling av digitale bibliotek*" **[AAL00]** og Lichtenberg's hovedfagsavhandling "*Lenker mellom informasjonsobjekter i det digitale bibliotek*" **[LIC00]**

7.2.3 E-handel

Kvanli **[KVA01]** tar i sin hovedoppgave opp teknologisk infrastruktur for e-handel med spesiell vekt på katalogtjenester.

Blant kravene Kvanli fremhever, når det kommer til grensesnitt, er at kunden, spesielt i Business To Consumer (B2C) øyemed, bør ha mulighet til å kommunisere med nettbutikken via forskjellige kanaler (TV, faks, telefon, mobiltelefon PC eller PDA). Videre kreves det spesielle egenskaper for å kommunisere mellom to bedrifter (Business To Businness, B2B) som standardiserte protokoller og kompatible metoder for utveksling av data. Det legges også opp til at det kan foregå en samhandling mellom to bedrifter (B2B) som en del av sluttsalg (B2C) for en eller begge parter.

Angrepsvinkler her vil kunne være være å se på hvordan agentbaserte arkitekturer og personifiserte grensesnitt kan benyttes for å understøtte de krav som finnes til e-handel-systemer.

7.2.4 Born-digital/deep searches

Born-digital problematikk, blant annet beskrevet i detalj av Rauber/Aschenbrenner **[RAU01]**, er et aktuelt tema, og ventes å bli enda mer aktuelt etter hvert som flere og flere publiserer gjennom digitale mediakanaler og informasjon i større grad kun lagres elektronisk. Det kan derfor være interessant å ta for seg langtidslagring av digitale objekter, og effektene dette trenden vil ha på dagens og fremtidens digitale bibliotek.

Rauber/Aschenbrenner tar også for seg tjenester som forsøker å kontinuerlig indeksere det som finnes av data på internett. Et gjennomsnittsdokument på internett har i dag et urovekkende lavt livsløp på kun 44 dager **[RAU01]**. En mulig anne angrepsvinkel kan derfor være å se på hvordan *deep searches*[24] i fra søkeroboter på best mulig måte kan indeksere digitale bibliotek som har grensesnitt som i utgangspunktet er veldig lukkede.

24. Søk som krever at søkeroboten ikke bare følger lenker, men aktivt søker og benytter tjenester for å prøve å finne alle data

8 Bibliografi

1.1 Mal for biblografiske referanser

Referansene bygger på formatet som de publisert blant annet i D-Lib Magazine og format benyttet i tidligere avhandlinger ved IDI. Nøkkelen er i hovedsak bygget opp av de tre første bokstavene på hovedforfatterens etternavn etterfulgt av årstall på to siffer, men noen avvik kan forekomme. Referansene er sortert alfabetisk og etter årstall - dette betyr at [EKS99] går naturlig forran [EKS00].

[Nøkkel] *Forfatter(e)*
Emne eller tittel på artikkel
Hvor artikkelen forekommer eller er utgitt og årstall
url:// (adresse til websted hvor artikkelen kan leses)

1.2 Artikkelreferanser benyttet i avhandlingen

[AAH00] *Trond Aalberg, Knut Hegna*
Arkitektur for digitale bibliotek
BIBSYS, 2000
http://www.bibsys.no/BDB/arkitektur/ArkDigBib.pdf

[AAL00] *Trond Aalberg*
IDI DIGLIB : Komponentbasert arkitektur for forskning og utvikling av digitale bibliotek
NTNU, IF-IDI 2000
Merknad: utgitt internt ved IDI i forbindelse med prosjektarbeid rundt DigLib

[ARM97] *William Y. Arms*
An Architecture for Information in Digital Libraries
D-Lib, 1997
http://www.dlib.org/dlib/february97/cnri/02arms1.html

[BIR95] *William P. Birmingham*
An Agent-Based Architecture for Digital Libraries
D-Lib, 1995
http://www.dlib.org/dlib/July95/07birmingham.html

[BUL98] *Dick C. A. Bulterman*
User-Centered Abstractions for Adaptive Hypermedia Presentations
CWI, 1998
http://www.cwi.nl/~media/publications/uca-mm98.pdf

[CHA99] *Philip K. Chan*
A non-invasive learning approach to building web user profiles
KDD-99 Workshop, "Web Usage Analysis and User Profiling", Australia, 1999
http://citeseer.nj.nec.com/chan99noninvasive.html

[CAT97] *Warwick Cathro*
Metadata: An Overview
"Matching Discovery and Recovery" Seminar, Australia, 1997
http://www.ifla.org/documents/libraries/cataloging/metadata/cathro.pdf

[CLDF98] *Carl Logoze, David Fielding*
Defining Collections in Distributed Digital Libraries
D-Lib Magazine, Nov. 1998
http://www.dlib.org/dlib/november98/lagoze/11lagoze.html

[DUR97] *E. H. Durfee, D. L. Kiskis, W. P. Birmingham*
UMDL Agent Architecture version of 4/15/97
University of Michigan, 1997
http://citeseer.nj.nec.com/3152.html

[FUHR00] *N. Fuhr, N. Gövert, C. Clas*
An Agent-Based Architecture for Supporting High-Level Search Activities in Federated Digital Libraries
University of Dortmund, 2000
http://ls6-www.informatik.uni-dortmund.de/bib/fulltext/ir/Fuhr_etal:00.pdf

[FUHR02] *N. Fuhr, C. Klas, A. Schaefer, P. Mutschke*
Daffodil: An Integrated Desktop for Supporting High-Level Search Activities in Federated Digital Libraries
DL02, 2002
http://ls6-www.cs.uni-dortmund.de/bib/fulltext/ir/Fuhr_etal:02.pdf

[GCES01] *N. Fuhr, P. Hansen, M. Mabe, A. Micsik, I. Sølvberg*
Digital Libraries: A Generic Classification and Evaluation Scheme
ECDL, 2001
http://ls6-www.informatik.uni-dortmund.de/bib/fulltext/ir/Fuhr_etal:01.pdf

[HOL98] *Sigbjørn Holmslet*
Arkitektur for digitale bibliotek - et digitalt bibliotek basert på Dienst
NTNU, IF-IDI, 1998
http://www.idi.ntnu.no/grupper/if/publikasjoner/Hovedoppgave.Holmslet.pdf

[KUR99] *Teppo Kurki, Sami Jokela, Reijo Sulonen, Marko Turpeinen*
Agents in Delivering Personalized Content Based On Semantic Metadata
Finland, 1999
http://citeseer.nj.nec.com/kurki99agents.html

[KVA01] *Jarl Kvanli*
Infrastruktur for elektronisk handel
NTNU, IF-IDI, 2001
http://http://www.idi.ntnu.no/grupper/if/publikasjoner/Hovedoppgave.Kvanli.pdf

[KW95] *Robert Kahn, Robert Wilensky*
A Framework for Distributed Digital Object Services
University of California at Berkley, 1995
http://www.cnri.reston.va.us/home/cstr/arch/k-w.html

[LIC00] *Einar W. M. Lichtenberg*
Lenker mellom informasjonsobjekter i det digitale bibliotek
NTNU, IF-IDI, 2000
http://www.idi.ntnu.no/grupper/if/publikasjoner/Hovedoppgave.Lichtenberg.pdf

[MOD99] **MyLibrary: A Model for Implementing a User-centered, Customizable Interface to a Library's Collection of Information Resources**
Merknad: Ikke lenger tilgjengelig online pr. september 2002. ***[MYL99]*** *er ikke identisk men inneholder deler av den orginale artikkelen.*

[MYL99] *Eric Lease Morgan*
MyLibrary@NCState: The Implementation of a User-centered, Customizable Interface to a Library's Collection of Information Resources
ACM SIGIR, 1999
http://www.ted.cmis.csiro.au/sigir99/morgan/

[OPP00] *Anita Iren Oppedal*
Alt er metadata: Bruk av metadata i et integrert brukersystem
NTNU, IF-IDI, 2000
http://www.idi.ntnu.no/grupper/if/publikasjoner/Hovedoppgave.Oppedal.pdf

[PER00] *Mike Perkowitz, Oren Etzioni*
Towards adaptive Web sites: Conceptual framework and case study
University of Washington, Seattle, 2000
http://citeseer.nj.nec.com/326006.html

[PRE94] *Jenny Preece, Yvonne Rogers m.fl.*
Human-Computer Interaction
ISBN 0-201-62769-8, 1994

[RAU01] *Anders Rauber, Andreas Aschenbrenner*
Part of Our Culture is Born Digital - Our Efforts to Preserve it for Future Generations
Vienna University of Technology, Østerrike, 2001
http://citeseer.nj.nec.com/496577.html

[SCH99] *Candy Schwartz*
LIS 462 - Digital Libraries Definitions
Simmons College, 1999
http://web.simmons.edu/~schwartz/462-defs.html

[SDL98] *Barry M. Leiner rep. for "D-Lib Working Group on Digital Metrics"*
The Scope of the Digital Library
D-Lib, 1998
http://www.dlib.org/metrics/public/papers/dig-lib-scope.html

[WAT98] *Donald J. Waters,*
What Are Digital Libraries?
CLIR, 1998
http://www.clir.org/pubs/issues/issues04.html#dlf

[WEI99] *Peter. C. Weinstein, William. P. Birmingham, Edmund H. Dunfee*
Agent-Based Digital Libraries: Decentralization and Coordination
University of Michigan, 1999
http://citeseer.nj.nec.com/weinstein99agentbased.html

[WOO95] *M. Wooldridge, N. Jennings*
Intelligent Agents: Theory and Practice
Queen Mary & Westfield College, England, 1995
http://citeseer.nj.nec.com/97055.html

1.3 Internettreferanser benyttet i avhandlingen

[W_ASP] **Active Server Pages .NET**
http://www.asp.net/

[W_AVA] **AvantGO**
http://www.avantgo.com/

[W_COR] **Corba Basics / FAQ**
http://www.omg.org/gettingstarted/corbafaq.htm

[W_DLF] **Digital Library Federation**
http://www.diglib.org/

[W_IDI] **Institutt for Datateknikk og Informasjonsvitenskap, NTNU**
http://www.idi.ntnu.no/

[W_LOC] **Library Of Congress**
http://www.loc.gov/

[W_MDL] **UMDL Technologies, Architecture: Agents And Ontologies**
http://www.si.umich.edu/UMDL/architecture.html

[W_MYL] **MyLibrary @ NCState**
http://my.lib.ncsu.edu/

[W_NCS] **NC State University**
http://www.ncsu.edu/

[W_NET] **Microsoft .NET**
http://www.microsoft.com/net/

[W_NTB] **NTB (Norges Telegrambyrå) og NTB Pluss**
http://www.ntb.no/

[W_PDF] **John Warnock on PDF: Its Past, Present and Future**
http://www.planetpdf.com/mainpage.asp?webpageid=1841

[W_PHP] **PHP**
http://www.php.net/

[W_PRL] **Perl**
http://www.perl.com/

[W_SQL] **MySQL**
http://www.mysql.com/

[W_XML] **XML, Extensible Markup Language Architecture, W3C**
http://www.w3.org/XML/

1.4 Ressurser benyttet under arbeidet med avhandlingen

[A_ADI] **AvantGO developer information**
http://developer.avantgo.com/developers/

[A_GOG] **Google**
http://www.google.com

[A_HA93] *Knut Halvorsen*
Å forske på samfunnet
ISBN 82-7037-794-5

[A_NEC] **NEC Research Institute ResearchIndex**
http://citeseer.nj.nec.com/

A PåVei Underholdning

A.1 Historie

Utviklingen som startet i det små som et forskningsprosjekt i forbindelse med forfatterens hovedoppgave, ble såpass godt motatt av de initielle testbrukerne, at det virket hensiktsmessig å gjøre systemet allment tilgjengelig. Systemet ble derfor, i Mars 2000, satt i drift av forfatteren. Etter tilbakemeldinger fra brukere og erfaringer som var blitt gjort, ble systemet videre redesignet og oppgradert Juni 2000.

Januar 2001 ble forfatteren kontaktet av Schibsted Telecom (Inpoc) om oppkjøp av tjenesten. Forfatteren ble som en følge av dette også ansatt på deltid i firmaet, ansvarlig for videre utvikling og drift av systemet. Støtte, informasjon og data kom fra tunge Schibsted-aktører som VG og Aftenposten, samt data fra bl.a. Filmweb, NTB (Norsk Telegram Byrå), OSE (Oslo Stock Exchange) og andre mindre leverandører.

I mangel på gode tekniske og praktiske muligheter for inntekt innenfor rammen av dette systemet, samt en fokusering av Schibsted sin virksomhet innenfor mobile tjenester, og generell pessimisme og nedgang i markedet, ble det i April 2002 desverre besluttet at det for Schibsted sin del ikke lenger var økonomsik hensiktsmessig å fortsette kommersiell drift av *PåVei Underholdning*.

A.2 Status

PåVei Underholdning vil i fremtiden, så lenge det er økonomisk og tidsmessig forsvarlig, bli drevet av forfatteren på frivillig basis. Dette i hovedsak grunnet den relativt store brukermassen, og forfatterens personlige interesse av å holde systemet ved like.

Fordelen med systemet nå er at det er enormt stabilt. Det kjørte i over et år på samme server, uten stabilitetsproblemer, og uten at det måtte gjennomføres reboot av serveren. Agentene som tar for seg alt backend-arbeid er også såpass automatiske, at den ikke trenger noe gjort manuelt såfremt det ikke skjer noe feil hos innholdsleverandører, eller det er nye tjenester som skal legges til.

Tjenesten finnes på **http://www.pavei.com/.** Man behøver ikke ha en PDA for å prøve ut tjenesten, da den inkluderer en *PDA-emulator* som også er blitt brukt for å illustrere bruken av PDA i denne avhandlingen.

B Datalager

Her følger en komplett oversikt over de databasetabellene som er brukt i implementasjonen.

Ubrukte tabeller er fjernet, og sensitive data er maskert bort. Det kan forekomme små uregelmessigheter i forhold til avhandlingens gjengivelse av disse tabellene, da avhandlingen har rettet opp, og justert navnene på, noen felter som ellers ville kunne bli missforstått eller ikke var beskrivende nok.

	Felt	Type	Attributter	Null	Standard	Ekstra
☐	id	int(10)	UNSIGNED	Nei		auto_increment
☐	brukernavn	varchar(31)		Nei		
☐	passord	varchar(31)		Nei		
☐	email	varchar(63)		Ja	*NULL*	

id	brukernavn	passord	email
5169	te or		tno .no
5170	b oltu		bjo o
5171	9 69986		rub surf.no
5172	b kum		ber 2i.net
5173	h dal		jha o.no
5174	b ff		boe .com
5175	p ick_vogel48		pat ntnu.no
5176	s es		sol .no

Fig B-1 Brukerdatabase <sql:brukere>

	Felt	Type	Attributter	Null	Standard	Ekstra
☐	id	int(10)	UNSIGNED	Nei		auto_increment
☐	brukernavn	varchar(255)		Ja	*NULL*	
☐	tekst	varchar(255)		Ja	*NULL*	
☐	dato	date		Ja	*NULL*	

id	brukernavn	tekst	dato
1	olebakk	Velkommen tilbake!	2002-08-17
2	olepuk	flott at dere kommer på nett igjen!	2002-08-18

Fig B-2 Chatboard <sql:chatboard>

	Felt	Type	Attributter	Null	Standard	Ekstra
☐	cid	int(10)	UNSIGNED	Nei		auto_increment
☐	kort	varchar(20)		Nei		
☐	lang	varchar(64)		Nei		

cid	kort	lang
1	3plus	3+ Danmark
2	3sat	3sat
3	animal	Animal Planet
4	bbcp	BBC Prime
5	bbcw	BBC World
6	tcn	Cartoon
7	cgul	Canal+ Gul Norge
8	cinema	Cinema
9	cnbc	CNBC
10	cnn	CNN
11	cplus	Canal+ Norge
12	discovery	Discovery

Fig B-3 TV-kanal mapping[a] <sql:tv_conv>

a. Denne tabellen tar for seg mapping/konvertering av kort-navn brukt i backend-systemer til full-tekst navn som er lettere identifiserbare av brukerne

	Felt	Type	Attributter	Null	Standard	Ekstra
☐	tid	int(10)	UNSIGNED	Nei		auto_increment
☐	id	int(10)	UNSIGNED	Nei	0	
☐	type	varchar(31)		Nei		
☐	tjeneste	varchar(127)		Nei		
☐	parameter	varchar(127)		Ja	*NULL*	

tid	id	type	tjeneste	parameter
61	1422	tv	tv_dager	3
62	1422	vaer	v_langtid	yes
63	1422	hovedside	tv	1
64	1422	hovedside	vaer	1
65	1422	hovedside	chatboard	yes
66	1422	hovedside	nyheter	yes
91	123	kino	sandvikakino	yes
68	123	ticker	tel	0@0
90	123	kino	askerkino	yes
89	123	okonomi	shareidx	yes
88	123	okonomi	aksjer	yes
87	123	tv	tv_dager	2
86	123	tv	kanaler	ntv2
85	123	tv	kanaler	nrk1
84	123	tv	kanaler	nrk2

Fig B-4 Tjenestedatabase <sql:tjenester>

	Felt	Type	Attributter	Null	Standard	Ekstra
☐	id	int(10)	UNSIGNED	Nei		auto_increment
☐	brukernavn	varchar(31)		Nei		
☐	dato	date		Nei	0000-00-00	
☐	tid	time		Nei	00:00:00	
☐	ip	varchar(31)		Nei		
☐	browser	varchar(40)		Ja	*NULL*	
☐	ref	varchar(150)		Ja	*NULL*	
☐	opt1	varchar(50)		Ja	*NULL*	
☐	opt2	varchar(20)		Ja	*NULL*	

id	brukernavn	dato	tid	ip	browser	ref	opt1	opt2
151	dvasaru	2002-08-17	18:59:20	22.80-202-24.nextgentel.com	Mozilla/4.0 (compatible; MSIE 6.0; Windo	http://www.pavei.com/main.php3	main.php	*NULL*
152		2002-08-17	18:59:33	callisto.chello.no	Mozilla/4.0 (compatible; MSIE 6.0; Windo		main.php	*NULL*
153	olebakk	2002-08-17	18:59:35	22.80-202-24.nextgentel.com	Mozilla/4.0 (compatible; MSIE 6.0; Windo	http://www.pavei.com/pavei_ipaq.html?id=olebakk	palm.php	*NULL*
154	dvasaru	2002-08-17	18:59:46	22.80-202-24.nextgentel.com	Mozilla/4.0 (compatible; MSIE 6.0; Windo	http://palm.pavei.com/palm.php?id=olebakk	palm.php	*NULL*
155	eiander	2002-08-17	19:00:47	callisto.chello.no	Mozilla/4.0 (compatible; MSIE 6.0; Windo	http://www.pavei.com	main.php	*NULL*
156	eiander	2002-08-17	19:01:20	callisto.chello.no	Mozilla/4.0 (compatible; MSIE 6.0; Windo	http://www.pavei.com/main.php3	main.php	*NULL*
157	olebakk	2002-08-17	19:01:45	22.80-202-24.nextgentel.com	Mozilla/4.0 (compatible; MSIE 6.0; Windo		palm.php	*NULL*
158	dvasaru	2002-08-	19:01:53	22.80-202-	Mozilla/4.0 (compatible; MSIE		palm.php	*NULL*

Fig B-5 Loginfo <sql:logs>

	Felt	Type	Attributter	Null	Standard	Ekstra
☐	kid	int(10)	UNSIGNED	Nei		auto_increment
☐	sted	varchar(31)		Nei		
☐	sal	varchar(31)		Ja	*NULL*	
☐	film	varchar(127)		Nei		
☐	tidspunkt	varchar(31)		Nei		
☐	aldersgrense	varchar(31)		Ja	*NULL*	
☐	lengde	varchar(31)		Ja	*NULL*	
☐	dato	varchar(31)		Nei		
☐	spraak	varchar(31)		Ja	*NULL*	

kid	sted	sal	film	tidspunkt	aldersgrense	lengde	dato	spraak
1	Kino Z	Sal 3 Stavanger	Yamakasi	15:30			18-08-2002	
2	Oslo Kinematografer	Saga 6	Yamakasi	21:10			18-08-2002	
3	Oslo Kinematografer	Saga 6	Yamakasi	19:00			18-08-2002	
4	Bergen Kino	KP 8	Windtalkers	21:15			18-08-2002	
5	Bergen Kino	KP 8	Windtalkers	18:30			18-08-2002	
6	Bergen Kino	KP 8	Windtalkers	15:45			18-08-2002	
7	Bergen Kino	KP 8	Windtalkers	13:00			18-08-2002	
8	Drammen Kino	Sal 4 - Perfekt	Windtalkers	20:30			18-08-2002	
9	Kino Z	Sal 3 Sandnes	Windtalkers	21:00			18-08-2002	
10	Kino Z	Sal 7 Stavanger	Windtalkers	20:30			18-08-2002	
11	Kristiansand Kino	1	Windtalkers	21:30			18-08-2002	
12	Kristiansand Kino	1	Windtalkers	18:45			18-08-2002	
13	Oslo Kinematografer	Eldorado 1	Windtalkers	20:50			18-08-2002	

Fig B-6 Kinodatabase <sql:kino>

	Felt	Type	Attributter	Null	Standard	Ekstra
☐	hid	int(10)	UNSIGNED	Nei		auto_increment
☐	ticker	varchar(12)		Nei		
☐	navn	varchar(75)		Nei		
☐	tid	varchar(12)		Ja	*NULL*	
☐	siste_omsatt	varchar(10)		Ja	*NULL*	
☐	forandring	varchar(10)		Ja	*NULL*	
☐	forandring_pros	varchar(10)		Ja	*NULL*	
☐	kjop	varchar(10)		Ja	*NULL*	
☐	salg	varchar(10)		Ja	*NULL*	
☐	vol	varchar(10)		Ja	*NULL*	
☐	high	varchar(10)		Ja	*NULL*	
☐	low	varchar(10)		Ja	*NULL*	

hid	ticker	navn	tid	siste_omsatt	forandring	forandring_pros	kjop	salg	vol	high	low
1	APR	A-pressen		0.00	n/a	n/a	100.50	110.00	0	0.00	0.00
2	ASC	ABG Sundal Collier		0.00	n/a	n/a	2.30	2.50	0	0.00	0.00
3	AFG	AF Gruppen		32.00	n/a	n/a	30.50	34.00	600	32.00	32.00
4	AZNW04F500HA	AZN Warrant HA ISIN		0.00	n/a	n/a	3.48	3.53	0	0.00	0.00
5	ACTA	Acta Holding		1.15	n/a	n/a	1.10	1.20	12000	1.15	1.10
6	ACS	Actinor Shipping		0.00	n/a	n/a	0.00	0.00	0	0.00	0.00
7	AAV	Adresseavisen		0.00	n/a	n/a	230.00	260.00	0	0.00	0.00
8	AMA	Aker Maritime		0.00	n/a	n/a	0.00	0.00	0	0.00	0.00
9	AIK	Aktiv Kapital		47.00	n/a	n/a	45.50	47.00	1000	47.00	47.00
10	ALX	Altinex		0.81	n/a	n/a	0.80	0.87	88000	0.85	0.80
11	AHM	Amersham		66.00	n/a	n/a	66.00	67.00	254820	68.00	66.00
12	APP	Apptix		1.50	n/a	n/a	1.55	1.70	15000	1.60	1.50

Fig B-7 Aksjedatabase <sql:aksje>

	Felt	Type	Attributter	Null	Standard	Ekstra
☐	fid	int(10)	UNSIGNED	Nei		auto_increment
☐	navn	varchar(75)		Nei		
☐	dato	varchar(12)		Nei		
☐	netto	double(16,4)		Ja	*NULL*	
☐	boy	double(16,4)		Ja	*NULL*	
☐	tolv	double(16,4)		Ja	*NULL*	
☐	treaar	double(16,4)		Ja	*NULL*	
☐	utvidelse2	varchar(10)		Ja	*NULL*	
☐	utvidelse1	varchar(10)		Ja	*NULL*	

fid	navn	dato	netto	boy	tolv	treaar	utvidelse2	utvidelse1
1	ABIF Aktiv	16.08	213.1700	-17.3500	-26.1200	-47.0200	-19.84	-5.40
2	ABIF Norge	16.08	117.7000	-15.8100	-24.0200	-38.5800	-12.49	-5.06
3	ABIF Norge +	16.08	74.1400	-15.7200	-23.8600	-38.3600	-11.07	-5.12
4	ABIF Norge ++	16.08	72.9200	-15.3700	-22.7600	-37.0400	-	-5.09
5	ABIF OBX	16.08	1969.0500	-19.5100	-27.0700	-37.2600	-8.89	-5.68
6	Avanse Norge	16.08	1317.4200	-23.0200	-30.6000	-44.8200	-24.39	-7.65
7	Avanse Norge Aktiv	16.08	13662.2300	-24.2500	-30.5800	-50.4400	-33.66	-6.57
8	Avanse Norge Aktiv II	16.08	1294.7000	-24.0000	-28.5600	-47.5700	-25.61	-6.32
9	Avanse OBX Indeks	16.08	1795.3000	-18.9700	-26.8500	-36.7700	-11.07	-5.71
10	Banco Humanfond	16.08	68.0600	-19.4400	-19.4600	-38.9100	-	-6.13

Fig B-8 Aksjefond-database <sql:aksjefond>

	Felt	Type	Attributter	Null	Standard	Ekstra
☐	id	int(10)	UNSIGNED	Nei		auto_increment
☐	trade_date	date		Ja	*NULL*	
☐	sec_id	int(10)		Ja	*NULL*	
☐	symbol	varchar(16)		Nei		
☐	isin	varchar(12)		Ja	*NULL*	
☐	sec_name	varchar(34)		Ja	*NULL*	
☐	bid	double		Ja	*NULL*	
☐	offer	double		Ja	*NULL*	
☐	open	double		Ja	*NULL*	
☐	high	double		Ja	*NULL*	
☐	low	double		Ja	*NULL*	
☐	last	double		Ja	*NULL*	
☐	off_share_turnover	double		Ja	*NULL*	
☐	off_turnover	double		Ja	*NULL*	
☐	nonoff_share_turnover	double		Ja	*NULL*	
☐	nonoff_turnover	double		Ja	*NULL*	
☐	list_name_no	varchar(60)		Nei		
☐	sector_no	varchar(60)		Nei		
☐	list_name_en	varchar(16)		Ja	*NULL*	
☐	sector_en	varchar(16)		Ja	*NULL*	

id	trade_date	sec_id	symbol	isin	sec_name	bid	offer	open	high	low	last	off_share_turnover	off_turnover	nonoff_sha
1	2002-08-16	6001	AAV	NO0003502809	Adresseavisen	230	260	0	0	0	0	0	0	
2	2002-08-16	6006	AFK	NO0003572802	Arendals Fossekompani	360	370	0	0	0	0	0	0	
3	2002-08-16	6019	AWS	NO0003083107	Awilco ser. A	14.1	14.5	14.5	14.5	14.5	14.5	2000	29000	
4	2002-08-16	6020	AWSB	NO0003083115	Awilco ser. B	13.1	16.5	0	0	0	0	0	0	
5	2002-08-16	6024	BEA	NO0003102105	Bergesen d.y ser. A	142.5	150	159	159	148	150	27450	4198900	
6	2002-08-16	6025	BEB	NO0003102113	Bergesen d.y ser. B	130	136	133	133	130	130	7600	1009000	
7	2002-08-16	6026	BEL	NO0003094104	Belships	0.3	4	0	0	0	0	0	0	
8	2002-08-16	6034	BLO	NO0003679102	Blom	4	4.5	0	0	0	0	0	0	
9	2002-08-16	6035	BNB	NO0005139402	Bolig- og Næringsbanken	193	200	195	195	195	195	10050	1959750	
10	2002-08-16	6036	BNR	NO0003099608	Bergen Nordhordland	50	63	0	0	0	0	0	0	

Fig B-9 Stock: Equity <sql:stock_equity>

	Felt	Type	Attributter	Null	Standard	Ekstra
☐	id	int(10)	UNSIGNED	Nei		auto_increment
☐	news_head	varchar(75)		Nei		
☐	source	varchar(15)		Nei		
☐	datetime	datetime		Ja	*NULL*	
☐	textfield	text		Ja	*NULL*	

id	news_head	source	datetime	textfield
1	SCH-INTERIM FINANCIAL STATEMENT PER 2Q02	Oslo Børs	2002-08-16 08:16:13	At the Board meeting today, the Board of Director...
2	RIS - RAPPORT PR. 2. KVARTAL 2002	Oslo Børs	2002-08-16 08:17:02	Vedlagt på www.ose.no følger rapport for Rieber S...
3	SST - RESULTAT PR. 30-06-2002	Oslo Børs	2002-08-16 08:18:09	Steen & Strøm`s samlede driftsinntekter utgjorde ...
4	SEN - RESULTAT 2. KVARTAL 2002	Oslo Børs	2002-08-16 08:18:56	Innfør.av lov om dekktrykksens.i biler USA har fø...
5	NOD - HOVEDTALL 2 KV. 2002	Oslo Børs	2002-08-16 08:19:31	Som vedlegg på www.newsweb.no følger delårsrappor...
6	OCE - HALVÅRSRESULTAT	Oslo Børs	2002-08-16 08:20:17	Vedlagt på www.newsweb.no følger halvårsresultat ...
7	ISSG - DELÅRSRESULTAT 2.KVARTAL 2002	Oslo Børs	2002-08-16 08:21:09	Vedlagt på www.newsweb.no følger børsmelding, st...

Fig B-10 Stock News <sql:stock_news>

	Felt	Type	Attributter	Null	Standard	Ekstra
☐	id	int(10)	UNSIGNED	Nei		auto_increment
☐	trade_date	date		Ja	*NULL*	
☐	sec_id	int(10)		Ja	*NULL*	
☐	symbol	varchar(16)		Nei		
☐	sec_name	varchar(34)		Ja	*NULL*	
☐	index_type	char(2)		Ja	*NULL*	
☐	open	double		Ja	*NULL*	
☐	high	double		Ja	*NULL*	
☐	low	double		Ja	*NULL*	
☐	close	double		Ja	*NULL*	

id	trade_date	sec_id	symbol	sec_name	index_type	open	high	low	close
1	2002-08-16	700	OSEAX	Oslo Børs All-share Index_GI	SI	138.54	139.33	137.03	137.82
2	2002-08-16	701	OSE10GI	OSE10 Energy_GI	SI	140.63	141.43	138.24	138.91
3	2002-08-16	702	OSE1010GI	OSE1010 Energy_GI	SI	140.63	141.43	138.24	138.91
4	2002-08-16	705	OSE15GI	OSE15 Materials_GI	SI	181.04	181.56	176.16	179.96
5	2002-08-16	706	OSE1510GI	OSE1510 Materials_GI	SI	181.04	181.56	176.16	179.96
6	2002-08-16	712	OSE20GI	OSE20 Industrials_GI	SI	113.57	114.34	112.07	114.25
7	2002-08-16	713	OSE2010GI	OSE2010 Capital Goods_GI	SI	126.05	126.65	124.12	124.82
8	2002-08-16	721	OSE2020GI	OSE2020 Commercial Service & Suppl	SI	5.39	5.39	5.29	5.29
9	2002-08-16	723	OSE2030GI	OSE2030 Transportation_GI	SI	109.5	111.26	108.11	111.09
10	2002-08-16	729	OSE25GI	OSE25 Consumer Discretionary_GI	SI	134.8	139.3	134.8	139.2
11	2002-08-16	730	OSE2510GI	OSE2510 Automobiles & Components_G	SI	18.86	18.86	17.01	17.41

Fig B-11 Stock Shareidx <sql:stock_shareidx>

	Felt	Type	Attributter	Null	Standard	Ekstra
☐	id	int(10)	UNSIGNED	Nei		auto_increment
☐	tegn	varchar(15)		Ja	*NULL*	
☐	tekst	text		Ja	*NULL*	

id	tegn	tekst
1	Væren	No one will fault you for telling a little white l...
2	Steinbukken	Its easy to remain calm when everything is going y...
3	Vannmannen	If you stay in one place for too long, you run the...
4	Fiskene	There are many people making promises right now --...
5	Tyren	Even if you have a crystal ball to look into, dont...
6	Tvillingene	You are a burst of intensity, and everyone around

Fig B-12 Horoskop <sql:horoskop>

	Felt	Type	Attributter	Null	Standard	Ekstra
☐	id	int(10)	UNSIGNED	Nei		auto_increment
☐	type	varchar(20)		Ja	NULL	
☐	dato_fra	varchar(100)		Ja	NULL	
☐	dato_til	varchar(100)		Ja	NULL	
☐	sted	varchar(100)		Ja	NULL	
☐	tekst	text		Ja	NULL	

id	type	dato_fra	dato_til	sted	tekst
46	korttid	12:29, 18.08.2002	00:00, 20.08.2002	Østfold	Delvis skyet opphold og for det meste pent væ...
47	langtid	12:00, 18.08.2002	25.08.2002	Akershus	Stort sett opphold. Onsdag skiftende bris. Til del...

Fig B-13 Været <sql:vaer>

	Felt	Type	Attributter	Null	Standard	Ekstra
☐	id	int(10)	UNSIGNED	Nei		auto_increment
☐	nr	int(2)		Ja	NULL	
☐	direction	varchar(10)		Ja	NULL	
☐	last	int(2)		Ja	NULL	
☐	high	int(2)		Ja	NULL	
☐	count	int(2)		Ja	NULL	
☐	title	varchar(75)		Nei		
☐	artist	varchar(75)		Nei		
☐	singlepl	int(2)		Ja	NULL	
☐	albump	int(2)		Ja	NULL	
☐	album	varchar(75)		Ja	NULL	
☐	etikett	varchar(75)		Ja	NULL	
☐	genre	varchar(30)		Ja	NULL	
☐	hid	int(10)		Ja	NULL	
☐	week	int(2)		Ja	NULL	
☐	year	int(4)		Ja	NULL	
☐	name	varchar(65)		Ja	NULL	

id	nr	direction	last	high	count	title	artist	singlepl	albump	album	etikett	genre	hid	week	year	name
1	1	Eq	1	1	12	A Little Less Conversation	Elvis vs. JXL	1	0	(single)		Dance	25400	32	2002	XML hit-40
2	2	Up	3	2	5	KÃƒÂ¤rleken vÃƒÂ¤ntar	Kent	0	4	Vapen & Ammunition		Rock	25451	32	2002	XML hit-40
3	3	Down	2	2	7	Lifelines	A-Ha	18	14	Lifelines		Pop	25427	32	2002	XML hit-40
4	4	Up	13	4	10	Ive Got You	Marc Anthony	0	6	Mended		Voksen Pop	25403	32	2002	XML hit-40
5	5	Up	6	5	10	Get Over You	Sophie Ellis Bextor	0	0	Read My Lips		Dance	25423	32	2002	XML hit-40
6	6	Up	8	2	16	Underneath Your Clothes	Shakira	4	16	Laundry Service		Dance	25368	32	2002	XML hit-40
7	7	Up	10	3	10	Can Let Go	Maria Arredondo	0	0	-			25425	32	2002	XML hit-40
8	8	Down	5	1	13	A Thousand Miles	Vanessa Carlton	0	0	Be Not Nobody		Voksen Pop	25399	32	2002	XML hit-40
9	9	Eq	9	9	9	Precious Illusions	Alanis Morissette	0	0	Under Rug Swept		Rock	25390	32	2002	XML hit-40
10	10	Down	4	3	19	If Tomorrow Never Comes	Ronan Keating	7	11	Destination		Voksen Pop	25344	32	2002	XML hit-40
11	11	Up	38	11	3	Im Alive	Celine Dion	0	0	A New Day Has Come		Voksen Pop	25470	32	2002	XML hit-40
12	12	Up	18	12	6	In My Place	Coldplay	0	0	A Rush Of Blood To The Head		Rock	25450	32	2002	XML hit-40
13	13	Down	7	7	12	I Used to Be	Number Seven	0	0	(single)		Pop	25398	32	2002	XML

Fig B-14 Hitcompany <sql:hitcompany>

C Kildekode, WEB-grensesnitt

Under følger komplett oversikt over kildekode til implementasjonens WEB-grensesnitt..

C.1 Oversikt

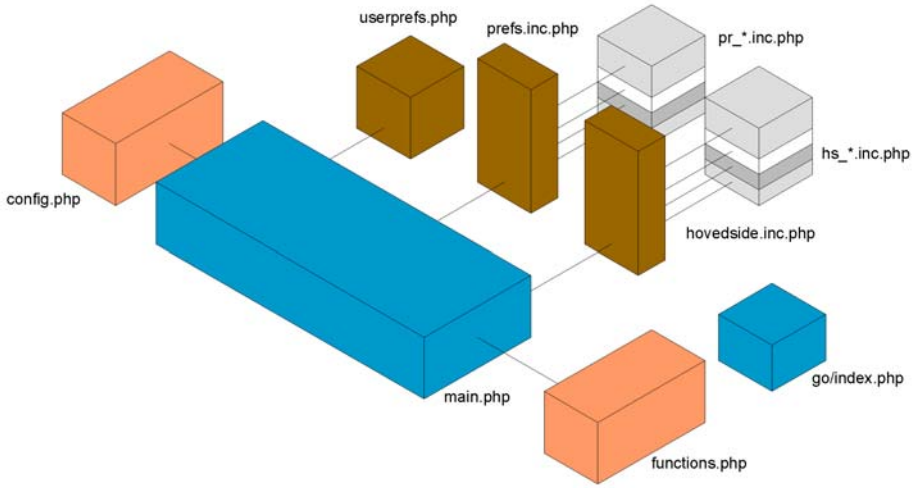

Fig C-1 Oversikt, kildekode til WEB-grensesnitt

[Fig C-1] viser forholdet mellom de enkelte kildekode-filene i webgrensesnittet. Figuren er simplifisert noe. Grafikk-elementer (gif og jpg) samt endel ubrukte filer er tatt bort fra figuren for å gjøre det hele mer oversiktelig.

Kildekoden er desverre ikke, av sikkerhetsmessige grunner, tilgjengelig denne elektoniske utgaven av avhandlingen.

D Kildekode, PDA-grensesnitt

Her følger kildekoden som tilhører PDA-grensesnittet til implementasjonen.

D.1 Oversikt

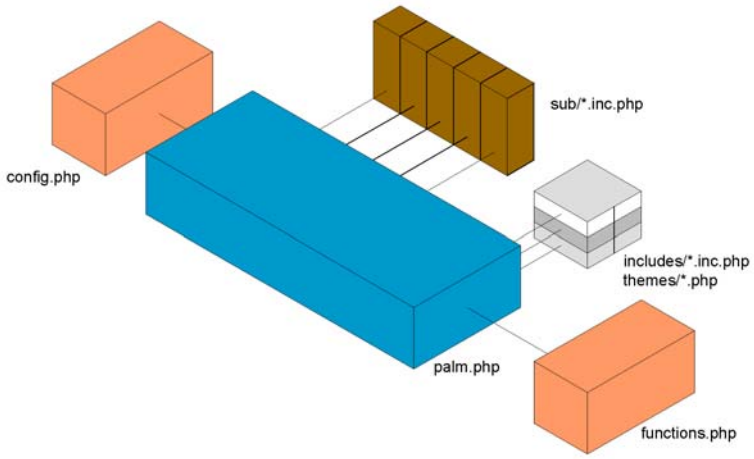

Fig D-1 Oversikt, kildekode til PDA grensesnitt

[Fig D-1] viser forholdet mellom de enkelte kildekode-filene i pdagrensesnittet. Figuren er simplifisert noe. Grafikk-elementer (gif og jpg) samt endel ubrukte filer er tatt bort fra figuren for å gjøre det hele mer oversiktelig.

Kildekoden er desverre ikke, av sikkerhetsmessige grunner, tilgjengelig denne elektoniske utgaven av avhandlingen.

E Kildekode, backend-agenter

Her følger noen eksempler på backend-agenter i prototypen..

E.1 Oversikt

Backend-agentene består av små induviduelle programmer som henter ned, bearbeider og lagrer data, brukerprofiler og samlingsmetadata.

De fleste av disse kjøres i regelmessige intervaller (ved hjelp av cron script) og oppdaterer databasene, rydder opp og tar backup uten noe manuelt arbeid.

Kildekoden er desverre ikke, av sikkerhetsmessige grunner, tilgjengelig denne elektoniske utgaven av avhandlingen.