Kontekstbasert arbeids- og informasjonsorganisering

Hovedoppgave
ved
sivilingeniørutdanning i
informasjons- og kommunikasjonsteknologi

av
Knut Håkon Tolleshaug Mørch

Grimstad, 28. mai 1999

Forord

Når jeg tenker tilbake på hva jeg gledet meg til å jobbe med i forbindelse med denne diplomoppgaven i forhold til hva jeg anser som det viktigste jeg har lært etter at oppgaven er ferdig, slår det meg at det er to ganske forskjellige ting. Før jeg begynte hadde jeg som hovedmål at noe skulle utvikles, og at erfaringene ved å utvikle en prototyp ville bli det viktigste. Disse erfaringene anser jeg fremdeles som verdifulle, men kanskje det viktigste jeg har oppnådd for egen del ved diplomarbeidet er de nye tankene jeg har fått omkring brukerens situasjon ved interaksjon med et informasjonssystem og visualisering av informasjon for brukeren. Problemstillinger, refleksjoner jeg har gjort meg og tanker jeg har fått under arbeidet med diplomoppgaven tror jeg vil være svært verdifullt å ha med seg ved videre arbeide med informasjonssystemer.

Derfor vil jeg ikke bare rette en stor takk til min veileder Lars Line for veiledning under diplomarbeidet, men også for de nye idéene og tankene jeg har fått omkring brukeres interaksjon med et informasjonssystem, som følge av veiledning og individuelt arbeide med diplomoppgaven. En takk rettes også til biblioteket ved HiA for stor hjelp ved søk etter informasjon, og Jan Veikko Granroth for lån av koordineringsmodellen som han har tegnet.

Grimstad, 28. mai 1999

Knut Håkon Tolleshaug Mørch

Innholdsfortegnelse

Sammendrag

I denne oppgaven ble først begrepet kontekstbasert arbeids- og informasjonsorganisering posisjonert i forhold til relaterte fagområder. Begrepet kontekstbasert arbeids- og informasjonsorganisering fantes ikke som frittstående begrep i den teorien som ble undersøkt. De ulike fagområdene behandlet i varierende grad problemstillinger relatert til kontekstbasert arbeids- og informasjonsorganisering. Knowledge Management ble funnet som det fagområdet som hadde mest fellestrekk med begrepet.

For å kunne klassifisere ulike kontekster ble en del parametre vurdert. De som ble funnet best egnet, basert på vurdering i forhold til avgrensninger satt for oppgaven og krav til informasjonssystemer som skal støtte kontekstbasert arbeids- og informasjons-organisering, ble presentert og siden implementert i Office 2000 og Exchange.

Metainformasjon viste seg å være velegnet som metode for kontekstbasert arbeids- og informasjonsorganisering (KAIO). Arbeidskonteksten kunne klassifiseres på grunnlag av noen parametre som ble diskutert og valgt, og disse parameterne kunne implementeres v.h.a. metainformasjon. Ved å ta utgangspunkt i Word-filer og Mail- og Document-objektet fra Outlook 2000 objekt-modellen, ble det vist at det er mulig å implementere de parameterne for klassifisering av kontekster som ble funnet, i Office 2000 og Exchange.

Office 2000, Exchange, Active Server Pages (ASP) og Collaboration Data Objects (CDO) ble studert for å sette seg inn teknologien som skulle illustrere løsningen. Det ble gjennomført en praktisk utprøving av Office 2000 og Exchange ved å teste ulike VBScript basert på ASP og CDO. Dette gav erfaringer med teknologien, og grunnlag for å vurdere hvilke muligheter teknologien gav for KAIO.

Det ble funnet at Word 2000 kunne benyttes sammen med Exchange for lagring av dokumenter. Document Management Extensions (DME) viste at denne koblingen var mulig. For at Word 2000 skal kunne benytte kontekstbasert arbeids- og informasjonsorganisering, er det nødvendig med utvikling av en løsning tilsvarende DME, men med mulighet for å benytte Document-objektet og setting av parameterne for klassifisering av ulike kontekster ved lagring og åpning av dokumenter.

Outlook 2000 var det programmet i Office 2000 som best egnet seg for kontekstbasert arbeids- og informasjonsorganisering. Programmet hadde sammen med Exchange mulighet for å manipulere dataelementer i foldere avhengig av ulike kontekster, basert på ulike begivenheter. Det var også mulig å assosiere en web-applikasjon med en Folder Homepage og v.h.a. denne presentere informasjon fra ulike informasjonsdomener basert på brukerens arbeidskontekst.

Ved å benytte kontekstbasert arbeids- og informasjonsorganisering vil brukeren få bedret muligheten for å få visualisert eller finne kun den informasjonen som er relevant for den aktuelle konteksten. Dette forutsetter at brukeren setter de ulike parameterne ved lagring av dataelementer. Dette kan sikres ved å benytte forms i Outlook 2000 eller en løsning som er lignende DME i Word 2000.

1 Innledning

Informasjonsteknologi er i svært mange organisasjoner og virksomheter en naturlig og nødvendig del av hverdagen. Oppgaver som før var tidkrevende og uoversiktlige kan i dag effektiviseres og tilrettelegges på en annen og i mange tilfeller bedre måte v.h.a. IT. Dette kan føre til organisatoriske omstruktureringer som gir betydelige fordeler, men også utfordringer.

Før man tok i bruk IT i organisasjoner i samme grad som i dag, ble mange av de samme oppgavene utført v.h.a. analoge lagringsmetoder og medier. Istedenfor databaser brukte man arkiver, og brev ble sendt med postvesenet istedenfor over Internett. Innføring av IT har ført til at mennesker i en organisasjon har tilgang til mer informasjon enn før, på en raskere måte.

Dette kan igjen føre til at individer får presentert mer informasjon enn de har behov for, og det kan være tidkrevende å finne frem til riktig informasjon avhengig av den oppgaven man skal utføre. Windowsbaserte systemer tilbyr mulighet for et personlig oppsett av desktop'en, eller det virtuelle arbeidsmiljøet. Dette gjør at en bruker får et grensesnitt som er optimalisert for denne ved pålogging til en arbeidsstasjon. Et mål for mange softwareselskaper i dag er at dette personlige oppsettet skal være tilgjengelig uansett hvor i verden man logger seg på et spesielt nettverk.

Denne oppgaven er gitt med en ide om at dette bare er en del av utviklingen mot et ideelt informasjonssystem som automatisk finner og lagrer riktig informasjon for brukeren, avhengig av hva denne jobber med. Som en ytterligere tilnærming er det nødvendig for systemet ikke bare å identifisere brukeren, men også hvilke oppgaver denne utfører eller hva slags roller vedkommende har. Dermed er det mulig å identifisere hvilken kontekst man jobber i, og koble brukeren opp mot riktig informasjon avhengig av dette.

Dette er en problemstilling som i varierende grad er behandlet i ulike teorier, og en del av oppgaven har derfor gått ut på å kartlegge nærliggende forskningsområder, og posisjonere begrepet kontekstbasert arbeids- og informasjonsorganisering i forhold til dette. Videre var det gitt at bruk av metainformasjon, spesielt arbeidskontekst, åpner for bedre mekanismer for lagring og gjenfinning av informasjon. Derfor har metadata blitt studert og redegjort for, og det er også for metadata gjort et søk etter relevant forskning for lære av andres erfaringer og problemer, få ideer og se på hvilke muligheter bruk av metainformasjon gir.

Problemet ved bruk av metainformasjon er hvordan man skal sikre kvalitet og minimalisere innsats. Denne problemstillingen er behandlet, og har resultert i valg av en del parametre for lagring og gjenfinning samt hvordan metadataene best kan organiseres. Dette har igjen resultert i utviklingen av en modell som skal sikre kvalitet og minimalisere innsats ved bruk av metainformasjon.

Modellen er benyttet sammen med Office 2000 og Exchange for å se hvordan denne teknologien kan benyttes for kontekstbasert arbeids- og informasjonsorganisering. Bruk og valg av parametre som finnes i denne teknologien er diskutert i forhold til modellen.

Til slutt er det diskutert hvordan kontekstbasert arbeids- og informasjonsorgansisering kan implementeres i Office 2000 og Exchange. Det gis eksempler på muligheter for dette med den aktuelle teknologien og hva som kan utvikles for ytterligere å støtte kontekstbasert arbeids- og informasjonsorgansisering.

2 Oppgaven

Nedenfor vil oppgaven først bli presentert som den ble gitt, med noen forklaringer av hva som legges i ulike begreper. Deretter vil ideen bak oppgaven ytterligere utdypes slik den har fremkommet i samtale med veileder og under arbeidet med diplomoppgaven. Til slutt vil det bli gjort noen avgrensninger for oppgaven og presisiering av målet for oppgaven og hva man ønsker å komme frem til.

2.1 Oppgaven

Nedenfor gjengis oppgaven slik den er gitt:

> ***Kontekstbasert arbeids- og informasjonsorganisering***
>
> *To sentrale problemer med dagens IT-verktøy og samarbeidsteknologi er at vi arbeider i adskilte informasjonsdomener samt at informasjonen i hvert domene i beste fall er enkelt organisert.*
>
> *Med informasjonsdomener menes her teknologiske plattformer som Internett, lokalnett, gruppevare eller databaser. Egentlig arbeider vi stort sett i forskjellige verktøy som i større og mindre grad forholder seg til de underliggende informasjonsdomenene. Teknologisk sett er det utvilsomt kun Internett baserte tjenester som evner å bygge bro mellom underliggende informasjonsdomener. Vi ser allerede klare tegn på at flere og flere verktøy forholder seg til Internett teknologi, både i brukergrensesnitt og til informasjonslagring. Se også http://fag.grm.hia.no/it4200/Litteratur/DOC/Forrester/InternetComputing.htm.*
>
> *Med enkel organisering menes at informasjonen ofte er organisert i enkle hierarkiske strukturer som kan være tungvint å operere i. Metainformasjon, spesielt arbeidskontekst, åpner for bedre mekanismer for lagring og gjenfinning. Problemet er hvordan en skal sikre kvalitet og minimalisere innsats for å spesifisere metainformasjon.*
>
> *Oppgaven består av en teoridel som skal belyse overnevnte problemer og søke etter relevant forskning. Problemstillingen skal videre illustreres ved å utvikle en prototyp basert på Weboffice, Exchange og Office 2000.*

Med begrepet kontekstbasert arbeids- og informasjonsorganisering menes det hvordan IKT kan støtte ulike arbeidskontekster ved bedre organisering av det virtuelle arbeidsmiljøet og underliggende informasjonsdomener. Det vil ikke bli fokusert på hvordan selve arbeidet bør utføres avhengig av konteksten.

2.2 Idèen bak oppgaven

Dagens informasjonsystemer tilbyr som oftest en form for personlig oppsett av det virtuelle arbeidsmiljøet (desktop'en) som vises på skjermen. En trend er at dette ansees så

viktig at man innenfor et nettverk ønsker å kunne tilby det samme oppsettet uavhengig av hvor man befinner seg i verden eller hvilken terminal man benytter for å koble seg på nettverket.

Dette har den fordel at systemet gjenkjenner brukeren og kan tilby aksess til informasjon brukeren benytter eller er gitt tilgang til. Brukeren vil imidlertid ofte utføre forskjellige oppgaver, og dermed befinne seg i ulike kontekster. Dette har som konsekvens at når brukeren er i en spesiell kontekst, får vedkommende presentert mer informasjon enn ønskelig eller har problemer med å finne frem til riktig informasjon. For at brukeren skal kunne arbeide effektivt og utnytte den informasjonen som finnes i informasjonssystemet (IS) er det et behov for oppdeling av informasjonen som presenteres avhengig av arbeidskonteksten.

Et ideelt informasjonssystem vil kunne tilby brukeren aksess til den – og bare den – informasjonen som er relevant for den oppgaven vedkommende skal utføre. Ideen er da at ved å gå et skritt videre fra personlig oppsett av det virtuelle arbeidsmiljøet til også å ha et kontekstbasert oppsett, vil man komme enda nærmere det ideelle informasjonssystem. Det må da være mulig å kunne spesifisere noen parametre for å bestemme hvilken arbeidskontekst man er i, og finne en metode for å sikre kvalitet og minimalisere innsats for å beskrive dette, som nevnt i kapittel 2.1.

2.3 Mål og avgrensninger for oppgaven

Utgangspunktet for denne oppgaven er kontekstbasert arbeids- og informasjons-organisering i grupper av liten til medium størrelse. Grupper på 15-20 personer er tenkt som et utgangspunkt. Den typen gruppe som er utgangspunkt for oppgaven kjennetegnes ved at individene i gruppen ikke bare forholder seg til de andre i gruppen, men også til personer eller aktører som ikke alltid har relevans i forhold til hverandre. Som et eksempel kan spesielt en prosjektgruppe trekkes frem, fordi denne kan bestå av et varierende antall medlemmer og har begrenset varighet.

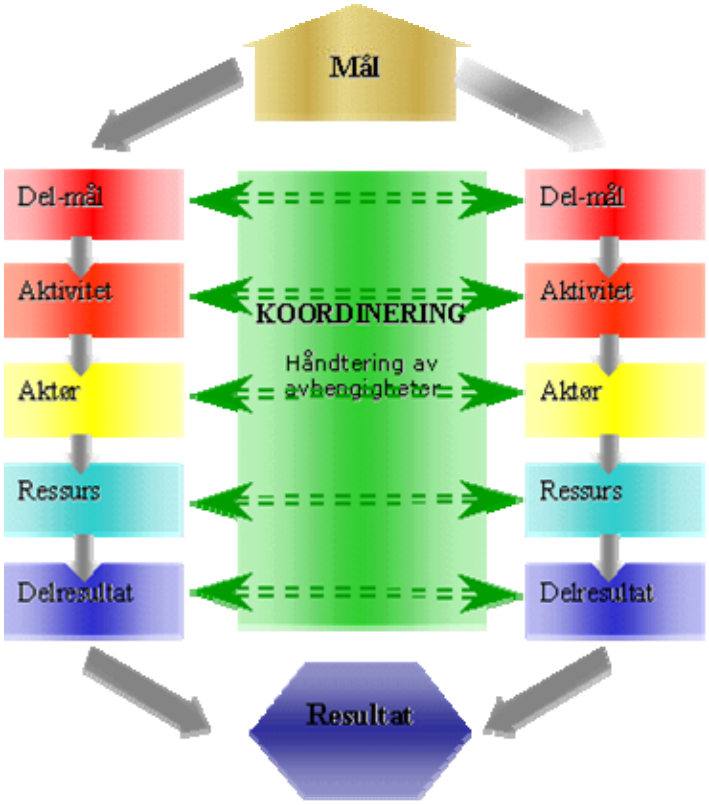
For det enkelte individ vil det imidlertid være mulig å være del av flere prosjektgrupper eller faste avdelinger. I en slik situasjon vil individet utfører flere ulike typer oppgaver og inneha ulike roller som ikke nødvendigvis har sammenheng med hverandre. Et mål for oppgaven er derfor å gjøre rede for hvordan man kan finne informasjon tilhørende en slik gruppe når man utfører oppgaver relatert til denne spesielle gruppen.

Problemstillinger i forbindelse med svært store grupper eller hele organisasjoner samt spesielle krav for enkeltbedrifter er derfor utelatt. Dette fordi et av målene med oppgaven er å finne en metode for kontekstbasert arbeids- og informasjonsorganisering som ikke krever store endringer for f. eks. hvert nye prosjekt.

Typen informasjonssystem (IS) som er utgangspunkt for denne oppgaven er ikke store rigide systemer med krav til sporbarhet. Det fokuseres på IS som ikke er laget spesielt til en type oppgaver, men som enkelt kontinuerlig må kunne tilpasses nye oppgaver. Dette kan f. eks. være et IS i oppdragsbasert virksomhet.

Oppgaven defineres også til ikke å omfatte indeksering av store informasjonsdatabaser. Databasene vil være relativt små, med informasjon tilhørende f. eks.ulike prosjekter av mindre størrelse og med grupper som beskrevet i begynnelsen av dette avsnittet.

For å illustrere hva man ønsker å oppnå med denne oppgaven er Schiefloe og Syvertsen [18] sin koordinasjonsmodell hensiktsmessig å benytte.



Figur 2.1 Koordinasjonsmodell etter Schiefloe og Syvertsen [18]

Som vi ser vil en oppgave medføre en aktivitet, som igjen vil medføre at en aktør benytter en ressurs for å oppnå et resultat. Målet med denne oppgaven blir derfor i forhold til modellen å finne ut hvordan man kan knytte riktig ressurs opp mot aktøren avhengig av hvilken aktivitet denne utfører. For at dette skal kunne realisere må følgende spørsmål besvares:

1. Hvordan bestemme riktig kontekst enkelt og samtidig sikre høy kvalitet?
2. Hvilke parametre må velges for å kunne klassifisere ulike kontekster?
3. Hvordan organisere informasjon/ virtuelt arbeidsmiljø slik at brukeren enkelt finner frem til ønsket informasjon?

Disse spørsmålene vil det være et hovedmål for oppgaven å besvare.

3 Beskrivelse av fremgangsmåte

Begrepet kontekstbasert arbeids- og informasjonsorganisering ble brukt som et utgangspunkt for oppgaven. Begrepet fantes ikke i annen kjent forskning. Idéen bak oppgaven var klar, men det var nødvendig å posisjonere begrepet i forhold til annen relevant forskning. Dermed måtte det gjennomføres et søk for å finne ut hva som var gjort av relevant forskningsarbeid, og om det fantes begreper som kunne brukes for å definere problemstillingen.

Det var ønskelig å finne ut om begrepet fantes i andre former, og om det var gjort funn i annen forskning som hadde relevans for arbeidet i denne oppgaven. Et søk etter relevant forskning ville også gi mulighet for å se hvordan andre tilnærmet seg problemstillingen, hvilken innfallsvinkel de hadde og hva de la vekt på samt i hvilke situasjoner problem-stillingen oppstår.

For å begrense søket ble det fokusert på forskningsarbeid eller teorier som:

- beskrev IT strategisk brukt i en organisasjon
- beskrev interaksjonen mellom bruker og IT-utstyr
- beskrev hvordan IT kan støtte personer, grupper og prosjektarbeide

Noen teorier ble også valgt ut i samarbeid med veileder fordi det ble antatt at de ville kunne belyse begrepet kontekstbasert arbeids- og informasjonsorganisering. Med dette utgangspunktet ble søket fokusert omkring følgende: Computer-Supported Cooperative Work (CSCW), IT-støttede prosessforbedringer, Knowledge Management (KM), Concurrent Engineering og kontekstbasert design. Søket ble gjort på Internett v.h.a. søkemaskiner som Altavista og Metacrawler, i ulike databaser fra IEEE og ASM og ved å benytte skolens bibliotek. Referanselister i papere og bøker ble også benyttet for å finne relevant forskning.

På bakgrunn av søket ble det foretatt en posisjonering i forhold til relaterte teorier for å få ideer og begrepsapparat fra forskning/ fagfelt som ligger nær opp til arbeidet i denne oppgaven. Dette gjorde det mulig å definere i hvilke situasjoner det er et behov for kontekstbasert arbeids- og informasjonsorganisering og hvilke krav det stilles til et informasjonssystem for å støtte dette.

Metainformasjon var gitt i oppgaven som utgangspunkt for bedre mekanismer for lagring og gjenfinning. Fordi bruk av metadata var relativt ukjent for meg, måtte metadata studeres nærmere for å sette seg inn i hva metadata var, og om det var definert hvordan metadata skulle representeres og implementeres, noe det var i ISO-standarden. Videre ble det gjennomført et søk blant relevant forskning for å få ideer for hvordan meta-informasjon kunne benyttes, hvilke muligheter som fantes og for å lære av andre erfaringer.

Søket gav, sammen med kravene som var funnet for IS som skulle støtte kontekstbasert arbeids- og informasjonsorganisering, mulighet for å velge parametre for lagring og

gjenfinning og dermed utvikling av en modell for kontekstbasert arbeids- og informasjonslagring. Denne modellen fokuserte på hvordan informasjonen ble lagret og hvordan ulike kontekster kunne identifiseres. Hvilken teknologi som skulle brukes for å illustrere hvordan et virtuelt arbeidsmiljø kunne være organisert for å støtte ulike arbeidskontekster var definert i oppgaven, og dette ble derfor skilt ut i et eget kapittel.

For å kunne illustrere hvordan et slikt viruelt arbeidsmiljø kunne organiseres v.h.a. valgte teknologi, var det nødvendig å sette seg inn i denne. Fordi Weboffice er utviklet med Active Server Pages (ASP) og Visual Basic(VB)-scripting, var det nødvendig å sette seg inn i ASP og VBScript. Det ble derfor brukt en del tid på å sette seg inn i disse to. Collaborative Data Objects (CDO) er grensesnittet til Exchange/ Outlook 2000 ved utvikling av applikasjoner basert på VB-script. En del tid ble derfor også benyttet til å sette seg inn i CDO.

Weboffice ble funnet som lite hensiktsmessig å benytte for illustrasjon av løsningen, bl. a. fordi dette ville bety å sette seg inn i enda en teknologi i tillegg til Office 2000 og Exchange. Med teknologi menes her Weboffice, ikke ASP eller CDO. Weboffice var valgt ut som teknologi for å tilby Internett-baserte tjenester.

Det ble etter vurdering funnet at de Internett-baserte tjenestene som ble funnet hensiktsmessig å tilby ved kontekstbasert arbeids- og informasjonsorganisering kunne kobles mot en Folder Homepage i Outlook 2000. En webside assosiert med en Folder Homepage i Outlook 2000 kunne konstrueres enklere enn Weboffice, fordi når Folder Homepage ble brukt for å aksessere en webside, ville brukeren ha tilgang på funksjoner i Outlook 2000.

Dermed var det ikke behov for å tilby funksjoner Outlook 2000 hadde i en web-applikasjon. I tillegg til dette ble det ved forespørsel til veileder avklart at Weboffice ikke for enhver pris måtte benyttes. P.g.a. disse faktorene ble Weboffice valgt bort som utgangspunkt for løsning.

Under studiet av ASP, VBScript og CDO, ble det funnet en ny teknologi som tilbød noen av de mulighetene som var tenkt utviklet i prototypen: Document Management Extensions (DME) fra 80-20 Software. Sammen med Word 2000, Outlook 2000 og Exchange gav denne teknologien nye muligheter for kontekstbasert arbeids- og informasjonsorganisering.

Etter vurdering av mulighetene DME gav, tiden som var til rådighet og i samråd med veileder ble det besluttet å bruke tiden som var igjen til å jobbe med en teoretisk løsning på hvordan kontekstbasert arbeids- og informasjonsorganisering kunne implementeres i Office 2000 og Exchange. Det videre arbeidet ble derfor konsentrert rundt utnyttelse av Office 2000 og Exchange og forslag til hvordan teknologien ytterligere kan forbedres for å støtte kontekstbasert arbeids- og informasjonsorganisering.

En god del tid hadde blitt brukt til å sette seg inn i en teknologi uten at det ble utviklet mye, men den kunnskapen som var blitt tilegnet gjennom dette studiet gav et godt

grunnlag for vurdering av hvordan teknologien kunne benyttes. Gjennom studiet ble objektmodellen for CDO nøye studert, og dermed hvilke muligheter som fantes for setting av parametre for bestemmelse av kontekster. Studiet gav også kunnskap om hvilke muligheter teknologien gav, og dermed også et godt utgangspunkt for å komme med forslag til forbedringer.

Til slutt i oppgaven ble det foreslått hvordan Office 2000 og Exchange kunne utnyttes for å støtte kontekstbasert arbeids- og informasjonsorganisering. Dette ble gjort med utgangspunkt i modellen utviklet tidligere og de parameterne som ble funnet for å bestemme ulike kontekster.

4 Kartlegging av nærliggende forskningsområder

Begrepet kontekstbasert arbeids- og informasjonsorganisering ble brukt som utgangspunkt for oppgaven. Det fantes ikke i annen kjent forskning. Idéen bak oppgaven var klar, men det var nødvendig å posisjonere begrepet i forhold til annen forskning. Dermed måtte det gjennomføres et søk for å finne ut hva som var gjort av relevant forskningsarbeid, og om det fantes begreper som kunne brukes for å definere problemstillingen. De teoriene som er behandlet her ble ansett som mest sannsynlige å kunne si noe om kontekstbasert arbeids- og informasjonsorganisering, og ble derfor undersøkt nærmere. Disse ble valgt ut i samarbeid med veileder.

4.1 CSCW og gruppevare

CSCW og gruppevare tar spesielt for seg hvordan IT kan brukes til å kommunisere, samarbeide og koordinere arbeid. Fagområdet er en viktig kilde til forståelse for hvorfor overgangen fra analoge til digitale lagringsformer og kommunikasjon har medført problemer og hva man har gjort for å løse disse de seneste årene.

4.1.1 Historisk bakgrunn

Computer-Supported Coopterative Work (CSCW) er et vidt begrep. Ifølge Grudin og Poltrock [3] er det i dag mer å regne som et forum for å dele ideer fra ulike disipliner, enn et eget avgrenset fagområde. Innenfor dette forumet har man bl. a. diskutert hvordan gruppevare kan brukes for å støtte arbeid i små grupper og prosjekter av ulike størrelser.

I 1960-årene fokuserte man på bruk av datastøtte fra store sentrale transaksjons-systemer for å booke flyseter, holde kontroll på lagre eller skrive ut lønnssjekker. Fra slutten av 1970-årene og i løpet av 1980-årene brukte man det man da omtalte som minidatamaskiner (som i dag har blitt utviklet til det vi kjenner som PC‘er) til å interaktivt støtte grupper og organisasjoner på andre måter, som f. eks. elektroniske bestillingssystemer, ekspertsystemer og dokumentbehandling. Dette hadde betegnelser som „Office Automation“ eller „Software Engineering“ [8]. Førstnevnte har fellestrekk med det som i dag kalles workflow-management.

Problemer ved å ta i bruk denne teknologien kom delvis av at den hadde begrensninger, men også av at man møtte sosio-tekniske utfordringer ved innføring av datastøttet samarbeid og kommunikasjon. Derfor ble CSCW opprettet av teknologer som et forsøk på å lære fra økonomer, sosiologer, psykologer, antropologer, organisasjonsteoretikere, lærere og andre som kunne bidra med kunnskap om gruppeaktivitet [3].

Med CSCW ble fokuset endret fra datastøtte til organisasjoner og individ, til datastøttet arbeide i mindre grupper og prosjekter. Innenfor dette området finner man produkter som omtales som gruppevare. Gruppevare består av applikasjoner for dokumenthåndtering, arbeidsflyt, e-post, konferanser og kalender/avtalebok. Funksjoner som blir støttet er kommunikasjon, informasjonsdeling samt samarbeid og koordineringsstøtte.

4.1.2 Definisjon av gruppevare

Gruppevare blir brukt for å beskrive de teknologiske løsningene og utfordringene, mens CSCW blir brukt for å beskrive forskning. Grudin [8] tar for seg hvordan ulike papere definerer gruppevare. Resultatet er interessant for hvordan de oppfatter betydningen av ulike roller og arbeidskontekst.

2 av 4 papere vil inkludere databaser blandt gruppevare. Det tredje (skrevet 1990) vil inkludere elektronisk mail og avansert gruppevare, men eksludere databaser og underliggende teknologier. Begrunnelsen for dette er at bortsett fra passordkontroll er ikke disse databasene klar over ulike roller eller kommunika-sjonsbehov i en gruppe. Ved denne definisjonen vil databaser som varsler spesielle mennesker om spesifikke endringer kvalifisere til betegnelsen gruppevare.

Det siste paperet – også dette fra 1990 – eksluderer også e-mail, med den begrunnelse at e-mail som kun brukes til å spre generell organisasjonsinformasjon ikke støtter gruppeaktivitet. For at den skal gjøre det må det være et e-mailsystem med brukere som lager aliaser, distribusjonslister og komplekse bruksmønstre som endres fra prosjekt til prosjekt. Det må tilføyes at dagens e-mailsystemer i stor grad tilfredsstiller dette kravet v.h.a. ulik funksjonalitet som har blitt utviklet senere år.

4.1.3 Sosiotekniske problemer og utfordringer ved bruk av gruppevare

Gruppevare fører til mange nye muligheter ved arbeide i mindre grupper eller et prosjekt, men også til problemer man før ikke hadde. Disse er ofte av sosioteknisk karakter, eller et resultat av at individet mottar så mye informasjon at det ikke klarer å nyttiggjøre seg denne optimalt. Dette fordi mennesker ikke har evnen til å forholde seg til mange elementer i en problemløsningssituasjon. Forskning innen kognitiv psykologi har vist at gjennomsnittsmennesket klarer å behandle 7 ulike elementer eller grupperinger på en gang [4].

Grudin og Poltrock [3] peker på både tekniske, sosiale og organisatoriske utfordringer ved bruk av gruppevare. De viktigste tekniske utfordringene er kompabilitet mellom ulik gruppevare og over ulike informasjonsdomener, og fleksibilitet i bruk av løsningene. Ulike gruppevareløsninger som e-mail, kalendere og dokumentbehandling må både kunne nåes av hverandre, og kunne brukes sammen med andre applikasjoner som f. eks. en skriveapplikasjon.

I paperet har de summert opp 8 punkter de anser som sentrale ved et sosialt/ organisatorisk perspektiv på gruppevare. I denne oppgaven er enkelte punkter spesielt relevante, og disse listes opp og utdypes nedenfor:

- Gruppevareapplikasjoner fører ofte til at folk må gjøre tilleggsarbeid. I mange tilfeller er ikke dette primære arbeidsoppgaver, og de vil ikke selv se effekten av eller ha nytte av dette arbeidet. Dette fører til at tilleggsarbeidet ikke blir utført.

Derfor er det et mål at nye gruppevareapplikasjoner fører til minst mulig tilleggsarbeid for brukerne.

- Behandling av spesielle og unntaksvise tilfeller. I gruppearbeid foregår det mye improvisasjon og forandringer som fører til nye handlingsmønstre. En applikasjon må derfor være adaptiv, og kunne brukes selv om mål eller arbeidsprosesser endres.
- Applikasjoner må være intuitive. Både for at de skal være lette å ta i bruk, og for om mulig å unngå behov for store ressurser for drift av løsningen.

4.2 IT-støttede prosessforbedringer

Teorien om „the economic man“ sier at mennesker eller aktører i et marked alltid vil maksimere de muligheter som finnes. Sett i denne sammenheng vil det alltid være et ønske om prosessforbedringer i virksomheter. Dette vil forbedre konkurranseevnen og øke investorenes avkastning. IT gir nye muligheter i denne sammenheng. Som vi skal se kan det også forekomme kontekstbaserte utfordringer ved slik bruk av IT.

4.2.1 Historisk bakgrunn

Prosessforbedringer er ikke et nytt fenomen, men fokuset og mulighetene har endret seg. På 60-/70-tallet var den dominerende konkurransefilosofien masseproduksjon. Man ønsket standardiserte varer og tjenester billigst mulig, og satset på en hierarkisk oppbygning av organisasjonen.

Mot 90-tallet ble kvalitetsproduksjon et sentralt begrep. Sterkt inspirert av Toyota Motor Company og andre japanske bedrifter fokuserte man på kontinuerlig forbedring og kvalitetsstyring. Effektivitet og standardisering er også her sentralt, men i tillegg et ønske om høyest mulig kvalitet. I slike bedrifter ble ofte selvstyrte grupper benyttet. En sentral teoretisk retning er Total Quality Management (TQM), en samling idéer og teknikker som skal bidra til å bedre en virksomhets konkurranseevne gjennom kontinuerlig forbedring av kvaliteten på virksomhetens produkter og prosesser [5]. TQM er også i dag en ledelsesfilosofi som står sterkt i mange bransjer.

I dag er masseskreddersøm et sentralt begrep, og på mange måter et svar på 90-tallets nye markeder og krevende kunder. Denne formen for konkurranse-tilpasning krever fleksibilitet og evne til rask omstilling av både produksjons-apparat, organisasjon og medarbeideres kunnskap. I tillegg til krav om effektivitet, lav pris og høy kvalitet har det kommet et krav om skreddersydde produkter og tjenester. Vedvarende eller økt konkurranseevne krever i dag evne til rask omstilling og virksomhetsutvikling. Dette setter krav til valg av produkter og markeder, og sentrale virkemidler er kompetanse og teknologi.

Venkatraman [9] beskriver 5 ulike nivåer for hvordan IT brukes ved endringsprosesser. I denne oppgaven er spesielt nivå 3 og 4 av interesse, BPR (Business Process Redesign) og BNR (Business Network Redesign). Disse blir karakterisert som revolusjonerende fordi de radikalt endrer ulike prosesser i virksomhetene. Nedenfor vil disse bli beskrevet.

4.2.2 Business Process Reengineering

De fleste virksomheter er i større eller mindre grad funksjonsorganiserte, dvs. etablert rundt funksjoner av typen salg, produksjon og økonomi. Oppmerksomheten er rettet mot de ulike aktivitetene som finner sted innenfor de ulike funksjonene. Målet har vært å skape funksjonell effektivitet, og ikke maksimere nytteverdien for kundene. Mellom funksjonene finnes det liten grad av kommunikasjon [5], og dette hindrer effektivitet og mulighet for å reagere på forandringer i markedssituasjonen. Som et resultat av dette må virksomheter endre de eksisterende prosessene. Et begrep for å betegne denne endringen av prosessene er Business Process Reengineering (BPR). BPR tar for seg endringer internt i organisasjonen [9].

BPR har som del av filosofien å oppnå en mer prosessorientert organisering. Dette skal bedre omstillingsevnen og legge grunnlaget for nyskapning og innovasjon. Konsekvensene ved en omstrukturering vil variere avhengig av virksomheten og oppgavene som berøres ved omleggingen. Prinsippene vil imidlertid i noe grad være de samme, og for å gi et praktisk eksempel taes det utgangspunkt i et paper som har beskrevet erfaringer ved en slik omlegging. Kim og Paik [6] har beskrevet hvordan de redesignet en prosess for å ansette medarbeidere i en virksomhet.

De så på prosessen fra et workflow-perspektiv, og skilte ut hvilke aktiviteter den bestod av, hvilke roller som var knyttet til de ulike aktivitetene og hvilke personer som fylte de ulike rollene. I følge det samme paperet var det estimert at 5.8 millioner virksomheter ville benytte workflow-teknologi i løpet av 1998, noe som tilsier at dette er et relevant eksempel.

IT ble benyttet som en muliggjører for å endre måten arbeidet ble utført på, både i kartleggingsprosessen og ved gjennomføring. Resultatet ble at prosessen ble mer strømlinjeformet, og man hadde bedre oversikt over de ulike aktivitetene og rollene. Bedret oversikt øker muligheten til hurtig å reagere på endrede krav i markedet. Med utgangspunkt i workflow-skjemaet de presenterer er det to hendelser som kan inntreffe, og som medfører en utfordring for kontekstbasert informasjonsbehandling.

To eller flere roller kan fylles av en person, eller flere personer kan fylle samme rolle over tid. Dette medfører et behov for at individet kan skille mellom tilgjengelig informasjon avhengig av hvilken arbeidskontekst man er i, og at ulike individer finner frem til og jobber med riktig informasjon når de skal fylle en bestemt rolle.

4.2.3 Business Network Redesign

BPR forutsetter at organisasjonens omgivelser er gitt eller ikke vil forandres i vesentlig grad [9]. Business Network Redesign (BNR) er en betegnelse på endringer v.h.a. IT i et forretningsnettverk hvor flere aktører utveksler informasjon.

BNR er en strategisk kobling mot ulike forretningspartnere for å skape avhengigheter mellom uavhengige organisasjoner, i den hensikt å oppnå konkurransefordeler.

Avgjørende for graden av suksess er virksomhetenes evne til å lage prosess-avhengigheter, forbedre beslutningstaking eller lage vesentlig verdiøkende tjenester.

Mulige forbedringsområder er:

- Transaksjonsprosessering. Mulighet for utveksling av data som bestillinger eller elektronisk betaling. EDI (Electronic Data Interchange) er i dag utbredt i mange organisasjoner.
- Lagerbehandling. Ved f. eks. bilproduksjon kan en underleverandør knytte seg opp mot bilprodusenten slik at denne har oversikt over tilgang på ferdige deler.
- Prosesskobling. Som det er beskrevet tidligere settes det i dag krav til masseskreddersøm. Et resultat av dette kan være at man outsourcer arbeid man ikke selv har spisskompetanse på, og integrerer dette i produksjons- eller utviklingsprosessen. F. eks. har Ford Motor Compant prosesskobling mot Goodyear Tire [9]. Begge firmaene har nytte av den andres spisskompetanse.
- Kunnskapsdeling. V.h.a. IT kan ulike organisasjoner koble seg opp mot hverandre og dele kunnskap. Mange mindre firmaer har i dag ekspertise innen et spesielt område, og kan tilby dette til andre som ikke har denne kunnskapen, eller gjensidig dra nytte av hverandres kunnskap.

Disse punktene medfører et behov for å kunne skille mellom informasjon tilhørende ulike eksterne aktører. De to første har effektiviserende fordeler av administrativ og operasjonell karakter.

Større utfordringer – og muligheter – finner man ved de to siste punktene. Prosesskoblinger mellom f. eks. utvikling og produksjon i to ulike firmaer setter store krav til informasjonsorganisering for å kunne jobbe med riktig informasjon avhengig av hvilken arbeidskontekst man er i. Kunnskapsdeling kan gi firmaer store konkurransemessige fordeler, men forutsetningen er at medarbeiderne i et firma finner og kan nyttiggjøre seg den eksterne kompetansen.

4.3 Concurrent Engineering

Concurrent Engineering (CE) er som BPR og BNR også et resultat av endrede krav i markedet. Ifølge Singh [10], kan tapet ved å innføre et produkt for sent i forhold til konkurrentene i et vekstmarked føre til opptil 1/3 tap av potensiell inntekt. Et produkt som blir sluppet 30% før konkurrentene kan imidlertid føre til 50% større inntjening i løpet av produktets livssyklus.

Concurrent Engineering bygger på mange gamle prinsipper, men navnet er bare noen få år gammelt. Problemet som er utgangspunktet for teorien er at mange av dagens organisasjoner er preget av hierarkisk struktur og et fokus på økonomisk fortjeneste. Dette har i mange tilfeller ført til at kompleksiteten og barrierene mellom gruppene p.g.a. tradisjon og intergruppekonkurranse har blitt for stor til å tilfredsstille dagens markedskrav [10].

En ytterlighet i denne problemstillingen er ulike grupper som gjør seg ferdig med sin jobb, sender den videre, og ved oppdagelse av eventuelle feil må en gruppe flere ledd tilbake i den serielle prosessen løse problemet. Deretter må man gå gjennom alle leddene igjen. Concurrent Engineering søker å løse dette problemet ved å tettere integrere ulike individer fra ulike fagfelt som deltar i livssyklusen til et produkt. Dette inkluderer utvikling, design, produksjon og støttefunksjoner.

For å gjennomføre dette kan man som i situasjonen som Singh [10] beskriver fra General Electric ha et felles informasjonssystem (IS) for alle de involverte gruppene og personene ved utvikling av et produkt. Dette skal integrere arbeidet mellom disse, og gi mer effektiv og sikrere utvikling og produksjon av produktet.

Singh [10] peker på at IT bedrer arbeidsmulighetene for grupper med medlemmer fra ulike fagdisipliner. Utover muligheten for å samarbeide og koordinere arbeid over avstander pekes det på mulighet for å dele informasjon, sikre kvalitet ved informasjonen som lagres samt mulighet for å lagre kunnskap som finnes i organisasjonen, f. eks. dokumentasjon av ferdige produkter eller hvilke erfaringer som er gjort ved et prosjekt.

Informasjonssystemet vil, fordi ulike personer og grupper fra forskjellige fagfelt skal benytte dette, ha kontekstbaserte utfordringer. Sentralt i Concurrent Engineering er som nevnt muligheten for utveksling av informasjon mellom ulike personer tilknyttet et produkt. Disse vil ha interesse for ulike deler av den informasjonen som finnes i informasjonssystemet. F. eks. vil en person som jobber med produksjon noen ganger ha bruk for produksjonsdata, og andre ganger kanskje ha et behov for å vite hvorfor ulike valg har blitt gjort under utviklingen. Derfor må brukerne kunne skille mellom den informasjon som finnes i informasjonssystemet avhengig av hvilken kontekst de arbeider i.

4.4 Knowledge Management

„Konseptet er basert på observasjonen om at relevant informasjon, intelligent og raskt kommunisert til riktig person kan gjøre forskjellen mellom det å gjøre svært gode eller dårlige avgjørelser.“ [1] Information Management kan forenklet forklares som en strukturert organisering av predefinerte data. Knowledge Management (KM) søker i tillegg å koble strukturert og ustrukturert informasjon til regler som hele tiden blir forandret av de som bruker dem.

Lagret informasjon vil ikke i seg selv alltid føre til mer kunnskap. Hvis informasjonsmengden blir for stor, vil brukeren oppleve „information overload“. Dette kan forklares med at brukeren får presentert så mye informasjon at denne ikke klarer å skille ut det som er av betydning for oppgaven vedkommende skal løse. Dette kan f. eks. oppstå ved søk i en stor dokumentdatabase eller etter bestemte e-mail blant svært mange.

Et Knowledge Management system har som mål å ha et sømløst grensesnitt mot ulike informasjonsdomener, og å kunne tilby brukeren den informasjonen som er nødvendig for å løse en bestemt oppgave. All kunnskap i organisasjonen skal lagres i dette systemet.

Det finnes forskjellige typer kunnskap. Kunnskap om f. eks. produkter og økonomi, og kunnskap som er et resultat av erfaringer. Det siste er vanskelig å få tak i, fordi den befinner seg inne i hodene til de ansatte, og er grunnlag for intuisjon og evne til å lære seg nye ting.

Enkelte mener at denne kunnskapen blir en svært viktig konkurransefaktor fremover [7]. Derfor er det viktig at andre kunnskapsarbeidere får tilgang til denne kunnskapen. Hvis ikke kunnskapen kan lagres, f. eks. som en rapport, kan man lage et Knowledge Management system som gir mulighet for at folk hurtig finner frem til tilgjengelige kunnskapspersoner basert på kunnskap og fagfelt. Dette kan i dag ofte være en tidkrevende prosess.

Et „whitepaper“ om Knowledge Management [11] trekker spesielt frem at et søk i et Knowledge Management system skal gi et resultat basert på kontekst og en forståelse for bakgrunnen for søket. Menneskelig interaksjon er innfallsvinkelen ved innsamling, distribusjon og gjenbruk av informasjon. Kunnskap må kunne katalogiseres etter behov og ikke på forhånd [7], fordi kunnskap er foranderlig og kan derfor ikke for evig plasseres i en kategori. Dette har som konsekvens at informasjonen må lagres og organiseres på en slik måte at den kan gjenfinnes v.h.a. parametre som er karakteristiske for den konteksten søket utføres i.

Teorien har fellestrekk med andre teorier, som Information Management, Document Management (DM) og tildels Concurrent Engineering, workflow og teorier relatert til endringsprosesser. Noen av disse teoriene kunne også vært trukket frem, men Knowledge Management bygger på disse, og har i tillegg et mer kontekstbasert fokus som det påpekes ovenfor.

Knowledge Management bygger f. eks. på Document Management og workflow ved at DM blir sett på som substansen for en virksomhets KM-infrastruktur. DM blir som del av et KM-system sett på som den delen som muliggjør søk etter og distribusjon av informasjon. Workflow software kan benyttes for å levere informasjonen til riktig person til rett tid. Når det i denne oppgaven referers til Knowledge Management, vil det derfor implisitt også bli referert til disse andre teoriene.

4.5 Kontekstbasert design

Et problem med applikasjoner kan være at de ikke tilfredsstiller brukernes behov. Dette kan føre til at disse ikke tar i bruk teknologien [20]. Kontekstbasert design er en betegnelse på metodikk som har til hensikt å kartlegge brukernes arbeidssituasjon, for dermed å lage applikasjoner som tilfredsstiller deres behov og krav.

4.5.1 Et praktisk eksempel

Endsley og Robertson [14] gir et eksempel på hvordan en kontekstanalyse kan benyttes. De så på hvilken betydning total oversikt over arbeidet som skulle utføres hadde for den

enkelte ved vedlikehold av fly. De fant at mange feil skyldtes mangelfull oversikt over hva som skulle gjøres eller hadde blitt gjort.

De utviklet derfor en kontekstanalyse for å finne ut hvordan oppgaver ble utført, og i hvilken sammenheng. De fant flere problemer og barrierer for å oppnå situasjonsoversikt på gruppenivå. Noen av disse var statusinformasjon, mangel på teamarbeid og informasjonsdeling, forandring av prosedyrer og behov for å følge deler v.h.a. et datasystem. Dette er kontekstbaserte utfordringer fordi ulike personer kan som vi har sett ved BPR inneha ulike roller eller ulike roller kan fylles av ulike personer over tid. Problemet i dette eksempelet var da også at ulike personer ikke fikk tak i riktig informasjon fordi rollene var uklare, og dermed var heller ikke systemet designet for å støtte de ulike rollene med riktig informasjon.

4.5.2 Kontekstanalyse

Beyer og Holtzblatt [12,13,14] presenterer en metode for å kartlegge ulike arbeidskontekster hos brukerne ved utvikling av programvare. Utviklerne kan dermed lage en kontekstmodell [15] for å bestemme hvilke faktorer som virker inn på arbeidet som utføres.

De kaller sin metode for „Contextual Inquiry“, og dette kan oversettes med kontekstbasert undersøkelse eller forskning. Denne metoden går ut på å få tak i data om brukeren mens de jobber med virkelige oppgaver i sitt arbeidsmiljø [14]. Dermed kan utviklingsteamet få en bedre forståelse for behovet hos brukerne.

Gjennom sitt arbeid har Beyer og Holtzblatt [12] funnet at ofte er arbeidsoppgavene preget av rutine, og derfor er man ikke klar over hvorfor og hvordan man egentlig gjør ting. Metodikken gir forståelsen for ulike arbeidskontekster ved at den ved analyse får frem alle detaljer og motivasjoner som implisitt finnes ved folks arbeidsmåter.

Metodikken har også som mål at data samlet fra enkeltpersoner skal danne grunnlag for å finne et felles mønster og struktur, men uten å miste individuell variasjon. Dette gir mulighet for et system som støtter kontekstbasert arbeide.

4.6 Human-Computer Interaction

Human-Computer Interaction (HCI) er studiet av hvordan mennesker designer, implementerer og bruker interaktive datasystemer samt hvordan bruk av datamaskiner påvirker individer, organisasjoner og samfunnet. Det fokuseres ikke bare på brukervennlighet, men også på nye interaksjonsmetoder for støtte av brukeraktiviteter, bedret tilgang til informasjon og å skape bedre former for kommunikasjon. Inkludert er også input-/ output-utstyr og interaksjonsteknikkene som hører sammen med disse, hvordan informasjon er presentert og søkt etter samt problemstillinger relatert til utvikling av systemene [17].

Et relevant tema innen HCI m.t.p. denne oppgaven, er universell aksess til store og komplekse distribuerte informasjonslagre. De potensielle brukerne av databaser og andre informasjons-systemer blir stadig flere og samtidig mindre teknisk skolerte. Derfor har de ikke kunnskap om teknologien bak f. eks. søk i en database. Dette medfører et problem ved interaksjon med et informasjonssystem (IS).

Grensesnittet for informasjonsaksess må tilby større fleksibilitet for hvordan søkestrenger og data visualiseres. Man må kunne tilby interaksjon mellom bruker og database utover to-trinnssøk (spesifiser/vis resultat), som f. eks. databrowsing, filtrering og dynamisk og inkrementerende søk. Ved å filtrere informasjon og ha mulighet for å foreta dynamiske søk, vil støtte for kontekstavhengige søk oppnåes. Fundamentale problemstillinger ved forskning innen HCI omhandler kontekst-basert interaksjon med et IS, og inkluderer hvordan man best kan organisere oppgaver mellom mennesker og datamaskiner, lage systemer som kan adapteres til ulike brukere og støtte endringer i konteksten ved bytte av oppgaver [17].

Bryson et al [17] viser til at empiriske studier har vist at riktig bruk av HCI-teknikker ved applikasjonsutvikling kan medføre betydelig gevinst p.g.a. mer effektiv gjennomføring av arbeidsoppgaver for brukerne, færre feil, stor reduksjon av usikkerhet ved bruk av systemet, redusert press på støttepersonell, eliminasjon av trening og unngåelse av endringer i software etter utgivelse.

5 Posisjonering

Relatert forskningsarbeid og fagområder som er gjennomgått i kapittel 4 har i varierende grad behandlet kontekstbaserte problemstillinger eller beskrevet hvordan kontekstbasert arbeids- og informasjonsorganisering bør være. I dette kapitlet vil det bli diskutert hvordan materialet som er gjennomgått kan belyse problemstillingen i denne oppgaven. Det vil bli foretatt en posisjonering m.t.p. hva som legges i begrepet kontekstbasert arbeids- og informasjonsorganisering i denne oppgaven i forhold til dette. Nedenfor er det relaterte forskningsarbeidet og fagområdene diskutert etter ulike innfallsvinkler til problemstillingen.

5.1 Parametre som bestemmer kontekst

Innen CSCW blir det påpekt at noe av grunnlaget for definisjon av gruppevare finnes ved å se på i hvilke grad ulike roller eller kommunikasjonsbehov i grupper blir støttet. Komplekse bruksmønstre må støttes, og det må taes høyde for at disse endres fra prosjekt til prosjekt. Dette er i samsvar med utgangspunktet for denne oppgaven. CSCW påpeker derfor som denne oppgaven at skal et informasjonssystem være mest mulig effektivt, må ulike arbeidskontekster gjenkjennes.

BPR og BNR behandler også ulike roller og aktører. Sentralt i disse teoriene er at det fokuseres på behovet for å kunne knytte disse ulike rollene og aktørene opp mot hverandre v.h.a. IT for bedre å utnytte muligheter. Teoriene behandler i stor grad hvordan selve arbeidet skal legges opp, og dette er tidligere definert til å ligge utenfor denne oppgaven. Sentralt er imidlertid den kunnskapen disse teoriene gir om hvorfor ulike roller oppstår og hvorfor ulike roller og aktører bør knyttes opp mot hverandre. Dette gir kunnskap om hva som kjennetegner ulike roller og aktører, og er dermed et viktig bidrag for å finne parametre for å bestemme disse.

Informasjonen som benyttes i forhold til ulike aktører eller i forbindelse med ulike roller må nemlig kunne kategoriseres slik at den kan kobles mot disse eller brukes i sammenheng med eventuelle nye roller eller aktører som måtte oppstå som følge av et nytt BPR- eller BNR-prosjekt eller endringer i markedet. Teoriene bidrar i noe grad til forståelse for betydningen av kontekstbasert arbeids- og informasjonsorganisering ved at de tar for seg strategiske fordeler for virksomheter ved å benytte IT på denne måten.

Concurrent Engineering behandler delvis kontekstbasert arbeids- og informasjons-organisering på prosjektnivå. Det påpekes at man raskt må kunne finne ønsket informasjon i et prosjekt avhengig av det arbeidet som skal utføres. Dette er i samsvar med utgangspunktet for denne oppgaven. Det hverken CSCW, BPR, BNR eller CE gjør, er å gi en løsning på hvordan ulike kontekster skal finnes. At det er et behov for å finne disse gir seg selv som et resultat av intensjonene til de ulike teoriene. Uten god kontekst-basert arbeids- og informasjonsorganisering vil det ikke være mulig å finne frem til riktige aktører eller informasjon avhengig av hvilken kontekst man jobber i.

Med utgangspunktet for oppgaven og på bakgrunn av de teoriene som er behandlet er det mulig å gi følgende definisjon på i hvilke situasjoner det er behov for kontekstbasert arbeids- og informasjonsorganisering:

Det er et behov for kontekstbasert arbeids- og informasjonsorganisering i situasjoner hvor brukeren forholder seg til ulike personer eller ulike eksterne aktører avhengig av hvilken aktivitet som utføres, eller der behovet for informasjon er bestemt av hvilken aktivitet som utføres.

Med aktivitet menes her arbeid for å utføre en bestemt type oppgave.

5.2 Krav i et sosio-teknisk perspektiv

CSCW og HCI behandler begge sosio-tekniske problemstillinger. Innen CSCW pekes det på – som også utgangspunktet for denne oppgaven er – at noe av problemet ved bruk av IT-verktøy er at det ikke er kompabilitet mellom ulike applikasjoner og over ulike informasjonsdomener. Det pekes også på ulike utfordringer ved bruk av IT-verktøy, mer eller mindre som et resultat av dette. Dette er nærmere beskrevet i kapittel 4.1.3.

Disse utfordringene er i høy grad sammenfallende med hva denne oppgaven søker å løse. Disse funnene gir derfor – sammen med funn fra andre teorier – grunnlag for å sette opp hvilke krav et informasjonssystem som skal støtte ulike arbeidskontekster må tilfredsstille. CSCW er sammenfallende med den delen av denne oppgaven som omhandler hvordan brukeren skal få tilgang til informasjon fra ulike kilder og hvordan denne informasjonen skal organiseres for best mulig å kunne utnyttes i ulike situasjoner.

Mens CSCW har et fokus på IT-støttet samarbeid, fokuserer HCI på interaksjonen mellom bruker og IT. Som det er påpekt i kapittel 4.6, behandler HCI bl. a. visualiseringen av data. Forskere innen dette fagområdet søker bl. a. å redusere usikkerheten og effektivisere arbeidet m.t.p. brukerne. Dette har sammenfallende problemstillinger med organiseringen av det virtuelle arbeidsmiljøet og hvordan brukeren skal kunne jobbe mot informasjon avhengig av ulike arbeidskontekster. Hvordan sikre kvalitet og samtidig minimalisere innsats for å spesifisere metainformasjonen er derfor teoretisk et HCI-problem.

Diskusjonen i dette kapitlet viser at CSCW og HCI har sammenfallende problemstillinger med kontekstbasert arbeids- og informasjonsorgansiering spesielt i den delen som omhandler hvordan det virtuelle arbeidsmiljøet skal organsiseres og hvordan brukeren skal kunne fortelle systemet hva slags informasjon brukeren ønsker.

5.3 Metoder for å bestemme kontekst

"Contextual Inquiry" er en metode som er trukket frem i kapittel 4 fordi den har til hensikt å finne hvilke ulike kontekster brukeren befinner seg i ved utførelse av arbeid. Denne metodikken tar for seg en problemstilling som også må besvares ved kontekst-basert arbeids- og informasjonsorganisering. Av denne metodikken går det frem at for å

få tak i de ulike arbeidskontekstene må man få frem alle detaljer ved arbeidet som utføres; både hvorfor det utføres (ulike roller gir ulik motivasjon) og hvilke ressurser, roller eller aktører brukeren forholder seg til. Metodikken har relevans for den delen av oppgaven som går ut på å identifisere ulike parametre for å bestemme ulike kontekster. "Contextual Inquiry" er derfor en metodikk som kan gi ideer til hvordan det skal kunne defineres ulike arbeidskontekster.

5.4 Informasjonsorganisering

Den teorien som er mest sammenfallende med problemstillingen kontekstbasert arbeids- og informasjonsorganisering er Knowledge Management. Som det påpekes i kapittel 4.4 er også Information Management og Document Management relaterte teorier, men fordi KM defineres til å dekke begge disse, refereres det kun til KM.

Knowledge Management påpeker også behovet for å presentere informasjon basert på kontekst for å unngå "information overload". Det påpekes også at dette kan være avgjørende for kvaliteten på de belutningene som taes. Teorien har ikke bare fellestrekk med denne oppgaven m.t.p. hvordan informasjon skal organiseres, men også hvorfor det er et behov for å organisere informasjon avhengig av ulike kontekster. Derfor er den viktig som kilde til ytterligere forståelse av problemstillingen og hvilke muligheter og utfordringer som finnes ved kontekstbasert arbeids- og informasjonsorganisering.

En svakhet i forhold til problemstillingen i denne oppgaven er at det ikke blir spesifisert hvordan ulike kontekster skal finnes. Av det materialet som er gjennomgått kan det virke som om det fokuseres på en hierarkisk lagring av informasjon, og at måten å finne dette avhengig av ulike kontekster er å søke på ulike emner eller andre parametre. I det siste har imidlertid forskning innen KM konsentrert seg mer om bruk av metainformasjon. Basert på diskusjonen i dette kapitlet er det rimelig å anta at denne oppgaven vil kunne bidra med kunnskap som kan komme et Knowlegde Management system til nytte.

5.5 Posisjonering av teorier

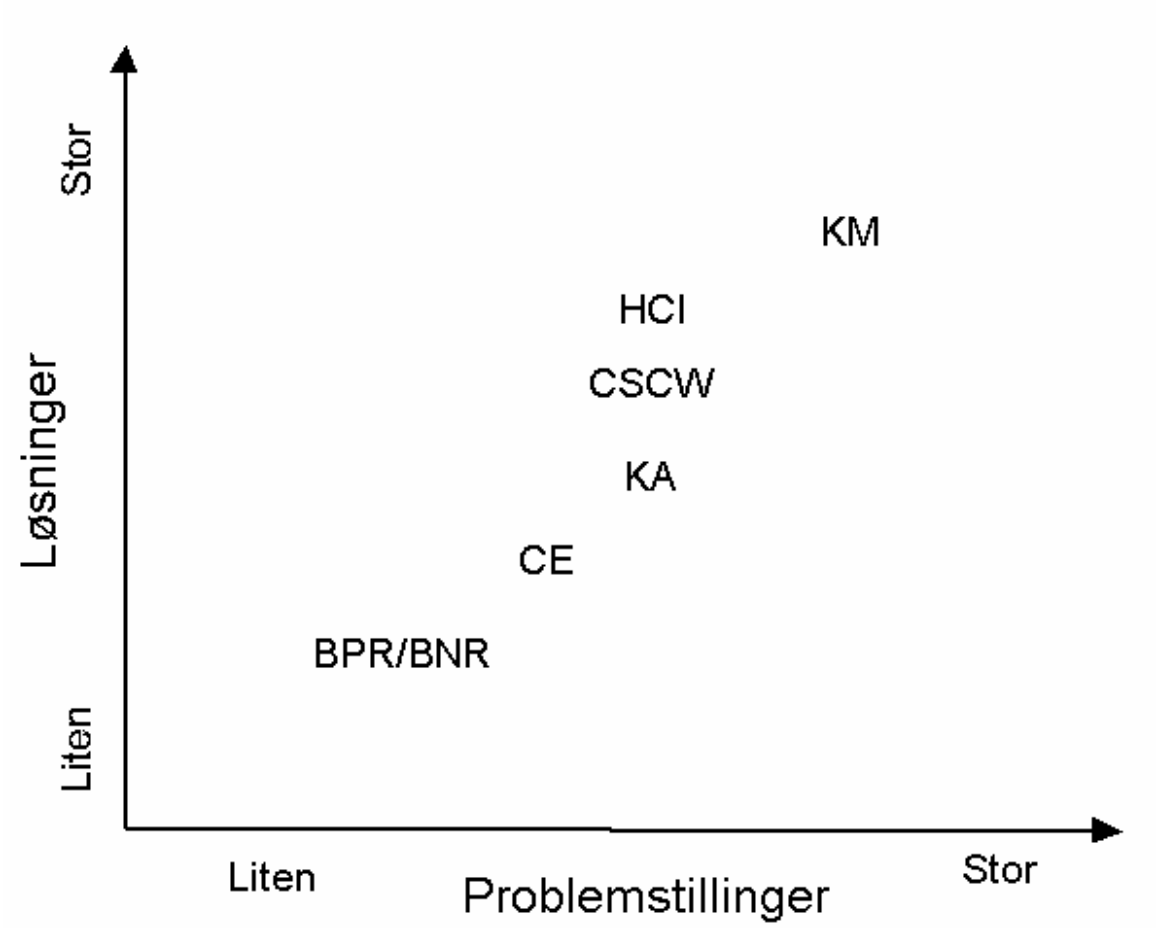
Som det er vist i dette kapitlet vil de ulike fagområdene og forskningsarbeidet som er gjennomgått i varierende grad ha betydning for kontekstbasert arbeids- og informasjons-organisering. Noen teorier som BPR og BNR behandlet i liten grad kontekstbaserte problemstillinger, og hadde heller ingen løsninger av betydning for hvordan kontekstbasert arbeids- og informasjonsorganisering bør være.

Concurrent Engineering behandlet i noe større grad disse temaene. Behovet for at et IS må støtte ulike kontekster ble i større grad pekt på som en utfordring, men hvordan informasjon skulle aksesseres avhengig av ulike kontekster ble ikke behandlet spesielt utover en informasjonsorganisering basert på en hierarkisk oppdeling med felles tilgang for alle de involverte gruppene og personene ved utvikling av et produkt.

Kontekstanalyse, CSCW og HCI behandler i enda større grad kontekstbaserte problemstillinger, og har dette som utgangspunkt eller sentral del av teorien og

forskningen. Disse teoriene kommer også i noe varierende, men i større enn de andre teoriene som er nevnt ovenfor med forslag til hvordan kontekstbasert arbeids og informasjonsorganisering bør være. Den teoretiske retningen som skiller seg ut ved stor grad av behandling av kontekstbaserte problemstillinger og beskrivelse av løsninger for kontekstbasert arbeids- og informasjonsorganisering er Knowledge Management.

Av det som er gjennomgått hittil i kapitlet forståes det at de ulike teoriene har noen fellestrekk ved problemstillingen i denne oppgaven, men i varierende grad. Nedenfor er det vist en figur hvor disse ulike teoriene er innbyrdes rangert etter i hvilken grad de behandler problemstillinger som sammenfaller med denne oppgaven. Langs x- aksen i hvilken grad det behandles kontekstbaserte problemstillinger, og langs y-aksen i hvilken grad det beskrives løsninger for kontekstbasert arbeids- og informasjons-organisering. Hensikten med denne oppgaven er ikke å veie ulike teorier i forhold til disse to aksene, men å posisjonere de i forhold til kontekst-basert arbeids- og informasjonsorganisering. Teorier som ligger svært nær hverandre i forhold til en av aksene er derfor plassert likt. Figuren må derfor sees som en skissering.



Figur 5.1 Posisjonering av teorier i forhold til oppgaven

5.6 Krav til informasjonssystemer

På bakgrunn av utgangspunktet for oppgaven og basert på de ulike teoriene som er behandlet, er det kommet frem til hvilke krav som må tilfredsstilles for at et informasjonssystem skal støtte kontekst-basert arbeids- og informasjonsorganisering. Disse kravene vil det måtte taes hensyn til i det videre arbeidet med oppgaven. Kravene kan sammenfattes i følgende punkter:

- Det må finnes støtte for ulike arbeidskontekster for å gi brukertilfredshet
- Det må være adaptivt m.t.p. eventuelle nye kontekster og bruksmønstre
- Det må støtte komplekse bruksmønstre
- Det må tilby rask tilgang til informasjon
- Det må tilby relevant informasjon av høy kvalitet

6 Bruk av metainformasjon

Metainformasjon var gitt i oppgaven som den teknologien som gjorde det mulig med bedre mekanismer for lagring og gjenfinning av informasjon. I dette kapitlet vil meta-informasjon og metadata presenteres og defineres. Videre ble det gjort et søk etter forskning innen dette fagfeltet for å lære av andres arbeid og ta del i deres erfaringer. Det forskningsarbeidet som var mest relevant blir presentert. Til slutt i dette kapitlet er tidligere forskningsarbeid benyttet som grunnlag for å sette opp hvilke muligheter bruk av metainformasjon gir, hvilke praktiske utfordringer som finnes og hvordan metadata kan organiseres.

Metainformasjon og bruk av metadata var et relativt nytt fagområde for meg, og derfor var det nødvendig med søk etter informasjon for å sette seg inn i teknologien. Fordi jeg ikke hadde erfaring med bruk av metadata var det også av denne grunn nødvendig å søke etter forskning som andre hadde gjort for å lære av deres erfaringer.

6.1 Metainformasjon og metadata

Overgangen fra analoge til digitale lagringsmedier har ført til at individet får tilgang til svært store mengder informasjon ved søk i ulike databaser, f. eks. på et intranett. Dette kan som vi har sett i kapittel 3 gi store fordeler, men også være en ulempe ved at individet opplever „information overload“ fordi man ikke klarer å skille ut relevant informasjon.

Grunnen til dette er at brukeren ikke vet nok om informasjon som blir presentert til å skille ut hva som er relevant, eller at man ikke finner relevant informasjon fordi denne ikke er organisert på en slik måte at den enkelt kan aksesseres med utgangspunkt i den arbeidskonteksten man for øyeblikket er i. I kapittel 5.3 vil erfaringer med aksessering av ulike typer informasjon bli nærmere beskrevet.

For at ulike arbeidskontekster skal kunne støttes, må informasjonen som vi har sett være organisert på en slik måte at den kan aksesseres avhengig av konteksten. Dette kan gjøres ved å lagre informasjonen hierarkisk, slik som f. eks. i et bibliotek, eller ved bruk av ulik tilleggsinformasjon. Denne tilleggsinformasjonen – eller metainformasjonen – brukes for å klassifisere, legge til attributter, beskrive, navngi og registrere dataelementer [26].

Metainformasjon representeres i informasjonsdomener som metadata. Metadata er i motsetning til kontekst-basert arbeids- og informasjonsorganisering klart definert. Newton [26] har summert opp hvordan metadata blir beskrevet i standardiserings-organisasjonen ISO. Nedenfor vil begrepet metadata bli utdypet med ISO-standarden som utgangspunkt.

6.2 Metadata

ISO/IEC JTC 1 Subkomité 14, Data Element Prinsipper, har laget en standard med seks deler som tar for seg standardisering av metadata. Spesifikasjon og Standardisering av

Dataelementer (ISO 11179) beskriver regler, prinsipper og retningslinjer for klassifisering, setting av attributter, beskrivelse, navngiving og registrering av dataelementer [26].

ISO 11179 sier at metadata er „informasjon og dokumentasjon som gjør datasett forståelige og muliggjør deling med brukere“. Ved søk i metadata vil brukere bli i stand til å forstå og lokalisere dataene de trenger. Dette fører til eliminasjon av redundant datainnsamling og prosessering [26].

Av dette forstår vi at metadata er en velegnet metode for å beskrive informasjon slik at ulike kontekster ved arbeide støttes. For å få en enda bedre forståelse for hva som legges i begrepet metadata vil det nedenfor bli gitt en kort forklaring på hvordan dataelementer spesifiseres og standariseres i ISO 11179. Standarden tar for seg seks ulike punkter: klassifisering, attributter, definisjon, navngiving og identifisering samt registrering av dataelementer.

6.2.1 Spesifikasjon og standardisering av dataelementer

Standard dataelementer er basert på et sett av komponenter som er avledet av konsepter om datamodeller. Disse komponentene er: objekt-klasser, properties og representasjon. Objekt-klassen representerer tingen fra den virkelige verden som dataelementet skal beskrive. Et sett av properties beskriver alle karakteristika ved tingen som er av interesse for og prosesserbart for informasjonsteknologi. Representasjonen beskriver hvordan ulike properties er lagret i informasjonssystemer.

Det er derfor mulig å ha mange representasjoner av et dataelement, men betydningen av informasjonen og innholdet i dataelementet kan være det samme. Som et eksempel kan vi tenke oss objektet Dokument som har en property Id_Forfatter. Ulike representasjoner av property‘en som f. eks. Ola, Kari, Joe eller et tall vil ikke endre oppbygningen av objektet. Representasjoner kan derfor brukes for å beskrive et objekt uten å endre innhold eller kvalitet. Denne beskrivelsen av et objekts innhold kalles metainformasjon.

6.2.2 Klassifisering

Dataelementer bygges opp v.h.a. de ulike komponentene, og disse komponentene må defineres med både innhold og navn. For å sikre et sett av diskrete og komplette dataelementer må komponentene bli definert på en metodisk måte, og ikke ved tilfeldigheter. Ved å gjøre dette har man, fordi komponentenes form er konsistent, mulighet for å lage relasjoner mellom dataelementene.

I praksis bør komponentene ideelt sett defineres ved at man velger blant et forhåndsdefinert sett av dataelementer. ISO har klassifisert flere slike etter ulike kategorier, og det foregår et kontinuerlig arbeide for å definere nye sett av dataelementer for ulike kategorier [26]. Ved en slik metodikk sikrer man at det ikke blir et utall av forskjellige typer dataelementer, og dermed har man mulighet for mer effektiv søk etter data.

6.2.3 Attributter

Dataelementer må i tillegg til å klassifiseres bli beskrevet på en standardisert måte som sikrer tilgang til dataene. Standardisering av attributter inkluderer spesifikasjon av et standard sett av attributter med bestemte tillatte verdier. Denne spesifikasjonen skal ikke være avhengig av f. eks. en bestemt applikasjon.

For alle dataelementer er det definert noen basisattributter. ISO 11179, del 4, definerer i tillegg andre attributter for å utvide mulighetene for klassifisering, navngiving, identifikasjon og registrering [26], men disse vil ikke bli behandlet her. Nedenfor vil de fem basiskategoriene av attributter bli beskrevet for å gi et innblikk i hvordan metadata kan settes.

Identifisering - Navn, kontekst, identifisering, versjon, registrerings autoritet, synonymt navn
Definering - Definisjon
Relasjoner - Klassifiseringsskjema, nøkkelord, relaterte datareferanser, type av relasjon
Representasjon - Representasjonskategori, form for representasjon, datatype for dataelementenes verdi, maksimum størrelse for dataelementverdi, minimum størrelse for dataelementverdi, layout for representasjon, tillatte dataelementverdier
Administrativt - Ansvarlig organisasjon, registreringsstatus, submitting organisasjon, kommentarer

Av dette ser vi at metadata kan gi mye informasjon om et dataelement. Hvis vi tenker oss at dataelementet er et dokument lagt i en database, gir bare disse attributtene gode muligheter for søk etter bestemt informasjon, som igjen gir støtte for ulike arbeidskontekster.

6.2.4 Definisjon av data

Regler for datadefinisjon må bli spesifisert av standarder for å oppnå konsistens i representasjonen. Siden definisjonen av en dataentitet som oftest er uttrykt som tekst i et naturlig språk (f. eks. engelsk), må ikke en standard for datadefinisjon bare bestå av regler (som kan testes mot en bestemt kilde), men også retningslinjer (av mer subjektiv art) for denne teksten.

ISO 11179, del 4, definerer regler og retningslinjer for datadefinisjon. Dette er gjengitt nedenfor, etter Newton [26].

En datadefinisjon skal:

- være unik innen de registerne den forekommer
- være definert som den eneste
- definere hva konseptet er, ikke bare hva det ikke er

- være en beskrivende frase eller setning
- inneholde kun allment forståtte forkortelser
- bli uttrykt uten innebygde definisjoner av andre termer

En datadefinisjon bør:

- uttrykke den essensielle meningen med konseptet
- være presis og utvetydig
- være konsis
- kunne stå alene
- bli uttrykt uten å inkludere rasjonell funksjonell bruk, domene informasjon eller prosedyre informasjon
- unngå sirkulær resonnering
- bruke samme terminologi og konsistent logisk struktur for relaterte definisjoner

Dette gir viktige prinsipper og retningslinjer ved utforming av egne definisjoner for dataentiteter.

6.2.5 Navngiving og identifiseringsprinsipper

Identifisering kan bli gjort på flere måter: uintelligente identifisering som f. eks. nummer (som ikke gir noen spesiell mening til mennesker) og intelligent identifisering som bruk av grafiske symboler eller navn med en bestemt betydning. ISO 11179, del 5, beskriver måter å identifisere dataelemeneter på, både intelligente og uintelligente [26]. For å unngå tvetydighet for alle dataelementer innen ulike registere, presenteres en unik identifiserer, IRDI (International Registration Data Identifier).

Prinsipper for navngiving av dataelementer blir også diskutert. Navn må være klare, korte, sammenfallende med gjeldende regler, uten fysisk kontekst og skal ha sammenheng med definisjonen av dataene. En god konvensjon for navngiving kan avsløre elementer der data har blitt sammenblandet med metadata. Dette kan forekomme hvis man har en lang rekke med dataelementer, og disse kun variere ved hva de beskriver [26].

6.2.6 Registrering av dataelementer

Mekanismen for deling av data skjer ved å ha et register for standard data-elementer. Flere registre blir vedlikeholdt for ulike temadomener med logiske og funksjonellt relaterte data. Disse registrene er lett aksesserbare, og gir mulighet for søk etter mulige kandidater for gjenbruk av data. Registreringsprosedyren for internasjonale standard dataelementer er definert i ISO 11179, del 6 [26].

6.3 Erfaringer med metadata fra ulik forskning

For å få en bedre forståelse for bruken av metadata er det hensiktsmessig å se på hvordan de er brukt i andre sammenhenger. Ved å gjøre dette er det mulig å lære av andres erfaringer, og man kan bruke deres metodikk hvis den er hensiktsmessig.

Søk etter relevant forskning har blitt begrenset til noen utvalgte områder hvor man har behov for og har jobbet en del med metadata, og som ansees som relevant for oppgaven. Disse områdene er digitale biblioteker (herunder dokumenter), multimedia (bilder, video og audio) og data som brukes av forskere da disse ofte bruker data fra svært store databaser.

Utfordringen er generelt å lokalisere filer som inneholder relevant data og søking etter små biter av data blant en stor mengde, slik som f. eks. hos Keller og Jones [28]. Nedenfor presenteres kort disse problemstillingene, hvilke erfaringer man har fått ved arbeide med disse og hvilke muligheter som finnes ved bruk av metadata.

6.3.1 Multimedia

Multimediadata (video, audio og bilder) blir brukt i mange vitenskapelige forskningsområder som medisin, geologi, klimatologi og kartografi, og i underholdningsøyemed. For å bli et nyttig medium for forskning må det være mulig å søke i og indeksere multimedia som ved alfa-numerisk informasjon. Pikselverdier har liten verdi i seg selv, og derfor er det et behov for å indeksere betydningen av ulike piksler. Dette kan f. eks. være om det er en bil i et bilde eller lyden av en løve i et lydklipp. Denne informasjone kalles metainformasjon [31].

Multimediadata har i de fleste tilfeller to typer metadata som kan bli indeksert. Dette er registreringsinformasjon, som f. eks. hvem som tok et bilde, når en video ble spilt inn eller kamerasettinger, eller semantisk informasjon som må bli lagt til manuelt (menneskelig eksaminasjon av bildet) eller automatisk (visuell prosessering) [31].

Griffioen et al [31] har utviklet en metode for automatisk identifisering av semantisk informasjon i et bilde. Dette har de gjort fordi mengden av informasjon er så stor at manuelle metoder ikke er tilstrekkelige. Spesielt for deres modell, MOODS, er at de kobler metadata fra f. eks. et bilde hvor det er identifisert et menneske opp mot annen metadata for om mulig å ytterligere spesifisere funnet.

Det er mulig for brukere å velge verdier på ulike parametre for bedre å støtte brukerens forventninger til hva systemet skal returnere ved søk. Dette kan sees som en mulighet for å støtte ulike kontekster. Ved å velge parametre vil systemet presentere relevante metadata, og unnlate å presentere metadata uten verdi. Identifiseringen av objekter gjøres v.h.a. regler, og et slik eksempel kan være hund←fire_ben&&hale&&hår&&høyde>1. Dette sier at hvis et objekt har blitt identifisert med fire ben, hale og hår, og høyden av objektet er større enn 1, så kan vi konkludere med at vi har funnet en hund. Dette er en metodikk som kan overføres til andre situasjoner.

6.3.2 Digitale biblioteker

Tradisjonelle biblioteker har kategorisert informasjonen innen ulike kategorier. Informasjonen finnes i hovedsak i form av trykte medier, og er som oftest lokalisert innen samme geografiske område. Digitale biblioteker har i tillegg til tekstinformasjon et stort antall multimediaobjekter lagret (bilder, video og audio). Disse kan igjen være lagret på ulike databaser spredt over hele verden. Dette skaper i seg selv et problem ved at ulike databaser gir semantiske problemer p.g.a. ulike lagringsmetoder. Metadata brukes både til å overkomme slike problemer [21,27,37] og til å kategorisere data slik at man kan søke i digitale biblioteker minst like godt og ved hjep av IT-baserte søkemekanismer bedre enn i tradisjonelle biblioteker [22,23,24,25].

For å oppnå bedre muligheter for søk utover fulltekstsøk, må dokumentsemantikk være et søkekriterie [25]. Ved å beskrive et dokument med andre ord enn de som finnes i dokumentet (kalt Abstracting & Indexing (AI)), kan semantisk innhold indekseres. Jo bedre indekseringen er, jo bedre er muligheten for å finne ønsket informasjon. En A&I record kalles metadata. Fordi nye dokumenter stadig legges til eller endres, må indekseringen automatiseres for konstant å være oppdatert [25].

Wilensky [23] fokuserer på hvordan digitale informasjonstjenester kan støtte opp om arbeidet til brukerne. Han peker på tre krav slike brukere har til et informasjonssystem:

- Brukerne bruker systemet ofte, og siden informasjonen kan være i flere dokumenter, må informasjonssystemet ha komplekse og kraftige analyse- og ekstraheringsmetoder for heterogene objekter.
- En arbeidsgruppe må kunne aksessere sin egen samling av dokumenter i tillegg til ekstern informasjon. Disse brukerne lager også kontinuerlig nytt materiale som også aksesseres av eksterne, og dette krever fleksible metoder for editering, strukturering og leveringsmekanismer.
- Informasjonssystemet må kunne integreres med gruppens arbeidspraksis og kunne brukes sammen med andre systemer.

Konteksten for det arbeidet systemet han beskriver skal støtte opp om er brukerspesifisert, og søket foregår på Internett. Systemet er konstruert på en slik måte at det prøver å finne ut hva brukeren er interessert i, og ut ifra dette presentere informasjon som er relevant. Dette kan skje ved at brukeren spesifiserer ulike parametre, eller jobber spesielt med et semantisk merket område på et bilde. Resultatet av søket eller identifiseringen av arbeidskonteksten kan da presenteres som en side med dynamisk genererte linker tilpasset brukerens arbeidskontekst.

6.3.3 Metereologi

Bruk av metadata kan lette aksess til databaser inneholdende meteorologisk informasjon [32,33]. Emanuelson og Hedlund [32] beskriver et system fra det svenske meterologi og hydrologi instituttet. De benytter ulike metadatakategorier knyttet til meteorologi som tid, geografisk område, meterologiske parametre (f. eks. temperatur), kvalitet, kilde og format (f. eks. ren tekst eller GIF).

De presenterer en konseptmodell hvor hver meteorologisk verdi som f. eks. temperatur blir knyttet til ulike metadata som tid, kilde, kvalitet etc. De ulike metadataene er delt inn i ulike kategorier, og dette letter informasjonssøk, bl .a. ved at det er ulike søkemønstre innen hver kategori. Det er ikke nødvendig å oppgi søkeparametre i alle kategorier for å foreta søk.

Metadataene er lagret for seg selv, og inneholder en referanse til hvor informasjonen finnes. Metadataene tilbyr sammen med programvare mulighet for å finne data, sammenligne ulike property'er for ulike sett av data, finne overlapp mellom sett av data, finne duplikeringer av data og aksessering av data. Dette gir flere bruksområder, og de peker på tre som kan være interessante som ideer også til problemstillingen i denne oppgaven.

Et bruksområde kan være at en kunde ønsker meteorologistatistikker som ennå ikke finnes. Tilbyderen ønsker da å finne ut om og eventuelt hvordan disse kan bli levert. Man kan enten se på basisdata, eller lete etter tilsvarende produkter. Problemet er da å finne passende eksisterende produkter og identifisere dataene de er basert på. Dette kan bli identifisert v.h.a. søk blant metadata.

Et annet bruksområde kan være bruk av metadata for å vedlikeholde informasjon. Dette kan være ønskelig fordi verktøy, brukergrensesnitt og informasjon endres over tid. Da er det et behov for å vite hvordan data kan aksesseres, hva som er kilden, om det er flere kilder til samme informasjon og hva er i tilfelle forskjellen mellom disse.

Et tredje tilfelle er automatisk produktgenerering. Dette kan være en værrapport til f. eks. en avis, hvor man har et bestemt kartunderlag hvor værdata blir lagt på avhengig av observasjoner og symbolene for været hentes fra en kundebestemt database. Dette viser at metadata kan brukes til verdiøkning av eksisterende data, sikre kompabilitet med fremtidige systemer og automatisering av oppgaver. Det viser også at metadata gjør et system adaptivt m.t.p. nye arbeidskontekster.

6.3.4 USA's atomvåpenprogram

Historisk sett har atomvåpenprogrammet vært baset på en individuell „need to know" praksis ved tilgang til informasjon. Lownsbery og Newton [34] beskriver i et paper hvordan dagen situasjon med forbud mot kjernefysiske tester, reduserte budsjetter, nøkkelindivider som pensjoneres og konsolidisering av fasiliteter fremmer et behov for standardisering av metadata for å lokalisere, aksessere og effektivt bruke data, informasjon og kunnskap fra både gamle og kommende våpenaktiviteter.

De har kommet frem til en spesifikasjon av metadata som er standardisert, men utformet slik at det mulig å spesifisere informasjon fra flere ulike kilder med den samme metadatastandarden. Denne vil ikke bli presentert her. Vesentlig er imidlertid at deres arbeid viser at metadata kan brukes i et kunnskapsnettverk for utveksling og innsamling av informasjon, og samtidig sikre at denne aksesseres av de rette personene. Metadata

kan da støtte opp om idéer innen Knowledge Management, noe som igjen har relevans for kontekstbasert arbeids- og informasjonsorganisering.

6.3.5 Datamining

Datamining er en prosess for å oppdage tidligere ukjente og potensielt nyttige mønstre i store databaser [36]. Det finnes i dag store informasjonssystemer med store mengder data lagret som standard filer. Selv om disse er teknisk sett fysisk aksesserbare, er den logiske tilgjengeligheten dårlig p.g.a. mangel på strukturert kunnskap om dataene, m.a.o. metadata [37].

Både Breitender et al [37] og Cleary et al [36] mener at med dagens teknologi er det ikke mulig med full automatisering av datamining, bl. a. fordi mye metadata ligger implisitt lagret i f. eks. filer, og derfor kreves noe kunnskap hos brukeren om informasjonsdomenet ved søk. Det er også praktiske hensyn, som at omgivelser og organisasjonsstruktur varierer, og derfor må dataminingprosessen tilpasses.

Dette betyr at ved implementering av datamining for et system må metodikken og reglene tilpasses miljøet det skal brukes i, men brukerne kan få automatisk genererte data. Systemet vil imidlertid ikke oppdage nye interessante mønstre i databasen uten medvirkning fra enten bruker eller den som implementerer systemet.

Metadata må derfor være definert eksplisitt for å muliggjøre lett aksess og avanserte søk. Disse søkene vil være avhengig av brukerens arbeidskontekst. Dette viser at metadata kan brukes for å gi ny kunnskap basert på den lagrede informasjonen tilpasset ulike arbeidskontekster.

6.4 Muligheter og utfordringer ved bruk av metainformasjon

6.4.1 Muligheter

Med utgangspunkt i ISO (1179) – standardens beskrivelse og på bakgrunn av studiet av andres erfaringer, er det mulig å generalisere mulighetene som finnes ved bruk av metainformasjon. Disse er presentert i tabell 5.1.

Beskrive dataelementer	Metadataene kan brukes for å beskrive ulike data-elementer. Dette gir mulighet for å få informasjon om et dataelement uten å undersøke selve elementet.
Søk	Metadata kan enten lagres som en del av dataelementene eller i en egen metadatabase. Hvis man har kunnskap om elementenes oppbygning kan man foreta søk ved implisitt lagring. Eksplisitt lagring gir mulighet for raskere søk, fordi man da ikke må ekstrahere metadata fra hvert enkelt element hver gang.
	Metadataene kan om ønskelig beskrive innholdet i

Transparent presentasjon	dataelementene uavhengig av elementets oppbygning og fysiske lokasjon. Dette gir mulighet for å få presentert relevante dataelementer uten å ha kunnskap om av hvilken type disse er eller alle steder relevant informasjon befinner seg.
Kunnskapsnettverk	Fordi en transparent presentasjon av metadata er mulig, kan flere ulike databaser knyttes mot en metadatabase. Disse vil da utgjøre et skalerbart kunnskapsnettverk. Er utformingen av metadataene generell nok, vil alle typer dataelementer kunne bli beskrevet i en metadatabase.
Verdiøkende	Ved at dataelementene blir bedre beskrevet og gjort mer tilgjengelige, vil virksomheter kunne utnytte den informasjonen de allerede har på en bedre måte, og dette vil øke verdien av eksisterende data. Hvis forholdene ligger til rette kan nye tjenester basert på de samme dataelementene tilbys som følge av innføring av metadata. Dette kalles Business Scope Redefinition [9].

Tabell 5.1 Muligheter ved bruk av metadata

Å beskrive dataelementer og gjennomføre søk må ansees som grunnleggende muligheter ved bruk av metainformasjon. Større muligheter for utnyttelse av virksomheters informasjonsdomener gir de tre siste mulighetene. Disse mulighetene viser at metadata kan benyttes for å støtte de kravene til et informasjonssystem som er definert i kapittel 5.6 og problemstillingen gitt i kapittel 2. Dataelementene kan v.h.a. metadata klassifiseres med ulike parametre, det er mulig å jobbe mot flere informasjonsdomener og gi brukeren tilgang til et stort kunnskapsnettverk uavhengig av fysisk lokasjon. Dette viser at metadata er velegnet ved kontekstbasert arbeids- og informasjonsorganisering.

6.4.2 Praktiske utfordringer

Erfaringene har også avdekket noen hensyn som må taes ved praktisk bruk av meta-informasjon. For at brukerne skal benytte seg av metainformasjon ved lagring, må de føle at de får noe igjen for det [30]. Dette vil si at brukerne bør ha et behov for de mulighetene bruk av metainformasjon gir, eller at bruken kan forenkle, effektivisere eller tilføre arbeidet deres ny verdi.

Systemet må heller ikke være for komplekst, da dette vil føre til at uerfarne brukere vil unngå å bruke det [35]. Store mengder metadata i et komplekst system vil faktisk føre til et behov for et eget metadatamanagement-system [29]! Dette vil igjen kreve ressurser, og for at bruken av metainformasjon skal være verdiøkende for virksomheten, må ressursbehovet ved bruk av et metamanagementdatasystem være mindre enn gevinsten ved effektivisering eller verdiøkning av arbeidet.

6.4.3 Lagring av metadata

Det er prinsipielt to mulige måter å lagre metadata på. De kan enten lagres sammen med dataelementene, eller lagres for seg i en egen database. I mange tilfeller lagres metadata

sammen med f. eks. en tekstfil, og metadataene kan da være dato, forfatter, emne eller andre.

Det er som vi har sett også mulig å lagre de for seg. Dette kan enten gjøres ved at de lagres eksplisitt når de blir definert, eller at de blir ekstrahert fra lagrede dataelementer. De ulike paperne pekte på at skal søk blant dataelementer være effektivt og transparent, er søk i en egen metadata-base med referanse til fysisk lokasjon mest hensiktsmessig.

7 Modell for informasjonssystem som støtter kontekstbasert arbeids- og informasjons-organisering

Med basis i de funnene som er gjort ved behandling av teori for kontekstbasert arbeids- og informasjonsorganisering og de omgivelsene og avgrensningene som defineres vil det bli foreslått en teoretisk modell for et informasjonssystem som støtter kontekstbasert arbeids- og informasjonsorganisering.

7.1 Mål og avgrensninger for modellen

Modellen skal tilfredsstille de kravene som har fremkommet ved teoretisk behandling av kontekstbasert arbeids- og informasjonsorganisering. I tillegg er det nødvendig å definere hvilke avgrensninger som er gjort. Omgivelsene for modellen er definert i oppgave-beskrivelsen i kapittel 2.3.

Denne modellen vil ha et fokus på å kunne lagre og organisere informasjon slik at denne senere kan gjenfinnes ved filtrering, søking og sortering på en enklest mulig måte avhengig av konteksten. Denne metainformasjonen må i samsvar med oppgaveteksten kunne spesifiseres med et minimum av innsats, men samtidig bevare kvaliteten. Ved å unngå kompleksitet for brukeren og samtidig gi mulighet for forenkling og effektivisering ved informasjonssøk, vil man møte de praktiske utfordringene beskrevet i kapittel 6.4.2.

Modellen vil derfor omfatte hvilken metainformasjon som er nødvendig samt hvordan denne skal organiseres i tillegg til selve dataelementene for å støtte kontekstbasert arbeids- og informasjonsorganisering. Det vil derfor i denne modellen ikke bli fokusert på hvordan metainformasjonen eller søkeresultatene blir visualisert. Heller ikke algoritmer for søk eller å knytte modellen opp mot et bestemt informasjonssystem. Hvilken fysisk eller logisk representasjon de ulike parameterne – eller attributtene – skal ha vil ikke bli behandlet. Metainformasjonen parameterne representerer vil bli det som blir diskutert. Hvordan parameterne skal implementers i et informasjonssystem er en separat problemstilling.

Denne modellen er nødvendig i arbeidet med å finne muligheter i Office 2000 og Exchange for å identifisere ulike kontekster. Modellen vil i kapittel 8 bli brukt som et utganspuntkt for løsningen. Som nevnt ovenfor er ikke Office 2000 og Exchange tatt hensyn til ved utvikling av denne modellen. Dette for at modellen skal kunne brukes også i forbindelse med andre informasjonssystemer, og for at modellen skal utvikles m.t.p. en optimal løsning av problemstillingen. Implementering i valgt teknologi er derfor skilt ut som en separat problemstilling.

7.2 Case

For å kunne eksemplifisere er det nødvendig å ha et case eller utgangspunkt. For å kunne utforme et virtuelt arbeidsmiljø er det i teorier innen kontekstbasert design og HCI pekt på at de ulike arbeidskontekstene må danne utgangspunkt for design. Dette case'et vil derfor også være et utgangspunkt i kapittel 10. Fordi oppgaven er gitt av HiA og undertegnede er student ved skolen, er et naturlig valg av utgangspunkt for eksemplifisering egen studiesituasjonen. Høgskolen i Agder, avdeling for teknologi, er derfor valgt som et case. Hvilke sider ved HiA det vil bli tatt utgangspunkt i er beskrevet nedenfor.

Høgskolen i Agder er delt i ulike avdelinger. En av disse er avdeling for teknologi. Denne avdelingen inneholder flere ulike studie retninger, bl. a. Høgskoleingeniør innen Telematikk og Sivilingeiør innen IKT. Disse ulike studieretningene består av flere ulike fag. Det er mulig å velge fag fra andre studieretninger i tillegg til fag fra egen som et supplement. Studenter vil derfor forholde seg til ulike fag.

Disse kan sammenlignes med prosjekter fordi man deltar i en begrenset tidsperiode, ofte 1 semester, og i løpet av en uke deltar man i flere ulike fag. Man vil imidlertid hele tiden være del av en fast organisasjon, nemlig Hia. De ulike studieretningene kan sammenlignes med avdelingene som det blir henvist til i kapittel 2.3.

I de ulike fagene vil det være lærere, studenter, kanskje studentassistenter, muligens flere lærere og i noen tilfeller forekomme det gruppearbeid hvor man jobber sammen med en mindre del av det totale antallet studenter i klassen. Disse ulike situasjonene gir ulike roller for de som tilhører det aktuelle faget. Dette kan sammenlignes med en prosjektsituasjon.

Disse ulike rollene gjør at f. eks. to studenter som jobber med et gruppearbeid vil forholde seg til hverandre i ulike kontekster. Enten som medstudenter i samme fag, eller som medlemmer av samme gruppe. I tillegg kan en av disse være tillitsvalgt for klassen. Av dette sees det at det kan det forekomme flere situasjoner, f. eks. ved mottak av mail, der det er behov for å skille informasjon fra hverandre basert på de ulike rollene. Det er også et behov for å kunne skille informasjon tilhørende studentassistenter, administrasjonen, ulike fag eller forskjellige lærere fra hverandre. I tillegg kommer varierende mengder informasjon av ikke-faglig karakter.

For lærere oppstår det også situasjoner som krever kontekstbasert arbeids- og informasjonsorgansiering. Ofte benyttes mail som metode for innlevering av rapporter og andre besvarelser, eller besvarelsene legges i felles områder på skolens nettverk. En lærer kan være tilknyttet flere ulike fag, og det er derfor behov for å skille inkommende mail eller lagret informasjon basert på disse ulike fagene samt hvilken tilkytning man har til de ulike fagene.

Lærere kan også komme i den situasjon at de har flere ulike roller i forhold til en klasse. De kan undervise i flere fag, det kan være fag med gruppearbeid hvor hver gruppe har ulike veiledere, og en av disse er også hovedforeleser i faget eller en lærer kan i tillegg til

å forelese et fag for en klasse ha administrative funksjoner som f. eks. studieleder. I tillegg vil også lærere forholde seg til ulike typer informasjon som ikke har relevans til de ulike studieretningene i varierende grad.

7.3 Organisering av metadata og dataelementer

For å organisere informasjon ble det i kapittel 4 funnet at metadata gir mulighet for dette. De mulighetene som metadata gir, egner seg godt til å imøtekomme kravene til et informasjonssystem for støtte av kontekstbasert arbeide. Kravene er presentert i kapittel 4.2.5, og mulighetene beskrevet i kapittel 5.4.1.

En sentral problemstilling ved bruk av metadata er hvilken av de to metodene for lagring av metainformasjon som skal benyttes. Som vi har sett er det mulig å enten lagre metainformasjonen sammen med dataelementene, eller i en egen metadatabase. For å velge mellom disse taes det utgangspunkt i kravene til et IS.

For å tilby rask tilgang til informasjon er det et krav at søk blant metadata må kunne utføres raskt. Som det er påpekt tidligere vil en implisitt representasjon av metainformasjon føre til ekstra prosessering ved søk fordi metadataene da må ekstraheres hver gang. Dette krever kunnskap om dataelementenes oppbygning.

Dette vil i tillegg vanskeliggjøre tilfredsstillelse av kravet til adaptivitet, fordi innføring av en eventuell ny type dataelementer vil kreve at brukeren eller systemet vet hvordan metainformasjon fra disse skal ekstraheres. Av denne grunn kan det være hensiktsmessig å plassere metadataene i en egen database, fordi dette vil både gi lettere prosessering ved søk, og samtidig gi mulighet for at metadataene kan beskrive et dataelements innhold, uten at brukeren har informasjon om dataelementets oppbygning.

Svakheten med denne løsningen er at det ikke er mulig å kontrollere om dataelementene til enhver tid befinner seg der metadatabasen gir en referanse til. En løsning på dette kan være en hybrid mellom de to ytterlighetene lagring av dataelementene hver for seg og lagring av metainformasjonen i en egen metadatabase. Hvis de ulike dataelementene legges i en tabell i en database, er det mulig å søke på ulike kolonner hvor metadataene kan være plassert.

Fjerning av et dataelement vil derfor også fjerne metadata fra kolonner som spesifiserer metainformasjon. For å gi tilgang til metadataene er det nødvendig med en internasjonal standardisert tilgangsmetode. Dette kan f. eks. være en Internett-tilgang basert på HTTP (Hypertext Transfer Protocol). Dermed vil en transparent presentasjon over ulike informasjonsdomener være mulig, basert på Internett-teknologi.

Etter å ha vurdert de ulike alternativene har det derfor blitt konkludert med at den beste måten å organisere dataelementer og metainformasjon, er å ha en database hvor både dataelementer og metadata kan legges i samme tabell. Disse dataene må kunne aksesseres via en internasjonalt standardisert tilgangsmetode som f. eks. HTTP.

7.4 Valg av parametre for informasjonsorganisering

Nedenfor vil hvorfor de ulike parameterne er valgt bli diskutert, og deretter presentert hver for seg med noe spesifisering av hvordan de bør settes. Til slutt vil det bli vurdert om modellen gir mulighet for å utnytte de mulighetene som er funnet ved bruk av metainformasjon tidligere.

7.4.1 Diskusjon av parametre

Ved valg av parametre – eller attributter i.h.t. ISO 11179 – er det tatt utgangspunkt i de kravene til et informasjonssystem som skal støtte kontekstbasert arbeids- og informasjonsorganisering presentert i kapittel 4.2.5 og målet med denne modellen, beskrevet i kapittel 7.1. Parameterne har prinsipielt to hovedmål:

- Klassifisere ulike kontekster
- Sikre kvalitet

Hvis ikke disse er oppnådd, vil heller ikke de andre målene som ønskes oppnådd med modellen nåes. Metainformasjonen skulle kunne spesifiseres med et minimum av innsats, men samtidig skal det sikres kvalitet. For å oppnå dette må det defineres hva som kjennetegner ulike arbeidskontekster, dette må være så generelt at flere ulike kontekster kan klassifiseres v.h.a. modellen og brukeren må i minst mulig grad aktivt måtte delta i setting av ulike parametre.

Bestemme type aktivitet

Koordinasjonsmodellen presentert i kapittel 2.3 egner seg godt som et hjelpemiddel. Problemet i oppgaven ble i forhold til denne modellen definert til å være hvordan man kan knytte riktig ressurs opp mot aktøren avhengig av hvilken aktivitet denne utfører. Konteksten vil m.a.o. avgjøre hvilken ressurs aktøren jobber med.

Aktiviteten vil for aktøren tilhøre en bestemt kategori av aktiviteter. En aktivitet kan f. eks. være en student som leverer inn en oppgave i forbindelse med et spesielt fag. Innen dette faget kan det forekomme flere ulike typer aktiviteter, men de vil alle sammen være tilhørende det samme faget. Derfor vil én parameter som bestemmer konteksten være hva slags kategori aktiviteten som utføres tilhører. Imidlertid må det for å bestemme konteksten spesifiseres noe mer. Som det er vist tidligere vil det være flere måter å kunne jobbe mot et fag for ulike personer. Disse ulike måtene medfører at aktøren har ulike roller i forhold til faget.

De ulike aktivitetene definerer implisitt en rolle, fordi hver aktivitet har spesielle særegenheter og krav til utførelse. Aktørens tilknytning til aktiviteten blir da å fylle denne bestemte rollen. Rollen vil sammen med parameteren kategori sette aktøren i en bestemt arbeidskontekst. For å definere en kontekst må det derfor både defineres i forhold til

hvilken kategori av aktiviteter (prosess, prosjekt, fag e.l.) en aktivitet utføres og hvilken relasjon aktøren har til denne aktuelle kategorien ved utførelsen av aktiviteten (rolle).

Denne resonneringen er også i samsvar med definisjonen i kapittel 5.1 av i hvilke situasjoner det er behov for kontekstbasert arbeids- og informasjonsorganisering. Også her fremkommer det at kjernen i problemet er å koble aktøren mot riktig informasjon, og at dette bestemmes av hva slags type aktivitet som utføres.

Svært mange kontekster kan bestemmes v.h.a. parameterne kategori og rolle. I tillegg vil det imidlertid være behov for noen andre parametere for å imøtekomme kravene definert i kapittel 5.6. Et viktig problem er om systemet blir adaptivt nok med de valgte parameterne. De parameterne som er funnet hittil kan som det er vist benyttes internt i en organisasjon eller gruppe. Problemer oppstår ved kontakt med eksterne aktører som kan være personer eller organisasjoner.

Eksterne aktører

Disse vil det kanskje ikke være mulig å plassere i andre kategorier enn ”aktiviteter i forhold til eksterne aktører”. Ved arbeide med disse må det en mer presis angivelse til for å finne informasjon. Informasjonen fra de eksterne aktørene vil i slike tilfeller være mulig å klassifisere basert på person og/eller organisasjon. Denne informasjonen kan f. eks. være mail man har mottatt eller noe man ønsker å finne i en database. Parameterne person og organisasjon gir også mulighet for bedre spesifisering ved organisering av informasjon, og dermed mulighet for raskere tilgang til relevant informasjon av høy kvalitet.

Format og navn på dataelement

En annen kontekst brukeren ofte befinner seg i er at man jobber med f.eks. et fag, men i ulike verktøy som f. eks. MS Word eller Borland C++. Da vil det være et behov for å finne informasjon basert på hvilket format dataelementene er lagret som. De ulike dataelementene må også navngis, og dermed er navn på dataelementet en parameter for å organisere informasjon.

I f. eks. et fag kan det tenkes at det finnes en tabell i en database hvor alle øvinger som skal leveres inn legges. Ved å navngi disse som øving1, øving2 osv, er det mulig å søke på kolonnen som inneholder navnet, få tak i alle som inneholder navnet øving2, og koble disse opp mot navnet på personen som har lagt filen der for dermed å få ut en liste over alle som har levert. Konteksten kan her være arbeide med øving2 for en lærer, og parameterne som bestemmer konteksten er i tillegg til rolle (lærer) og kategori (et fag) parameterne navn på dataelement og navn på eieren av filen. Dette viser at disse to siste parameterne kan ytterligere støtte opp om kontekstbasert arbeids- og informasjons-organisering.

Versjoner

For ulike dataelementer som ligger lagret i en database vil det – forutsatt at brukerne har tilgang – som oftest være mulig å endre innhold etter lagring. Dataelementene kan f. eks. være dokumenter, og en situasjon som kan oppstå er at i en prosjektgruppe i et fag skal det lages en rapport. Denne skal flere skrive sammen, og dette kan gjøres ved at man jobber med rapporten etter tur, og lagrer siste versjon etter å ha utført sin del av arbeidet.

Dermed oppstår det et behov for å kunne skille mellom ulike versjoner. En naturlig måte å gjøre dette på er å skille mellom ulike tidspunkter for lagring. Tiden er en parameter som automatisk kan lagres av systemet. En problemstilling er om man skal ha mulighet for å spore alle endringer som er gjort i forhold til tidligere versjoner. I denne modellen vil det ikke bli tatt hensyn til dette ved valg av parametre. Dette er i samsvar med avgrensningene gjort i kapittel 2.3. Det finnes mulighet i ulike programmer, som f. eks. MS Word 2000, for å sammenligne ulike versjoner og identifisere endringer.

Skal ulike versjoner ligge lagret, d.v.s. man lagrer ikke oppå den gamle versjonen, kan tiden brukes for å skille, men fordi det ikke er sikkert man lagrer alle ulike versjoner, er det nødvendig med en egen parameter for å angi hvilken versjon det er. Dette gir også mulighet for å definere f. eks. et dokument som ferdig revidert. Dette gir igjen mulighet for automatikk i informasjonsorganiseringen, ved at f. eks. et script kan flytte alle ferdige dataelementer til et spesielt sted basert på de andre parameterne.

Fordi flere ulike personer kan skrive de forskjellige versjonene, er det et behov for å skille mellom hvem som har gjort endringer i den aktuelle versjonen. En parameter for hvem dataelementet er revidert av er nødvendig. Dette muliggjør også søk etter dataelementer generert av en spesiell person. I en gruppe kan det tenkes at bare gruppeleder er autorisert for å kunne definere et dataelement som ferdig. Ved at det kan sjekkes om alle ferdige dataelementer er lagret av en som har autorisasjon for dette, er det mulig å sikre kvalitet.

Disse parameterne kan avhengig av program og operativsystem i stor grad settes automatisk, men eventuelt gi brukeren mulighet for å endre f. eks. versjonsnr. Dette sikrer kvalitet ved dataelementene som lagres.

Fysiske parametre

I enkelte kontekster er det hensiktsmessige å kunne jobbe opp mot bestemte informasjonsdomener. Dette kan f. eks. være en database som Exchange. Fysisk lokasjon blir derfor en parameter. Denne er også nødvendig for å kunne lokalisere hvor dataelementene befinner seg.

Hvis konteksten til en student blir definert til å være student og at man jobber med dokumenter i et spesielt fag, kan parameteren fysisk lokasjon være nødvendig for å koble

studenten til riktig informasjonsdomene hvor f. eks. oppgavesamlinger i faget befinner seg.

Vurdering av parameterne opp mot kravene til informasjonssystemer

Disse parameterne ble etter vurdering av også flere andre parametre funnet som et minimum for å kunne spesifisere ulike arbeidskontekster og benyttes ved kontekstbasert arbeids- og informsjonsorganisering. Diskusjonen i dette kapitlet viser at ulike arbeidskontekster kan støttes v.h.a. disse parameterne.

De er valgt slik at de gir gode muligheter for å klassifisere flere typer kontekster, og de forskjellige parameterne gir tilsammen mulighet for å organisere etter flere ulike måter å jobbe på. Dette gjør modellen adaptiv samtidig som komplekse bruksmønstre støttes.

Parameterne vil, fordi kontekstbasert arbeids- og informasjonsorganisering gir bedre mulighet for å koble aktøren mot riktig ressurs avhengig av aktiviteten, gi raskere tilgang til aktuell informasjon. Parametre for f. eks. versjonskontroll sikrer kvalitet ved informasjonen som er lagret. I neste kapittel vil det også spesifiseres nærmere hvordan setting av ulike parametre kan utføres for å sikre kvaliteten. Av dette kan det trekkes den slutningen at modellen tilfredsstiller de kravene som er spesifisert i kapittel 5.6.

7.4.2 Presentasjon av valgte parametre

Nedenfor blir de ulike parameterne kort presentert. Det blir vurdert om de bør settes mauelt eller automatisk, eventuelt som en kombinasjon av disse to mulighetene.

Kategori

Typer kategori kan være fagområde, prosjekt, sak, prosess e.l. For å unngå uoversiktlig bruk av kategorier, bør disse innenfor en organisasjon velges fra et forhåndsdefinert sett av kategorier. Hvis brukeren har blitt identifisert til å jobbe i en kategori, kan denne kategorien settes som standard ved arbeide i forhold til ulike informasjonsdomener. En kategori kan identifiseres ved at brukeren velger å jobbe med et spesielt dataelement, og kategorien kan finnes ved å se på metadataene som tilhører dataelementet. En bruker kan også selv velge å sette hvilken kategori som er aktuell for konteksten man er i.

Rolle

Ulike roller kan være student, lærer, veileder, prosjektleder e.l. For å unngå uoversiktlig bruk av roller, bør disse innenfor en organisasjon velges fra et forhåndsdefinert sett av roller. Hvis brukeren har blitt identifisert til å jobbe i en rolle, kan denne rollen settes som standard ved arbeide i forhold til ulike informasjonsdomener. En rolle kan identifiseres ved at brukeren velger å jobbe med et spesielt dataelement, og rollen kan finnes ved å se på metadataene som tilhører dataelementet. En bruker kan også selv velge å sette hvilken kategori som er aktuell for konteksten man er i.

Format

Ulike formater kan være ASCII, JPG eller andre filformater tilhørende ulike programmer eller internasjonalt definert. Hvis en bruker jobber i et spesielt program eller jobber med et spesielt dataelement, kan type format finnes enten ved å se på metadata tilhørende dataelementet eller programmet kan angi hvilke type dataelementer det gjenkjenner. Formatet kan derfor enten settes manuelt ved lagring eller søk, eller det kan settes automatisk som et resultat av identifisering av kontekst eller fordi et program ikke gjenkjenner andre typer dataelementer.

Fysisk lokasjon

Fysisk lokasjon vil bety hvilken database eller hvilket informasjonsdomene man jobber mot. Dette kan enten settes manuelt av brukeren, eller hvis arbeide mot en spesiell lokasjon er identifisert kan denne være standard ved videre arbeide med informasjon. Ved søk blant en transparent presentasjon av data, vil denne parameteren fortelle hvor informasjonen befinner seg, og vil være en parameter som ytterligere kan klassifisere informasjon i forhold til konteksten.

Tid

Denne parameteren settes automatisk av systemet, og er et tidsstempel på dataelementet. Den bør ikke være mulig å manipulere p.g.a. krav til kvalitet ved IS. Parameteren må settes eller oppdateres hver gang et dataelement lagres.

Revisjon

Denne parameteren kan inneholde et tall eller tekst. Brukeren bør kunne bestemme hvordan denne parameteren skal representeres. Det vil i praksis si at parameteren må kunne settes v.h.a. ASCII tegn. Hvis den representeres som et tall, kan systemet designes slik at det automatisk inkrementerer tallet, og dermed er det mulig med automatisk generering av denne parameteren.

Opprettet av

Denne parameteren beskriver hvem som opprettet eller lagret dataelementet. Hvem denne personen er, kan i de fleste tilfeller identifiseres av systemet fordi brukeren som oftest må logge seg på systemet med brukernavn og passord for å kunne aksessere de ulike informasjonsdomenene. Denne parameteren bør settes automatisk for å sikre kvalitet ved informasjonen som ligger lagret samt for å ivareta sikkerhet. Med dette menes ikke sikkerhet som for store systemer med krav til sporbarhet m.t.p. hvem som har utført alle aktiviteter, men fordi det ikke skal være mulig å lagre noe i en annens navn.

Revidert av

Denne parameteren beskriver hvem som opprettet eller lagret dataelementet. Hvem denne personen er, kan i de fleste tilfeller identifiseres av systemet fordi brukeren som oftest må logge seg på systemet med brukernavn og passord for å kunne aksessere de ulike

informasjonsdomenene. Denne parameteren bør settes automatisk for å sikre kvalitet ved informasjonen som ligger lagret samt for å ivareta sikkerhet. Med dette menes ikke sikkerhet som for store systemer med krav til sporbarhet m.t.p. hvem som har utført alle aktiviteter, men fordi det ikke skal være mulig å lagre noe i en annens navn.

Organisasjon

Denne parameteren beskriver hvilken organisasjon brukeren som har opprettet dataelementet tilhører, eller den kan være en den eneste parameteren som identifiserer hvem som har opprettet dataelementet. Også denne parameteren bør settes automatisk for å sikre kvalitet, men fordi det kan oppstå situasjoner, f. eks. ved konsulentvirksomhet der denne parameteren ønskes for å beskrive hvilken organisasjon det jobbes for ved det aktuelle prosjektet, må den også kunne endres manuelt.

Navn på dataelement

Denne parameteren navngir dataelementet, og må settes av brukeren ved lagring, hvis ikke dataelementet er åpnet for f. eks. revisjon, og derfor vet systemet allerede navnet på dataelementet.

7.5 Modell

Nedenfor vil modellen bli kort presentert som en oppsummering på det som er diskutert hittil i dette kapitlet.

7.5.1 Krav som modellen tilfredsstiller

Modellen tilfredsstiller følgende krav:

- Støtte for ulike arbeidskontekster
- Adaptiv
- Støtter komplekse bruksmønstre
- Tilbyr rask tilgang til informasjon
- Tilbyr relevant informasjon av høy kvalitet

7.5.2 Oversikt over modellen

Modellen vil kort bli beskrevet ved å angi hvilke parametre som er valgt, hvordan de skal lagres og hvordan organiseringen av informasjon bør være.

Lagring av metadata og dataelementer

Etter å ha vurdert de ulike alternativene har det blitt konkludert med at den beste måten å organisere dataelementer og metainformasjon, er å ha en database hvor både

dataelementer og metadata kan legges i samme tabell. Disse dataene må kunne aksesseres via en internasjonalt standardisert tilgangsmetode som f. eks. HTTP.

Parametre

Nedenfor blir parameterne presentert. Det er også gitt oversikt over om de bør settes manuelt (M) eller automisk (A). I tilfeller der diskusjonen har vist at begge tilfeller kan være aktuelle, er de gitt ved M/A.

Parameter	Automatisk/ manuell setting av parameteren
Kategori	M/A
Rolle	M/A
Format	M/A
Lokasjon	M/A
Tid	A
Revisjon	M/A
Opprettet av	A
Revidert av	A
Organisasjon	M/A
Navn på dataelement	M/A

Tabell 7.1 Oversikt over parametre i modellen og metode for setting av disse

8 Presentasjon av teknologi

I dette kapitlet vil teknologien som er utgangspunktet for illustrasjon av kontekstbasert arbeids- og informasjonsorganisering bli presentert. Exchange og Office 2000 var gitt som utgangspunkt i oppgaveteksten. Weboffice var også gitt som utgangspunkt, men det ble funnet at det var mer hensiktsmessig å konsentrere seg om en løsning basert på Exchange og Office 2000. At fokuset ble konsentrert om de to sistnevnte teknologiene ble også gjort i samråd med veileder. Weboffice er derfor ikke presentert her, fordi den ikke er brukt i løsningen.

Det vil bli presentert sider ved teknologien som er relevant for oppgaven og som benyttes i løsningen. For Office 2000 er det valgt å konsentrere seg om Word 2000 og Outlook 2000. Dette fordi dokumenter og mail er to dataelementer det ofte er stort behov for å behandle avhengig av kontekster. Det vil være mulig å behandle andre dataelementer tilhørende andre programmer i Office 2000-pakken tilsvarende som Word 2000 og Outlook 2000 er behandlet her. En annen grunn til at kun disse to er behandlet er for å begrense omfanget av oppgaven.

De to programmene er valgt ut fordi de kan nært kobles mot Exchange, og er gode som utgangspunkt for å diskutere mulighetene for kontekstbasert arbeids- og informasjons-organisering i Office 2000 og Exchange. Fordi andre verktøy tilhørende Office 2000 i varierende grad kan kobles mot Exchange på samme måte, vil det ikke være nødvendig å diskutere disse også. Det vil som nevnt over bli fokusert på hvordan kontekstbasert arbeids- og informasjons-organisering kan implementeres i Office 2000 og Exchange. De andre verktøyene vil ikke kunne illustrere bedre hvordan implementeringen kan gjøres, og er derfor ikke diskutert.

Til slutt er Document Management Extensions kort presentert. Denne teknologien ble oppdaget under arbeidet med diplomoppgaven. Da den tilbyr interessante muligheter som kan benyttes ved kontekstbasert arbeids- og informasjons organisering, er den kort presentert. I kapittel 10 vil det bli referert til mulighetene den tilbyr ved diskusjon av mulighetene for kontekstbasert arbeids- og informasjonsbehandling med Office 2000 og Exchange.

Alle bilder i dette kapitlet er hentet fra Outlook 2000 Product Enhancement Guide [40].

8.1 Exchange server

Nedenfor vil Exchange Server bli beskrevet. Funksjonene vil bli kort presentert, men fokus vil bli på mulighetene for lagring av data, fordi dette kan benyttes ved kontekst-basert arbeids- og informasjonsorganisering.

8.1.1 Generelt

Microsoft‘s Exchange Server tilbyr en infrastruktur for koordinering og utveksling av informasjon. Kjernen i systemet er en objektdatabase. Denne kan brukes for lagring av

svært mange typer objekter. Disse kan være tilhørende Office 2000, eller av en annen ukjent type, men av en slik form at dataene kan lagres i databasen. I en database er det mulig å lagre opptil 16 terabyte med informasjon.

De ulike objektene kan lagres i samme tabell, eller folder som de kalles i Exchange. Hvis objektet er av en type som gjenkjennes av Exchange er det mulig å ha flere ulike visualiseringer, eller views, av dataene. Det er f. eks. mulig å sortere på parametre tilhørende Word-dokumenter. Tabellene er strukturerte. Derfor er metadata og f. eks. et dokument lagret sammen. Metadataene ligger lagret på en strukturert måte slik at det er mulig å søke i ulike kolonner med sammen type metadata. Dermed kan metadatasøk gjøres uten å undersøke hele dataelementet.

Serveren må kjøres sammen med en MS NT Server. NT brukes som operativsystemet Exchange kjører på. Det er mulig å koble brukerne i Exchange opp mot brukerne i NT for felles administrasjon. Mellom ulike Exchange-servere og mellom en Exchange-server og en Outlook-klient er det mulighet for replikering av objekter. Ved replikering blir bare forandringer sendt mellom de aktuelle enhetene.

8.1.2 Exchange Server Directory

Dette er en hierarkisk katalogstruktur som inneholder kritisk informasjon om en organisasjon. Det er mulig å replikere informasjonen, slik at endringer foretatt på en server blir oppdatert på alle servere i organisasjonen eller at en bruker kan jobbe med hele eller deler av Exchange Server directory offline.

Exchange Server directory fungerer både som en database og en tjeneste. Katalogen inneholder informasjon om alle objektene som er nødvendig for å sende og motta meldinger innen en site. En site er et nettsted, og kan tenkes på som et virtuelt punkt som utgjør f. eks. en organisasjons kontakt med Internett. Informasjon som befinner seg her inkluderer alle e-mail mottakere innen siten, hvilke servere som befinner seg innenfor siten og all form for rutingsinformasjon.

De ulike brukerne for Exchange blir som det er nevnt lagret i Exchange Server directory. I tillegg til bare å definere ulike brukere er det mulig å legge til ulik tilleggsinformasjon om de ulike brukerne, som f. eks. telefonnummer og adresse. En tjeneste som er et resultat av Exchange Server directory er Global Adress List (GAL). GAL er implementert i Outlook 2000 ved at e-mail adresser kan hentes fra denne, i tillegg til ulik tilleggsinformasjon som er lagret i forbindelse med den aktuelle Exchange brukeren.

8.1.3 Foldere

Kjernen i Exchange-serverens samarbeidsteknologi er bruk av „public folders“, eller foldere der innholdet er gjort tilgjengelige for flere brukere. „The Information Store“ er betegnelsen på en type tjeneste for lagring av informasjon. Hver Exchange server inneholder en „Information Store“ som kan inneholde en eller to databaser. En privat for ulike mailbokser, og en offentlig for ulike public folders.

Offentlige foldere kan bli aksessert av mange typer klienter og ved å bruke ulike typer protokoller for kommunikasjon. Disse folderne kan inneholde standard former for å legge til eller lese informasjon fra folderne, og ulike visualiseringer og filtrering kan benyttes for å vise informasjonen. Dette vil bli nærmere beskrevet i kapittel 8.4.2.

Folderne blir vist i en virtuell trestruktur, som ikke nødvendigvis forteller på hvilke servere informasjonen befinner seg. Dette for å gjøre det lettere å finne informasjon. Denne informasjonen blir gjort tilgjengelig på tre ulike nivåer:

- Global. Default, og gir tilgang for alle i organisasjonen. Også default setting for anonyme brukere.
- Gruppe. Tilgang for en spesiell gruppe brukere som befinner seg på en liste.
- Bruker. Alle brukere som skal ha tilgang må spesifiseres.

I tillegg er det mulig å sette rettigheter til ulike roller i forhold til folderen, som f. eks. author, owner, editor og reviewer. Disse rollene brukes bare for å kunne skille mellom ulike sett av rettigheter, og personer kan defineres til ulike roller for å oppnå ulike rettigheter. De ulike folderne kan settes opp slik at en person godkjenner alt som legges der. Det er også mulig å sende e-mail til en slik folder for dermed å spare plass ved at ulike brukere av systemet henter informasjon fra et felles område istedenfor at alle får en e-mail med samme informasjon.

For å aksessere disse ulike folderne kan klienten Outlook brukes, som er spesielt designet for Exchange. Det er også mulig å aksessere Exchange via andre klienter. Exchange kan aksesseres via Internettklienter som benytter enten Network News Transfer Protocol (NNTP), Internet Mail Access Protocol versjon 4 (IMAP4) eller Hypertext Transfer Protocol (HTTP).

NNTP er en internett-standard som definerer server til server replikering av data i form av artikler. Disse er organisert som et hierarki av nyhetsgrupper, som er identisk med diskusjonsgrupper i Exchange Server. Exchange Server støtter både server til server replikering og muligheten for at en enkeltstående klient kan lese informasjon i folderne.

IMAP4 er en internett-standard som definerer en måte for klienter å aksessere meldings-informasjon på serveren. Exchange Server er IMAP4-kompatibel, så en klient som støtter IMAP4 kan derfor aksessere meldingstjenester på Exchange. Noen av disse tjenestene inkluderer sende og motta mail samt aksessere foldere. En IMAP4-klient vil dermed kunne bruke disse folderne.

HTTP er den primære protokollen som brukes for å distribuere informasjon på World Wide Web. Den benyttes for å sende grafikk og dokumenter fra en web-server til en web-browser. HTTP er en klient til server protokoll, d.v.s. at en klient som kjøres på brukerens maskin sender forespørsel om data, og serveren mottar forespørselen og sender relevant data tilbake. HTTP-servere som Exchange kan i tillegg gjøre mer enn bare å sende tilbake data ved mottak av forespørsler. Script som aksesserer andre server-tjenester kan kjøres

som et resultat av en forespørsel, og serveren kan da sende data til klienten avhengig av resultatet av scriptet som kjøres. Dette er tilfellet med Active Server Pages (ASP), som beskrives i kapittel 7.2.

8.1.4 Innebygde Information Management verktøy

Exchange har både innebygde og konfigurerbare funksjoner for manipulasjon av informasjonen i databasen. Hvis to brukere jobber med det samme dokumentet e.l. og prøver og lagre samtidig, vil de bli varslet om dette. Det er da mulig å lagre det ene som master og integrere endringer fra det andre dokumentet i master-dokumetet, eller begge verjonene kan lagres.

I en folder er det mulig å sette en begrenset tid informasjon kan være lagret. Informasjonen kan etter denne tiden slettes eller flyttes til en anne folder. Dette hindrer store mengder unyttig informasjon å ligge i folderen og oppta plass. For å muliggjøre dette tilbyr Exchange bruk av ulike regler.

Regler velges blant et utvalg av muligheter, og kan være av typen reagere på en hendelse, f. eks. mail til „inbox“ fra en person eller med et spesielt „subject“, og utfør noe som et resultat av denne hendelsen, f. eks. flytte mailen til en annen folder. Dette gir i noe grad mulighet for kontroll av informasjonsflyten.

Mer fleksibilitet tilbys med en mulighet kalt „Event Scripting Agent“. Med denne er det mulig å starte et script basert på en av følgende hendelser i folderen: En melding (mail, dokument etc.) blir laget, en melding blir endret, en melding blir slettet eller en timer slår til (f. eks. en gang daglig). Ofte er det CDO som benyttes for å utføre hendelser, men også andre objekt-bibliotek som ADO (for databaser) eller ADSI (for manipulering av mapper). Det er også mulig å definere egne COM-komponenter. For scripting kan Active X-script-språk benyttes. Dette vil i praksis si VBScript eller JScript.

Tjenestearkitekturen for Event Scripting Agent er strukturert slik at tjenesten, events.exe, sender events (eller begivenheter), som f. eks. opprettelsen av en ny melding i en folder, til rette eventhandler (behandleren for begivenheten), en agent, med noe informasjon om kilden til begivenheten, meldingen og folderen som started den aktuelle begivenheten. Begivenheter som støttes er opprettelse av nye dataelementer i en folder, forandringer av dataelementer, sletting av dataelementer og en timer som løses ut f. eks. hver 5. time.

V.h.a. replikeringsmekanismer er det mulig å jobbe offline, og ved kobling mellom klienten og Exchange serveren igjen sjekke hvilke begivenheter som eventuelt har funnet sted. Denne teknologien kalles Incremental Change Synchronization (ICS). ICS sørger for at ikke begivenheter unngår å bli oppdaget ved arbeide offline.

8.2 CDO og ASP

For å kunne lage script som kan kjøres på Exchange serveren eller en IIS-server, er det nødvendig å benytte CDO og ASP. Nedenfor er disse teknologiene kort beskrevet. Disse

teknologiene var i mindre grad familiære for undertegnede før arbeide med diplom-oppgaven begynte. For å ha grunnlag for å vurdere hvordan Office 2000 og Exchange kunne benyttes for kontekstbasert arbeids- og informasjonsorganisering, var det derfor nødvendig å sette seg inn i disse to teknologiene. Disse teknologiene ble også av veileder ved begynnelsen av arbeidet med diplomoppgaven pekt på som av vesentlig betydning å sette seg inn i og lære å bruke.

8.2.1 Beskrivelse av CDO

Collaboration Data Objects (CDO) er et objekt-bibliotek som tilbyr interaksjon med Microsoft's Messaging Application Programming Interface (MAPI). Istedenfor å kreve bruk av C/C++ som MAPI gjør, tilbyr CDO mulighet for programmering i alle utviklingsverktøy som lager COM-objekter, som f. eks. ASP, Visual Basic og Microsoft Visual C++.

CDO er tidligere kjent som OLE Messaging og Active Messaging. Siste versjon når dette blir skrevet er CDO 1.21. CDO innstalleres enten ved å innstallere Outlook på en maskin eller Outlook Web Access på web-serveren. Script som benytter CDO kan da kjøres på den aktuelle maskinen/serveren. Det finnes en egen hjelp-fil tilhørende CDO. Denne befinner seg på innstallasjons-CD'en til Outlook, Exchange eller ServicePack 1 eller høyere for Exchange. Den kan også lastes ned fra cdolive.com [42].

CDO er fordelt på to dll-filer (dynamic link library). CDO.dll inneholder kjernefunksjoner for samarbeid som sending av meldinger, aksessering av kataloger og se ledig/opptatt-kalenderfunksjoner. CDOHTML.dll er CDO Rendering-biblioteket, d.v.s. det tillater automatisk konvertering av informasjon lagret inne i Exchange Server til HTML ved å bruke ulike views, farger og formater. CDO Rendering-biblioteket innstalleres på web-serveren ved innstallasjon av Outlook Web Access.

I kapittel 8.3 vil Outlook 2000 og tilhørende objektmodell bli presentert. Objektmodellen for Outlook og CDO er komplementære teknologier. Outlook objektmodellen brukes ved aksess av spesiell informasjon i Outlook, som f. eks. Task eller Journal dataelementer. Nedenfor vil objektmodellen for CDO bli kort beskrevet. En fullstendig beskrivelse vil ikke bli gjort fordi dette kan finnes i CDO-hjelp filen, men de viktigste mulighetene og sammenhengene vil bli presentert.

CDO biblioteket er hierarkisk oppbygget, og består av objekter og samlinger. Det er mulig å opprette child-objekter av de opprinnelige (parent). For forskjellige objekter finnes samlinger som tilsynelatende er identiske, men de informasjonen hver samling aksesserer er unik for det objektet de er referert til.

Session-objektet er det objektet som befinner seg på det høyeste nivået i den hierarkiske strukturen. Dette fordi det er nødvendig med en sesjon for å kunne aksessere informasjon i en Exchange Server database. Under Session-objektet kommer viktige objekter som Inboks, Outbox og Infostores. Under disse kommer f. eks., som tilfellet er for Outbox,

Messages og Attachments. Strukturen er derfor ganske logisk, og ble funnet oversiktlig å jobbe med ved utprøving av teknologien vist i kapittel 9.

8.2.2 Beskrivelse av ASP

Active Server Pages (ASP) er standard tekstfiler som inneholder HTML og script. Et script kan bli skrevet ved å benytte ethvert ActiveX-språk, som VBScript eller JScript. ASP-filer skiller seg fra ordinære HTML-sider ved at de har endelsen .asp i filnavnet. Ved installasjon av IIS installeres en komponent kalt Internet Server Application Programming Interface (ISAPI). Denne leser alle filer med en .asp endelse.

Den andre måten ASP-filer skiller seg fra HTML-filer er at de blir prosessert på selve webserveren, og resultatet blir sendt til en nettleser som f.eks. Internet Explorer som en HTML-side. Denne siden kan, men må ikke, inneholde script som skal kjøres hos klienten. Fordelen med dette er at ved aksessering av en ASP-fil kan ulike prosesser settes i gang. Dette kan være et resultat av at en som aksesserer ASP-filen via Internett skriver inn ulike parametre. Fordi resultatet sendes som HTML, vil de fleste moderne nettlesere være i stand til å visualisere resultatet for brukeren.

Det anbefales å benytte CDO istedenfor objektmodellen til Outlook 2000 fordi CDO ble designet for å være basert på å være multiuser og serverprosessert. En web-side vil ofte være designet for at flere brukere skal kunne benytte denne, og ASP-scriptet kjøres på serveren.

De ulike scriptene inneholder HTML-kode blandet med script-kode. Script-koden, f. eks. skrevet i Visual Basic (VBScript), merkes ut fra annen kode med tegnene <% og %>. En webserver med ASP-script vil ofte ha en egen fil med endelsen .asa. Denne filen inneholder kode for hva som skal utføres ved start/slutt for både sesjoner og selve web-applikasjonen.

Fordelen ved å bruke global.asa-filen er at det kan settes globale variable for applikasjonen slik at det f. eks. er mulig å vite hvilke asp-sider en bruker har vært innom for den aktuelle applikasjonen, og om det eventuelt har blitt gjort noen valg eller satt noen parametre som kan lagres i globale variable.

Global.asa starter tjenesten Application_OnStart når en bruker aksesserer en asp-fil (men ikke en HTML-fil) innen den virtuelle katalogen som utgjør applikasjonen. Når denne tjenesten blir kalt, kan f. eks. globale variable settes. En slik type variable kan være en variabel som inkrementeres hver gang en web-applikasjon blir aksessert. Applicaton_OnStart startes også ved oppstart av selve web-applikasjonen, f. eks. som et resultat av rebooting av en IIS-server.

Når brukeren som aksesserte en ASP-side også browser en .asp-fil i web-applikasjonen, blir Session_OnStart kalt. Med ASP blir hver bruker av web-applikasjonen tildelt en egen sesjon. Når en bruker aksesserer sider i web-applikasjonen, blir status for det enkelte brukeren vedlikeholdt v.h.a. cookies, som er små filer som lagres på brukerens lokale

maskin. Denne slettes hvis brukeren kobler seg fra web. For at web-applikasjonen skal fungere som den er ment, må brukeren akseptere bruk av cookies.

Session_OnStart kan benyttes for å initialisere sesjons-variable for en individuell bruker, og gi denne f. eks. individuell aksess til Exchange. Oversending av variable fra brukeren kan gjøres v.h.a. overføring via HTTP, enten med get eller post metoden. Get legger til informasjon til url-navnet, post gjemmer informasjonen nede i HTTP-headeren.

Session_OnEnd kjøres hvis ikke brukeren har aksessert en webside innen en gitt tidsperiode, eller ved å kalle Abandon metoden for Session-objektet. Application_OnEnd kjøres ved stans av web-serveren, eller ved å bruke Unload-knappen i IIS-administrator programmet. For å ta vare på globale variable må disse lagres til et fysisk medium.

ASP gir mulighet for å aksessere databaser som støtter en ODBC (Open DataBase Connectivety)-forbindelse. Med et ASP-script kan det sendes over SQL-søkestrenger, og resultatet kan leses av web-applikasjonen. Resultatet kan deretter sendes på ønsket form til brukeren som en HTML-side.

8.2.3 Mulighet for arbeids- informasjonsorganisering med CDO og ASP

ASP gir sammen med CDO mulighet for å aksessere Exchange. En web-applikasjon kan ved å bruke ulike sesjoner gi brukerne individuell aksess til Exchange. Parametre for hva slags informasjon brukeren ønsker kan sendes til web-applikasjonen v.h.a. get eller post metoden. Det er også mulig å bruke disse parameterne ved søk i andre databaser. Resultatet av den ønskede informasjonen basert på parameterne kan visualiseres ved å sende en HTML-side tilbake til brukeren med resultatet.

Dette kan f. eks. være informasjon om antall mail i brukerens mailboks. Parameterne som oversendes til web-applikasjonen kan spesifisere f. eks. hvilke mail som skal vises basert på avsender. Det er også mulig å spesifisere andre måter å filtrere eller søke blant dataelementene i Exchange avhengig av om CDO gir mulighet for å aksessere ulike property'er eller metainformasjon for det aktuelle dataelementet.

8.3 Office 2000 – Word

I dette kapitlet vil kort Word 2000 bli presentert. Dette programmet forutsettes så kjent at det ikke vil bli beskrevet i detalj. Imidlertid vil parametere for word-filer bli presentert, fordi disse bil bli benyttet for implementering av parameterne som ble funnet i kapittel 7. Til slutt vil det kort bli diskutert hvilke muligheter som finnes for informasjonsorganisering.

8.3.1 Parametre

Word 2000-filer har en del parametre som beskriver dataelementet. Denne metainformasjonen er listet opp nedenfor for å vise hvilke parametre det velges blant i

kapittel 10.1. Noen parametre vil bli litt nærmere forklart fordi det vil bli referert til dette i kapittel 10.1. Parameterne vil m.a.o. her bli presentert, men ikke diskutert.

Parametre satt av systemet:

Paragraphs
Pages
Characters
Characters with spaces
Lines
Words
CreationTime
Format
Author
Company

Author settes automatisk lik den som Word 2000 er registrert på. og Company settes lik den organisasjonen Word 2000 er registrert på. CreationTime er et tidsstempel som settes automatisk. Format er betegnelsen på hvilken type dataelement det er. Dette kan for Word 2000 være flere typer, fordi programmet gjenkjenner ulke formater som kan benyttes i programmet. Eksempler kan være en ASCII-tekst fil eller en Word 2000-fil. Imidlertid vil det som nevnt innledningsvis her kun bli vist parametre tilhørende rene Word 2000-filer.

Parametre som kan settes av bruker:

Password
Title
Subject
Manager
Category
Keywords
Comments
Contents
Filename

Filnavnet må settes, men de andre er det ikke nødvendig å sette. Det finnes i tillegg en del annen metainformasjon som kan settes. Denne fremkommer i dialogboksen som tilleggs-informasjon i forhold til den metainformasjonen som er vist over. Ved sammenligning med objektene for Outlook 2000-objektmodellen ble denne påstanden støttet ved at det som er kalt tilleggsinformasjon ikke var tilgjengelig i disse objektene.

Derfor blir de parameterne som er vist hittil ansett som kjernen i metainformasjonen som kan settes for en Word-fil. Siden tilleggsinformasjonen ikke fantes for Outlook 2000-objektene, var det heller ikke mulig å benytte samme parametere ved implementering av modellen spesifisert i kapittel 7. Derfor er det som er blitt vurdert som tilleggsinformasjon kun listet opp nedenfor uten forklaring. De blir også vist for å understreke at å sette metainformasjon i Word 2000 kan kreve en del innsats fra brukeren. Ved innføring av kontekstbasert arbeids- og informasjonsorganisering er derfor målet å minimalisere innsats for å spesifisere parametre for gjenfinning av informasjon.

Checked by, Client, Date completed, Department, Destination, Disposition, Division, Document number, Editor, Forward to, Group, Language, Mailstop, Matter, Office, Owner, Project, Publisher, Purpose, Received from, Recorded by, Recorded date, Reference, Source, Status, Telephone number, Typist.

8.3.2 Mulighet for arbeids- informasjonsorganisering

Word 2000 tilbyr få muligheter for arbeids- og informasjonsorganisering i den formen som programmet fremstår uten tilleggsprogrammer som f. eks. Document Management Extensions. Mulighetene begrenser seg til lagring i kataloger som er hierarkisk strukturert, og visualisering av denne strukturen i form av en trestruktur. Dialogboksen for lagring og åpning av filer gir mulighet for å se denne trestrukturen.

Andre muligheter i denne dialogboksen er bruk av hurtigknapp for tilgang til personlig katalog på maskinen som benyttes, og mulighet for å legge kataloger til et valg kalt favourites, som inneholder en snarvei til de mest brukte katalogene (som må velges av bruker). Utover dette kan det velges hva slags format dataelementene (filene) som vises i dialogboksen for lagring og åpning av filer skal ha.

8.4 Office 2000 – Outlook

I dette kapitlet vil kort Outlook 2000 bli presentert. Først vil to objekter som er representative fra Outlook 2000-objektmodellen bli valgt ut. Deretter vil parametre for disse objektene, som kan benyttes ved diskusjon av hvordan parameterne fra kapittel 7 kan implementeres i Office 2000 og Exchange, bli presentert. Til slutt vil det kort bli diskutert hvilke muligheter som finnes for informasjonsorganisering, og disse mulighetene vil bli vist.

8.4.1 Parametre

Outlook 2000 objekt-modellen inneholder følgende objekter: Contact, Appointment, Meeting, Task, TaskRequest, TaskRequestAccept, TaskRequestDecline, TaskRequestUpdate Report, Mail, Journal, Post, Document,Distributionlist, Note og Remote. De fleste av disse har mange sammenfallende parametre eller property'er.

Derfor ble det ikke funnet hensiktsmessig å behandle samtlige, fordi den diskusjonen som vil bli foretatt i kapittel 10.1, om hvordan parameterne fra kapittel 7 kan implementeres, vil uten store forandringer kunne adapteres til disse andre objektene. Da vil de mulighetene som blir funnet for de to objektene som velges her også kunne utføres for disse andre.

Objektene Mail og Document ble funnet representative fordi begge benyttes ofte i Outlook, og fordi Word var valgt var det naturlig å behandle Document-objektet. Nedenfor vil de fleste parameterne for disse to objektene bli presentert, men ikke diskutert før i kapittel 10.1. Først vil Mail-objektet bli presentert, og deretter Document-objektet.

Outlook Mail-objekt

Parametre satt av systemet:

ExpiryTime
CreationTime
LastModificationTime
RecievedTime
SenderName
EntryId

EntryId er unik for hvert dataelement i en folder, og derfor en parameter som kan skille elementer med samme navn. Alle parametre satt av systemet er ikke vist, da disse ble ansett som mest relevante.

Parametre som kan settes av bruker:

AlternateRecipientAllowed
AutoForward
BCC
BillingInformation
Body
Categories
CC
Companies
DeferredDeliveryTime
DeleteAfterSubmit
FlagDueBy
FlagRequest
FlagStatus
HTMLBody
Importance
MessageClass
Mileage
NoAging
OriginationDeliveryReportRequested
ReadReceiptRequested
ReceiptReassignmentProhibited
ReminderOverrideDefault
ReminderPlaySound
ReminderSet
ReminderSoundFile
ReminderTime
RemoteStatus
Sensitivity
SentOnBehalfOfName
Subject
Submitted
To
UnRead
VotingOptions

VotingResponse

Dette er samtlige parametre som kan settes av bruker i dette objektet.

Outlook Document-objekt

Parametre satt av systemet:

CreationTime
LastModificationTime
Saved
EntryId

Alle parametre satt av systemet er ikke vist, da disse ble ansett som mest relevante. Saved er tatt med for å vise at det er mulig å bruke dette objektet for tilgangskontroll slik som i Document Mangement Extensions.

Parametre som kan settes av bruker:

BillingInformation
Body
Categories
Companies
Importance
MessageClass
Mileage
NoAging
Sensitivity
Subject
Unread

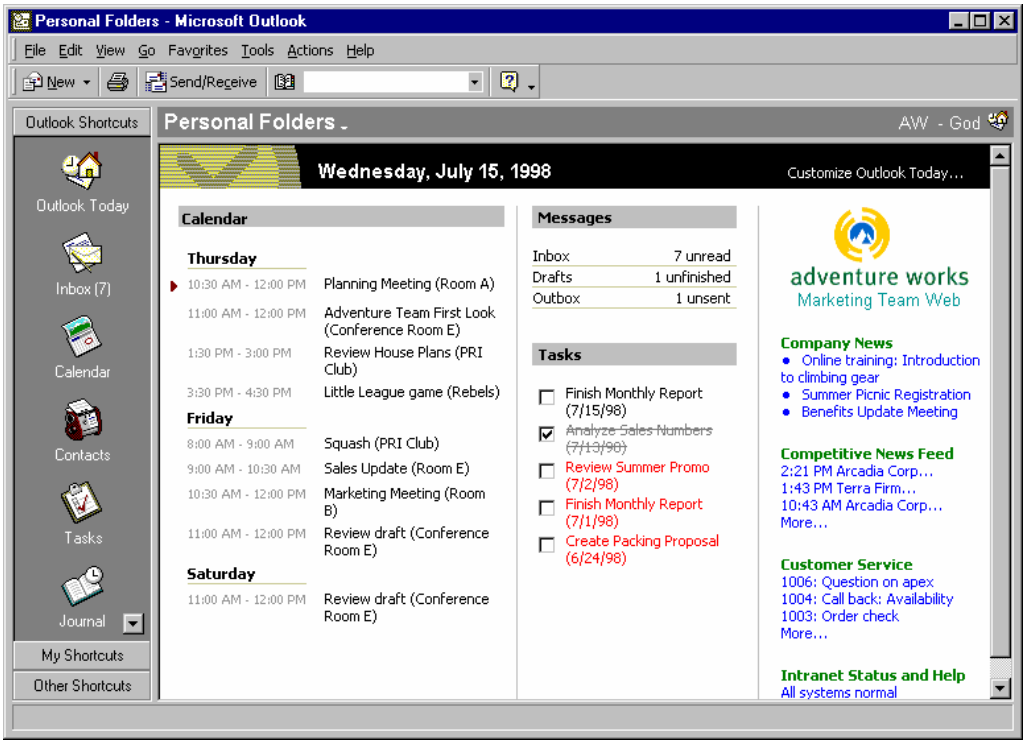
Dette er samtlige parametre som kan settes av bruker i dette objektet.

8.4.2 Mulighet for arbeids- informasjonsorganisering

I Outlook 2000 finnes det flere metoder for å organisere informasjon. Disse er funksjonene find, organize og Outlook Today. Event Scripting Agent kan også settes opp i Outlook 2000, men er allerede beskrevet i kapittel 8.2.2.

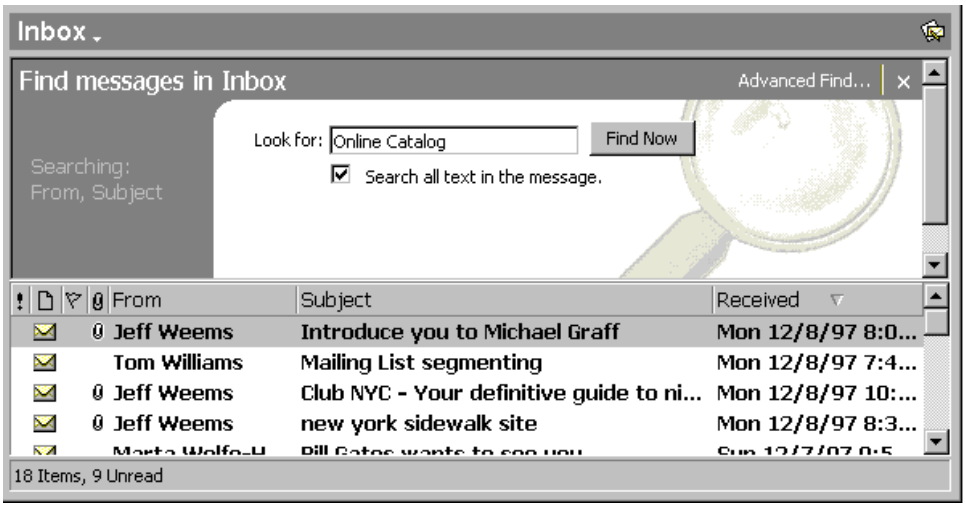
Outlook Today benytter en mulighet kalt Folder Homepage. Folder Homepage vil si å assosiere en hjemmeside med en folder. Ved å klikke på folderen kommer ikke innholdet i folderen opp, men en hjemmeside som kan lastes enten lokalt eller fra Internett, forutsatt at maskinen er koblet opp mot dette. Outlook Today gir oversikt over ting som totalt antall mail, antall uleste mail, aktiviteter for den aktuelle dagen og andre ting som eventuelt kan programmeres inn i Outlook Today.

Dette gir mulighet for et grensesnitt som raskt kan gi en oversikt over informasjon som befinner seg på Exchange, og hva slags informasjon som skal vises kan ved eventuelt å lage en dedikert Folder Homepage for aktuell informasjon bestemmes av brukeren eller en systemadministrator. Outlook Today er vist i figur 8.1.



Figur 8.1 Outlook Today i Outlook 2000

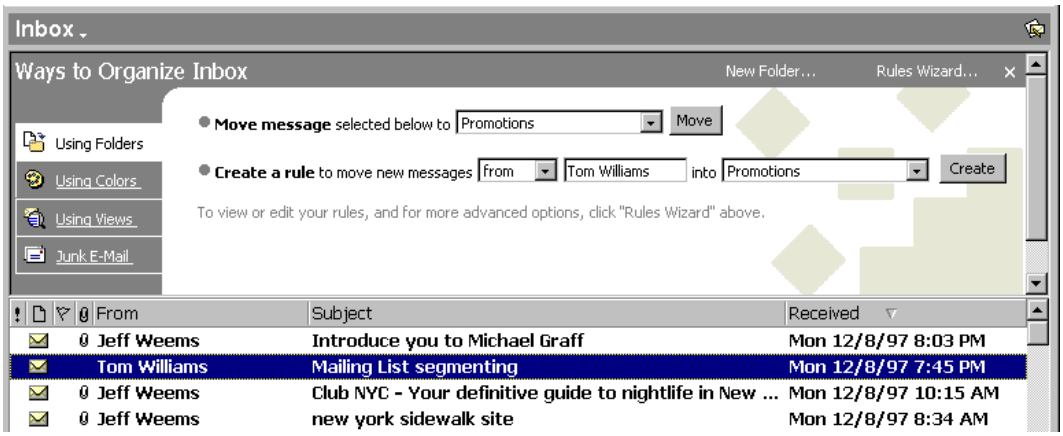
En annen mulighet for å finne informasjon er å benytte funksjonen Find. Brukergrensesnittet for denne funksjonen er vist i figur 8.2. Find brukes i basis-form for søk etter mail, og sjekker da etter et ord i feltene From, Subject eller To. Det er også mulig å gjøre avanserte søk, og det kan da søkes på f. eks. alle felter for Mail- og Document-objekter.



Figur 8.2 Find-funksjonen i Outlook 2000

En tredje mulighet for arbeids- og informasjonsorganisering er funksjonen Organize. Organize benyttes for enten flytte dataelementer som er merket, som vist i figur 8.3, eller for å lage regler for behandling av dataelementer. En enkel regel kan være at mail fra bestemte personer skal merkes ved at teksten som vises ved listing av mail tilhørende den aktuelle personen blir merket med en spesiell farge.

En nyttig mulighet med Organize som vil bli benyttet i kapittel 10, er å lage regler for hvordan nye meldinger i en folder skal behandles. Alle mulighetene vil ikke bli beskrevet her, men disse er generelt at det er mulig å sjekke ulike parametre ved meldingen (f. eks. et Mail- eller Document-objekt) mot valgte kriterier, og deretter utføre en handling på meldingen. Dette kan være å flytte den til en annen folder eller å slette denne.



Figur 8.3 Funksjonen Organize i Outlook 2000.

For å sikre at ulike properties blir fylt ut f. eks. ved sending av mail, er det mulig å benytte forms. Forms er en dialogboks som kan konstrueres for å dekke ulike behov og situasjoner. I formen er det mulig å definere ulike felter som må fylles ut av brukeren. Et eksempel kan være ved sending av mail. En form kan lages for tvinge brukeren til å fylle ut felter som setter f. eks. property'ene Category og SentOnBehalfOfName.

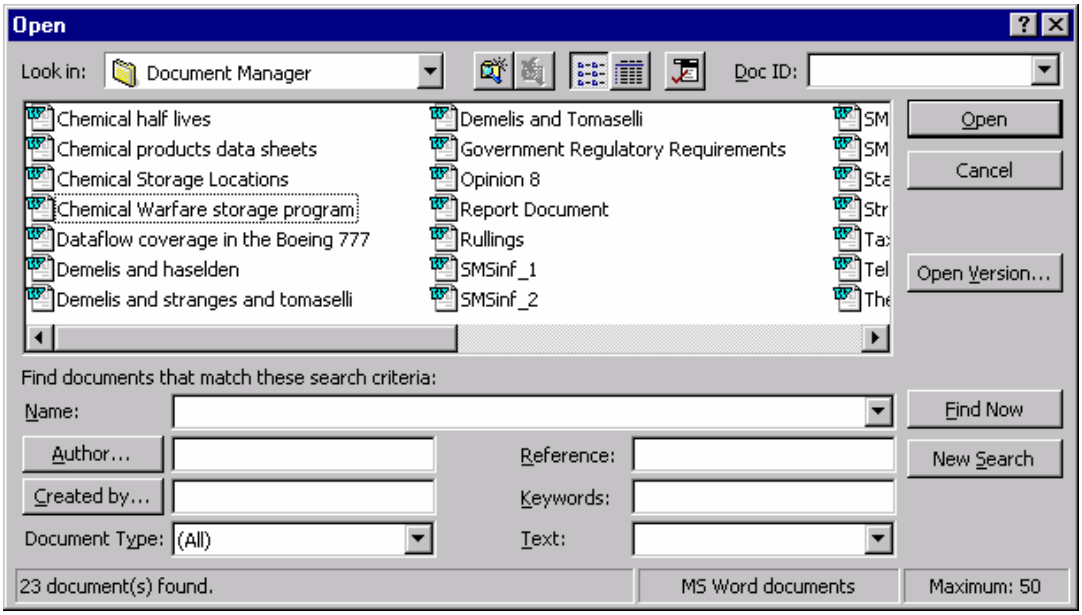
8.5 Document Management Extensions

Document Management Extensions (DME) fra 80-20 Software [38] er her kort beskrevet for å illustrere at det er mulig å koble Word 2000 mot Exchange. DME vil ikke bli brukt i løsningen utover det at det kan eksemplifisere hvordan Word 2000 og Exchange kan kobles sammen. Det ble ikke utviklet en egen prototyp, og derfor ble det ansett som nødvendig å illustrere hvordan koblingen kan utføres v.h.a. et praktisk eksempel.

8.5.1 Kort beskrivelse og diskusjon

Document Management Extensions er et tilleggsverktøy for Word for å tilby mulighet for Document Management. Løsningen er basert på å koble Word opp mot Exchange databasen. DME benytter ikke Document-objektet fra Outlook objekt-modellen. Isteden settes visse parametre ved lagring, og disse blir lagret sammen med Mail-objektet som blir lagret i Exchange. Word-dokumentet blir lagret som et eget vedlegg.

Word blir koblet mot Exchange databasen både ved at dokumentene lagres der, og at grensesnittet for lagring og åpning av dokumenter blir endret ved innstallering av DME. Det blir etter installasjon mulig å søke i selve Exchange databasen. Grensesnittet får også en del nye parametre som må settes ved lagring, og som kan brukes ved søk. Det blir med dette grensesnittet nemlig mulig å søke direkte i Exchange databasen, og basert på de kriteriene som fremgår av grensesnittet. Dette gjøres ved å sette parametre og benytte search-knappen. Grensesnittet er vist i figur 8.4, og er hentet fra hjemmesidene til 80-20.com [38]. Dette er grensesnittet for Office 97. Grensesnittet for Office 2000 var ikke lagt ut som eget bilde.



Figur 8.4 Grensesnittet ved lagring og åpning av filer i Word etter innstallasjon av DME

DME kan ikke benyttes som en del av løsningen for denne oppgaven, fordi DME ikke benytter Document-objektet, som vil bli benyttet for implementering av parameterne diskutert i kapittel 7. Imidlertid viser DME at det er mulig å koble Word opp mot Exchange, og er et eksempel på hvordan parametre kan settes.

Undersøkelsen av denne teknologien gav derfor idéer til hvordan setting av ulike parametre kan implementeres i et grensesnitt.

9 Utprøving av Office 2000 og Exchange

For å få større forståelse for CDO, ASP og de mulighetene Office 2000 og Exchange gav for kontekstbasert arbeids- og informasjonsorganisering, ble det foretatt noe praktisk utprøving av teknologien. Det ble satt opp en egen server, Diplom, hvor det ble installert serverne Exchange, NT og IIS. Dette gav mulighet for å få erfaring med oppsett og tildels drift av Exchange. Ved et slikt ”hands-on” arbeide økte forståelsen for de ulike delene av Exchange, og gav innsikt i teknologien. Det ble også prøvd ut to muligheter med teknologien, Event Scripting Agent og webgrensesnitt mot Exchange. Arbeidet med disse er beskrevet nedenfor.

9.1 Event Scripting Agent

For å forstå hvordan Event Scripting Agent (ESA) fungerte ble det funnet et script på hjemmesidene til CDOLive [43]. Dette ble testet for å øke forståelsen for bruksområdet for ESA, og få idéer til hvordan dette kunne benyttes ved kontekstbasert arbeids- og informasjonsorganisering. Nedenfor er scriptet kort presentert, og det blir trinnvis fortalt hvordan Exchange og Outlook ble satt opp for å kunne kjøre scriptet.

9.1.1 Script

Scriptet er vist i vedlegg 2. Dette scriptet looper igjennom alle mailboksene på Exchange, og skriver ut informasjon om navnet på mailboksen, størrelse på mailboksen, antall mail i mailboksen, antall uleste mail i mailboksen (Inbox) og e-mail-adressen til en Excel-fil.

Det blir benyttet CDO for å aksessere mailboksene på Exchange serveren. For å få scriptet til å fungere måtte konstanten g_Const_MBX endres til Service for at scriptet skulle sende til riktig mailboks. Scriptet finner selv hvilken Exchange Server som tilhører den påloggede brukeren.

Deretter går scriptet gjennom alle mailboksene den finner i Global Adress List (GAL), og ekstraherer den informasjonen som er nevnt i begynnelsen av kapitlet. Feilmeldinger blir skrevet til en egen fil for debugging.

9.1.2 Oppsett og installasjon

Nedenfor blir det trinnvis beskrevet hvordan det ble gått frem for å sette opp Exchange og kjøre scriptet.

- Det ble sjekket at ikke mailbokser eller ”public folder” var skjult for Global Adress List (GAL). Ellers ville ikke scriptet fungere fordi det er avhengig av å kunne aksessere mailboksene og ”public folder”.
- I MS Exchange Administrator ble en ny mailkonto kalt ”Service” opprettet. Denne ble assosiert med MS Exchange Server Service kontoen, som på Diplom-serveren

ble kalt Service. Dette ble gjort fordi bare denne kontoen har rettigheter til å aksessere alle mailboksene.

- Under ”Permissions” for mailboksen ble diplom-kontoen lagt til for at denne kontoen siden kunne brukes for å åpne mailboksen og installere scriptet siden.
- Deretter ble det i MS Exchange Administrator lagt til ”Service” til *Folders/System Folders/Events Root/EventConfig_Diplom* med ”Owner” rettigheter for å sikre at denne kan lage og modifisere MS Exchange Server Scripting and Routing scripts.
- Deretter ble det på egen maskin laget en ny MAPI-profil i Outlook kalt ”Service”, og denne ble assosiert med mailboksen med samme navn.
- For å installere scriptet ble Outlook startet og det ble logget på som ”Service”.
- For å få mulighet til å velge ”Agents” fra property-dialogboksen for Inbox, måtte man gå inn i Tools-menyen, finne Options/Other/Advanced Options. Der ble det i Add-in Manager dialogboksen valgt ”Server Scripting”. Gikk deretter ut fra menyen.
- Høyreklikket deretter på Inbox-folderen, og valgte properties. Valgte Agents/New, og laget en ny Agent kalt ”Mailbox Report”. Valgte ”A Scheduled Event Occurs”, valgte så ”Schedule” og satte tiden til hver time.
- Valgte ”Edit Script”, og brukte cut’n’paste for å få inn scriptet fra Notepad. Lagret deretter scriptet og gikk ut av dialogboksen.
- Neste gang mailboksen ”Service” ble sjekket, hadde mail fra ”Service” kommet med filen som attachment og dermed oversikt over de ulike mailboksene på Diplom.

9.2 Webgrensesnitt mot Exchange

I oppgaven ble det pekt på at Internett var den teknologien som egnet seg best for å bygge bro over ulike informasjonsdomener. Derfor var det naturlig å prøve ut webgrensesnitt mot Exchange. Dette ble gjort ved å finne en web-applikasjon på Microsoft sine hjemmesider, og modifisere noe på denne. Uvisst av hvilken grunn er disse filene fjernet fra det området der de ble funnet hos Microsoft, så derfor er det ikke gitt noen referanse.

Nedenfor vil de ulike filene bli kort beskrevet. Deretter vil det bli beskrevet hvordan oppsettet av IIS må være, og hva resultatet av aksessering av web-applikasjonen med en nettleser vil bli.

9.2.1 Filer

Nedenfor er de ulike filene og deres funksjoner kort presentert. Disse filene finnes i sin helhet i vedlegg 1.

Default.htm
Denne filen starter AuthLogon.asp. Filen er tatt med i tilfelle nettleseren ved aksessering av web-applikasjonen ikke gjenkjenner og starter AuthLogon.asp, som er satt som default i IIS administrator-verktøyet (se neste kapittel). Nettleseren vil i et slikt tilfelle lese Default.htm, og derfor er denne tatt med for å gi mulighet for at brukeren kan starte riktig fil (som er AuthLogon.asp).

Global.asa
Denne filen setter noen globale parametre ved oppstart av web-applikasjonen og for hver nye sesjon. Sørger for å rydde opp når en bruker ender en sesjon.

AuthLogon.asp
Denne filen benytter seg av AuthLogin.inc. Kaller kun sistnevnte fil, og benytter denne for autentisering. Etter autentiseringen sørger den for å kalle SendMail.inc og dermed bygge opp HTML-siden for sending av mail.

AuthLogin.inc
Denne filen benyttes av AuthLogon.asp for å autentisere sesjonen og sjekke at brukeren eksisterer. Denne filen kjøres helt til autentiseringen er gjort og en sesjon opprettet.

AuthSendMail.inc
Brukes for å sende mail og sjekke sesjonen som er aktiv. I denne filen ble variabelen bstrServer satt lik Diplom for å muliggjøre sending av mail fra riktig Exchange-server.

SendMail.inc
Denne filen kalles etter at innloggingen er gjennomført. Den bygger opp en side for å kunne sende mail. På denne siden er det lagt til kode for setting av Category og lesing av antall mail i mailboksen. Det er benyttet både kall til CDO-objekter og ved å bruke hex-kode. Hex-kode benyttes egentlig ved client-side scripting hvis denne ikke har CDO.dll installert. Teoretisk unødvendig i dette tilfellet, men det var hensiktsmessig å prøve om det virkelig fungerte, noe det også gjorde.

AuthLogoff.asp
Denne filen trigger Session_OnEnd i global.asa, og ender dermed sesjonen for brukeren. Dette gjøres ved å kalle Session-objektets Abandon metode. Deretter starter den login-siden for web-applikasjonen.

9.2.2 Oppsett og resultat

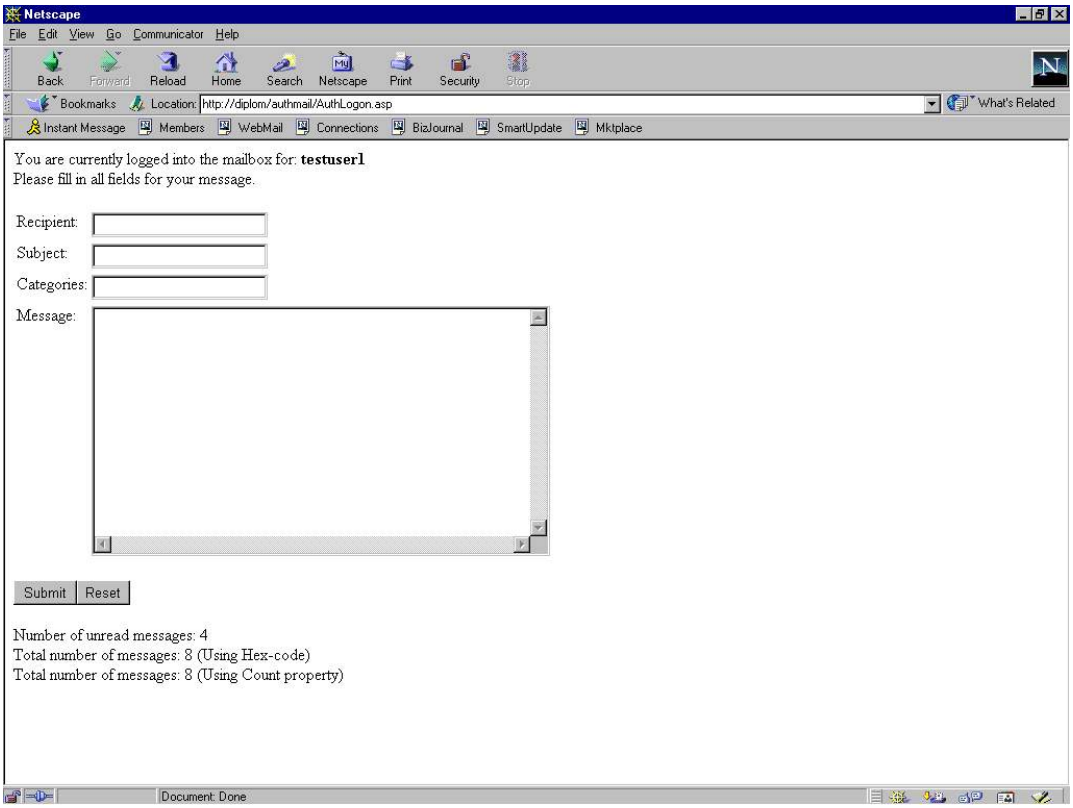
Nedenfor er det vist oppsett av IIS Server for å kunne kjøre web-grensesnitt mot Exchange.

1. Innstallerte IIS 4.0 fra NT 4.0 Option Pack
2. For å få innstallert CDO må enten Exchange Server være på samme maskin som IIS-serveren, eller mail-klienten Outlook. I denne oppgaven ble Exchange og IIS kjørt på samme server.
3. Åpnet UserManager
4. Under menye “Policy”, ble “User Rights” valgt.
5. I dropdown-boksen “Right” ble “log on Locally” valgt for at brukeren skal måtte logge seg på den maskinen som kjører IIS ved aksessering av web-siden.

6. Testbrukerne ble lagt til slik at de kunne brukes til å aksessere web-siden.
7. Asp- og include-filene ble lagt under en mappe på IIS-serveren.
8. Åpnet Microsoft Management Console.
9. Høyreklikket på “Default Web Site” og valgte “New… Virtual Directory” for lage et virtuelt område som enkelt kunne aksesseres av brukeren. Dette fremkommer i URL-navnet som en mappe under web-roten. Ved aksess av dette virtuelle området blir brukeren koblet mot det fysiske området og koden det blir kjørt. Brukeren forholder seg imidlertid kun til det virtuelle området.
10. Valgte navnet AuthMail som navn på den nye web’en og klikket “Next”.
11. Valgte mappen med asp- og include-filene og trykket “Next”.
12. Valgte mulighet for “Read”, “Script” og “Execute” og klikket deretter “Finish”. Dette for å gi brukeren mulighet til å lese og kjøre skript.
13. Høyreklikket på web’en AuthMail og valgte “Properties” og deretter “Virtual Directory”.
14. I “Application Setting”-rammen ble “Create” trykket. Dette ble gjort for å definere det virtuelle området som en web-applikasjon, som er nødvendig for å kunne kjøre filen global.asa korrekt.
15. Valgte “Directory Security” og klikket på “Edit”-knappen under “Anonymous Access and Authentication Control”.
16. De-select "Allow Anonymous" and "NT Challenge Response".. Select "Basic Authentication". This will force the user to supply logon credentials to IIS, which will use those credentials to authenticate the user to Exchange Server.
17. Valgte bort “Allow Anonymous” og “NT Challenge Response”. Valgte “Basic Authentication”. Dette vil tvinge brukeren til å logge seg på IIS, som igjen vil danne grunnlag for pålogging til Exchange Server. Dermed er oppsettet av IIS klart.

I figur 9.1 vises resultatet av pålogging av brukeren testuser1 til webapplikasjonen. Først ble det vist en dialogboks for innlogging. Det ble logget inn med brukernavn=testuser1 og passord=t1. URL-adressen til web-applikasjonen var http://diplom/authmail .

Ved å skrive inn mottaker lik testuser1 og sende mail v.h.a. web-applikasjonen ble det observert at antall mail som ble telt opp økte ved oppdatering av siden. Dette viser at web-applikasjonen fungerte, og at det med ASP og CDO er mulig å aksessere Exchange fra Internett.



Figur 9.1 Resultat av aksessering av web-applikasjonen med nettleser etter innlogging

9.3 Erfaringer

Utprøving av Event Scripting Agent og webgrensesnitt mot Exchange viste at ASP, CDO og VBScript egner seg godt for å ekstrahere data fra Exchange. Studiet av CDO og ASP har vist at det er mulig å manipulere dataelementer i Exchange databasen, og det er mulig å utføre handlinger på dataelementene avhengig av en hendelse eller timer i en spesiell folder.

Utprøvingen gav god innsikt i teknologien som ikke kunne blitt tilegnet kun ved teoretisk studie. Dermed er det et godt grunnlag for å se på hvordan implementering av kontekstbasert arbeids- og informasjonsorganisering kan gjøres i Exchange.

10 Mulig implementering i Office 2000 og Exchange

I dette kapitlet vil det bli diskutert og pekt på muligheter for hvordan Office 2000 og Exchange kan benyttes for implementering av kontekstbasert arbeids- og informasjons-organisering. Som det er forklart i kapittel 8 vil bare Word og Outlook bli brukt av programmene som finnes i Office 2000. Document Management Extensions vil bli brukt som eksempel på hvordan Word 2000 kan kobles mot Exchange.

Det er tre dataelementer eller objekter som vil bli behandlet. Vanlige Word-filer og Mail- og Document-objektene fra objektmodellen tilhørende Outlook 2000. De andre objektene i objektmodellen har som det er påpekt i kapittel 8.4 tilsvarende properties, og vil derfor kunne benyttes sammen med parameterne funnet i kapittel 7 tilsvarende slik de er vist benyttet sammen med de to valgte objektene i kapittel 10.1.

Vanlige Word-filer kan ikke manipuleres på samme måte som hvis Word bruker Exchange for lagring av dokumentene. Uten bruk av Exchange må det lages egne programmer for tilgang og søk basert på ulike properties. Dette defineres til å ligge utenfor oppgaven, da Office 2000 og Exchange skulle benyttes. Fordi Word-filer kan legges ved som attachment til mail sendt v.h.a. Exchange, og fordi det kan om ønskelig bli utviklet programvare som kan søke gjennom Word-filer basert på ulike properties eller utvikles objekter som gjør at Word-filer kan lagres i Exchange med tilgang til alle property‘ene, er det tatt med hvordan parameterne funnet i kapittel 7 kan implementeres i vanlige Word-filer.

Ved videre diskusjon av muligheter for kontekstbasert arbeids- og informasjonsorganisering i Office 2000 og Exchange er det kun tatt utgangspunkt i de to objektene som er valgt fra Outlook 2000 objektmodellen. Disse vil være representative for hvordan ulike objekter kan benyttes v.h.a. de mulighetene og løsningene som diskuteres i dette kapitlet.

10.1 Implementering av parametre

Modellen funnet i kapittel 7 viste hvilke parametre som er nødvendige ved kontekstbasert arbeids- og informasjonsorganisering. Utfordringen nå er da å implementere denne modellen i Office 2000 og Exchange med de mulighetene som finnes i denne teknologien. Nedenfor vil de valgte parameterne fra modellen bli diskutert både i forhold til Word og Outlook. Som det er fortalt i kapittel 8, vil objektene Mail og Document fra Outlook 2000-modellen bli brukt for lagring i Exchange.

Vanlige dokumentfiler i Word har noen andre properties enn Document-objekter fra Outlook sin objekt-modellen. Dette gir litt andre muligheter. Derfor er det ikke mulig å komme med et identisk forslag for setting av parametre for Document-objekter og vanlige Word-filer. Dette vil bli diskutert.

Nedenfor er de ulike parameterne diskutert i forhold til hvordan de kan implementeres i Office 2000 og Exchange

Kategori

I alle tre objektene er det mulighet for å sette en property kalt kategori (Category). Denne propery'en kan inneholde flere kategorier. Property'en kategori for de ulike objektenen kan derfor uten videre benyttes for parameteren kategori fra modellen.

Rolle

Ingen av de tre objektene har en property kalt rolle. Parameteren kan derfor ikke uten videre implementeres i Office 2000 og Exchange. Som det er nevnt ovenfor er det imidlertid mulig å definere flere ulike kategorier i property'en kategori. Dermed kan property'en kategori benyttes for å implementere parameteren rolle. Dette kan gjøres ved at den første kategorien i property'en settes lik parameteren kategori, mens den neste settes lik parametern rolle. Ved lagring og søk må derfor denne formen brukes av alle for å oppnå konsistens i lagringen. Dette kan oppnåes ved å bruke standardiserte metoder for lagring og informasjonssøk. Dette er nærmere beskrevet i kapittel 10.2

Format

Hvilket format en fil eller et objekt har vil fremkomme automatisk av programmet som benytter f. eks. en Word-fil. Hvordan dataelementer skal lagres er standardisert i forhold til hvilket operativsystemet som benyttes, i dette tilfellet en versjon av MS Windows. Innholdet i dataelementene er standardisert i forhold til hvilket verktøy som skal utføre operasjoner på disse. Derfor er parameteren format mulig å implementere i Office 2000 og Exchange fordi dataelementer brukt sammen med denne teknologien har en bestemt oppbygning. Ulike formater er bestemt av verktøy og operativsystem, og parameteren kan av denne grunn benyttes som beskrevet i kapittel 7.

Lokasjon

Hvilken lokasjon dataelementen har fremkommer ved søk i f. eks. Exchange. Outlook og Word identifiserer hvor et dataelement befinner seg ved å vise dette som en trestruktur. Denne trestrukturen viser både på hvilken server dataelementet befinner seg, og hvor i den hierarkiske lagringsstrukturen på serveren det befinner seg. Parameteren lokasjon lar seg derfor implementere i Office 2000 og Exchange.

Tid

For alle de tre objektene ble property'en tid automatisk satt ved lagring. Det var litt forskjell i hvilken tid som ble lagret ved flere gangers lagring av samme objekt. For en

vanlig Word-fil lagres tiden som filen ble lagret. Informasjon om når filen første gang ble opprettet er ikke mulig å sette. De andre to objektene, fra Outlook objekt-modellen, får automatisk satt tid for opprettelsen av dataelementet og siste gang objektet er modifisert. Disse property'ene kalles CreationTime og LastModificationTime. Parameteren tid lar seg derfor implementere for alle de tre objektene. Ved at objektene som benyttes i Exchange får satt to ulike typer tidsstempel, åpner det seg som det vil bli vist i neste avsnitt flere muligheter ved setting av revisjon.

Revisjon

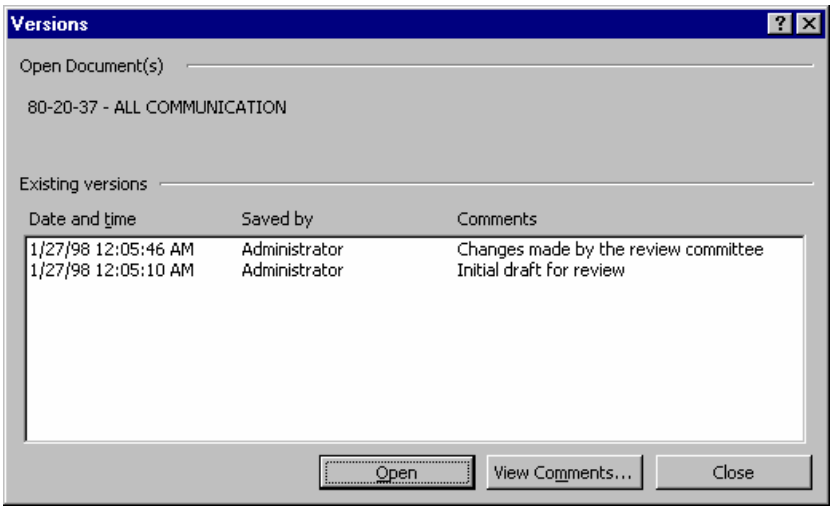
Det finnes ikke en egen property i noen av de tre objektene for setting av parameteren revisjon slik den er beskrevet i modellen i kapittel 7. Derfor må revisjon settes på annen måte. En metode kan være tilsvarende slik verktøyet Document Managament Extensions (DME), beskrevet i kapittel 8.5, har innført versjonskontroll. Dialogboksen for versjonskontroll i DME er vist i figur 10.1. Her fremgår det at tidspunkt for siste lagring utgjør versjonskontrollen. Det er også mulig å lage flere versjoner, men da som egne filer. Dette er i samsvar med modellen fra kapittel 7. Der ble det konkludert med at skulle flere versjoner være tilgjengelige, måtte disse lagres som egne filer.

Ved lagring av vanlige Word-filer vil en metode for å innføre parameteren versjon være å se på tidsstempelet for når filen er lagret. For objektene tilhørende Outlook sin objekt-modell, vil det måtte benyttes tidsstempelet i property'ene CreationTime og LastModificationTime. Imidlertid vil det ikke være mulig å lagre Word-filer med samme navn i samme katalog. Derfor må navnet på disse dataelementene som forskjellig, og dette medfører et problem ved å måtte skille de ulike dokumentene fra hverandre, ikke bare versjoner.

Et annet problem ved bare å skille versjoner på basert på tid, er at hvis ikke alle versjonene lagres er det ikke mulig v.h.a. kun tidsstempelet å bestemme hvilken versjon det er, bare om et dataelement er lagret før eller etter andre. Derfor vil parameteren versjon slik den er skissert i modellen ikke være godt nok implementert ved kun denne metoden. Property'en Category er mulig å benytte. De to første "kategorifeltene" som settes i denne property'en er som et resultat av tidligere diskusjon reservert for parameterne kategori og rolle. Versjon kan derfor lagres som et tredje felt.

Å benytte enda et felt i property'en Category vil ved svært store søk medføre mye prosessering. Nye utgaver av objektene bør derfor inneholde en property for versjon. Imidlertid er det i oppgaven definert at brukerne tilhører relativt små grupper, og informasjonen det jobbes med vil derfor ikke være store informasjonsdatabaser. Teknologien som skulle brukes var også definert på forhånd. En slik innføring av parameteren versjon er derfor akseptabel.

Diskusjonen har så langt pekt på en del problemer og mulige løsninger. For objektene tilhørende Outlook objekt-modellen, vil det være mulig å lagre flere dataelementer med samme navn i samme folder. Dette er mulig fordi parameteren EntryId skiller



Figur 10.1 Dialogboks for versjonskontroll i Document Management Extensions

dataelementene som unike. Dermed kan dataelementer med samme navn skilles ved bruk av property'ene CreationTime eller LastModificationTime og setting av det tredje feltet i Category lik et versjonsnummer eller eventuelt ferdig versjon. Dermed er implementering av parameteren versjon muliggjort.

Word-filer kan benytte de samme property'ene, men må i tillegg ta hensyn til at det ikke er tillatt med like navn på dataelementer (filnavn) i samme katalog. Hvis flere versjoner skal lagres, må derfor en spesiell katalog dedikeres til det aktuelle dokumentet. Hvis nyere versjoner av dataelementene lagres "oppå" de gamle, kan flere ulike dokumenter befinne seg i samme katalog.

Opprettet av

For en vanlig Word-fil lagres automatisk hvem som har lagret filen. Dette objektet har også en property author. Denne settes imidlertid lik den brukeren selve programmet Word er registrert på. Derfor kan ikke author brukes for å implementere parameteren opprettet av. Parameteren må for dette objektet implementeres ved å benytte property'en for hvem som har opprettet filen. Parameteren settes automatisk, i samsvar med modellen i kapittel 7.

Mail-objektet fra Outlook objekt-modellen får også satt automatisk hvem som opprettet eller sendte mailen. Dette vil være samme bruker som systemet har registrert som pålogget. Property'en som benyttes til dette er SenderName. Det er også mulig å sette en property SentOnBehalfOfName. Denne vil ikke bli brukt som del av illustrasjonen, fordi den ikke forteller hvem som opprettet dataelementet og fordi den kan settes automatisk vil ikke kvaliteten ved denne property'en være god nok.

Objektet Document har ikke mulighet for automatisk å sette parameteren opprettet av. Det finnes heller ikke en egen property for å knytte et navn til dataelementet. Dette må

kunne sees som en svakhet ved objektet. For å omgå dette problemet ble det vurdert som mest hensiktsmessig å benytte et fjerde felt i property'en Category til parameteren opprettet av. Dette er ingen idéell løsning, men basert på de mulighetene som fantes, ble den vurdert som den beste.

Diskusjonen har vist at parameteren opprettet av kan automatisk settes i vanlige Word-filer og mail-objektet. For Document-objektet må det benyttes et felt i property'en Category.

Revidert av

Ingen av de tre objektene har en egen property som kan brukes for å angi denne parameteren. Det kunne vært mulig å implementere også denne parameteren i property'en Category, men dette ville ikke vært hensiktsmessig m.t.p. kompleksitet. Derfor er det mest hensiktsmessig å implementere parameteren revidert av v.h.a. samme property som er valgt for de ulike objektene i diskusjonen om parameteren opprettet av.

For vanlige Word-filer vil den som lagrer ulike versjoner automatisk bli registrert som eier av filen. Mail-objektet blir automatisk påført hvem som er avsender. For Document-objektet vil det også for denne parameteren være nødvendig å bruke property'en Category. For å implementere parameteren revidert av, er det som det fremgår ovenfor valgt å benytte de samme property'ene som for parameteren opprettet av. Dette gjør at det etter lagring av revisjoner ikke er mulig å bestemme hvem som opprettet dataelementet. Dette er ikke idéelt, men med tilgjengelige muligheter i teknologien som skal benyttes, ble dette vurdert som den beste løsningen.

Organisasjon

Alle tre objekter har mulighet for å sette en property kalt Company. Denne kan derfor benyttes for å implementere parameteren organisasjon. I Word og Outlook settes denne automatisk lik den organisasjonen programmet er registrert på, men det er mulig å endre dette. Dette er i samsvar med modellen i kapittel 7.

Navn på dataelement

For vanlige Word-filer lar dette seg enkelt implementere fordi dataelementene lagres som separate filer med et navn som spesifiseres av brukeren. For mail-objektet finnes det ikke en property tilsvarende navnet på filen, men property'en Subject er egnet for å navngi dataelementet. Subject inneholder som oftest informasjon om hva temaet for mailen er, og er derfor også passende for å implementere parameteren Navn på dataelement, fordi denne property'en vil være beskrivende for dataelementet. For Document-objektet vil også Subject være best egnet for å implementere parameteren Navn på dataelement.

10.2 Identifisering og bruk av ulike kontekster

I dette kapitlet vil det bli diskutert hvordan systemet kan identifisere ulike kontekster, og hvordan dette kan utnyttes ved arbeide med informasjon. Dette vil gi grunnlag for ytterligere diskusjon av hvordan Office 2000 og Exchange kan benyttes for kontekstbasert arbeids- og informasjonsorganisering i påfølgende kapitler.

10.2.1 Identifisering av kontekster

Implementeringen av de ulike parameterne gir mulighet for å identifisere ulike kontekster. Slik implementeringen er gjort her vil det være mulig å jobbe mot ulike personer, roller eller kategorier (prosjekt/fag e.l.), informasjonsdomener eller fysiske lokasjoner, versjoner basert på tid eller status (versjonsnr./ferdig), typer dataelementer eller organisasjoner/selskaper.

Ved at man jobber mot en av disse mulighetene, vil man nødvendigvis befinne seg i ulike kontekster. Dette er diskutert spesielt i kapittel 5 og 7. Dermed vil systemet ved å benytte de parameterne som er beskrevet i forrige kapittel kunne kategorisere brukeren til en spesiell kontekst. Et eksempel kan være en student som jobber mot en folder med dokumenter tilhørende et spesielt fag.

Dataelementene som befinner seg i denne folderen vil, forutsatt at parameterne er satt, kunne identifiseres til ulike kontekster. Dette er mulig fordi det er mulig å identifisere f. eks. rolle og kategori. Disse parameterne kan leses når brukeren jobber med dataelementene. Som det er vist i kapittel 8.1.4, kan et script kjøres hvis et dataelement blir lagt til, endres eller slettes fra en folder.

Ved å kjøre et script når det jobbes med dataelementer i en folder kan ulike parametre finnes. Utprøving av CDO og ASP i kapittel 9 har vist at det er mulig å aksessere Outlook-objektene fra et script. For å finne parameterne må et slikt script inneholde kode som aksesserer et objekt i folderen. Dette kan gjøres enkelt ved å anta at alle dataelementene i folderen er relaterte, og kun sjekke det første. Det kan også gjøres litt mer avansert. Ved å sjekke parameterne opprettet av og redigert av kan det finnes hvilke som eventuelt brukeren som er logget på maskinen har opprettet.

Ved å se på hvordan parameterne for et slik dataelement er satt, kan f. eks. rolle fastsettes mer sikkert ved å se på hvilken rolle dataelementer lagrer av den aktuelle brukeren er satt til. Dette gir to ulike grader av sikkerhet ved identifisering av kontekst. Et script bør kunne kjøre begge testene. Først sjekkes det om det finnes dataelementer som er opprettet eller redigert av brukeren som er pålogget. Om dette ikke lykkes må det sjekkes hva parameterne for det første dataelementene er. Med den siste testen vil det imidlertid ikke være mulig å bestemme rolle, fordi den som lagret dataelementet sist kan ha hatt en annen rolle.

En svakhet med Event Scripting Agent er at den ikke kan kjøres ved lesing av dataelementer. Det er heller ikke mulig å definere selve folderen til en spesiell arbeidskontekst ved å ha mulighet for å sette parametre tilsvarende som for mail-objektet. Dette kunne vært ønskelig ved senere versjoner av Outlook, for å kunne klassifisere en brukers kontekst ved å se på hvilken folder denne jobber med.

Av diskusjonen ovenfor er det tydelig at skal konteksten detekteres automatisk, må dette være et resultat av arbeide med et dataelement i folderen. Det finnes to grader av sikkerhet. Mest sikkert kan konteksten settes hvis et dataelement identifiseres som lagret av samme bruker som den som er pålogget maskinen. Da kan roller settes. Hvis ikke kan bare de andre parameterne bestemmes.

Et problem blir ved arbeide med informasjon hvordan systemet skal utnytte at den har identifisert konteksten. Selve parameterne for konteksten kan scriptet legge i en tekstfil som legges i en midlertidig katalog (f. eks. C:\temp). Denne kan ved f. eks. bruk av find-funksjonene i Outlook hentes frem for å sette en del parametre automatisk. Problemet som diskuteres i neste kapittel er i hvilke tilfeller klassifisering av konteksten ansees så sikker at denne kan hentes frem fra temp filen, og i hvilke tilfeller det er nødvendig for brukeren selv å sette parameterne.

10.2.2 Bruk av ulike kontekster

Situasjoner hvor konteksten automatisk kan finnes ble det vist ovenfor var ved arbeide med dataelementer i en folder. Det ble også funnet at det var to grader av klassifisering av kontekst. Disse to vil bli behandlet likt i det videre arbeidet, men med den forskjell at hvis ikke rollen er identifisert, vil ikke denne bli automatisk satt av systemet.

Situasjoner hvor man arbeider med foldere og kan ha nytte av konteksten er ved lagring av ny versjon av et dokument, flytting av virtuelt arbeidsområde til en annen folder eller ved bruk av funksjonen find i Outlook for å finne relatert informasjon. Det er rimelig å anta at hvis man er i en kontekst og skal lagre en ny versjon eller søke etter informasjon, så vil kontekstparameterne være de samme. I neste kapittel vil det bli vist hvordan man praktisk kan sette disse parameterne ved søk og lagring, men anta i det videre problemet ikke er hvordan, men bare om parameterne fra forrige kontekst skal være gjeldende.

For å sikre kvalitet kan problemet løses enkelt ved at brukeren via en dialogboks får spørsmål ved lagring, søk eller bytte av folder om samme kontekst skal være gjeldende. Ved mye bytte av foldere kan imidlertid dette bli oppfattet som unødvendig ved ofte å måtte besvare dette spørsmålet. Derfor vurderes det som en bedre løsning at ved bruk av find eller lagring i samme folder, beholdes parameterne som standardverdier. Ved bytte av folder ansees det som sannsynlig at konteksten også byttes, og derfor vil ikke de lagrede parameterne bli hentet frem i denne situasjonen.

At parameterne beholdes som standardverdier vil si at når grensesnitt for lagring eller søk kommer opp på skjermen, er valgene for disse parameterne allerede satt til siste kontekst basert på den lagrede tekstfilen. Dette vil bli nærmere beskrevet i neste kapittel.

10.2.3 Lagring/ sending av informasjon

For å legge et dokument i en folder, poste en melding eller sende et brev er det som beskrevet i kapittel 8.4.2 mulig å benytte forms. Disse kan utvikles slik at parametre som kan settes manuelt (se tabell 7.1) kan skrives inn, settes lik lagret kontekst eller velges fra en drop-down boks.

Parameterne rolle og kategori er i modellen diskutert i kapittel 7 vurdert som mest hensiktsmessig å velges fra et predefinert utvalg. Dette kan implementer ved at det finnes en database på Exchange serveren som inneholder mulige roller og kategorier. Dette kan implementeres ved å bruke en folder som er skjult for brukerne og legge inn to dokumenter hvor det ene inneholder mulige roller og det andre kategorier.

Utvalget av roller og kategorier kan bestemmes av en systemadministrator. Valgene kan realiseres i en form ved å ha en drop-down boks hvor de fremkommer. En slik form kan eventuelt benyttes ved kun intern mail innad i organisasjonen der alle bruker Outlook 2000 som mail-leser og en Exchange server for mail.

For å lagre informasjon i en folder (eventuelt poste) kan også egne foldere benyttes. Foldere for lagring kan være ganske like de som benyttes for mail m.t.p. setting av parametere for bestemmelse av kontekst. Spesielle funksjoner for mailobjektet eller dokumentobjektet må selvfølgelig også være med, men i denne oppgaven vil fokuset være mot parametre som kan identifisere ulike kontekster.

Ved sending av mail er det ikke alltid senderen vet hvilken rolle mottakeren har i forhold til senderen og den informasjonen senderen sender. Da vil det være vanskelig for senderen å sette rolle for mottaker. For å omgå dette kan formen for mail utelate mulighet for å sette rolle. Hvis det er kritisk at mottaker får mail basert på rolle, kan dette isteden implementeres ved at senderen sender til en mailadresse som betegner er bestemt rolle.

Et eksempel kan være en student som ønsker å kontakte studieleder. Studenten vet kanskje hvem dette er, eller personen er ukjent for studenten. Ved å benytte Global Adress List (GAL), kan det defineres en mailmottaker som heter studieleder. Dermed er det mulig å sende til en rolle. Dermed trenger studenten hverken å vite hvem som utgjør rollen eller sette rollen som parameter i mailen.

Ved å benytte Event Scripting Agent, som beskrevet i 8.1.4, er det mulig å lage et script som for hver nye mail setter en bestemt rolle, sender mailen til personen som innehar rollen og sletter orginal mail etter at den er sent. Dermed får man mailen med parameterne satt slik at den kan organiseres avhengig av konteksten, men uten at brukeren må sette rollen. Bruk av Event Scripting Agent for informasjonsorganisering vil bli beskrevet i kapittel 10.3.

Ved å benytte former for å sette parametre som bestemmer kontekster, er det mulig å tvinge brukerne til å sette disse parameterne. Dette sikrer kvalitet ved metainformasjonen som blir lagret. Ved at brukeren får lettere tilgang til informasjon i forhold til aktiviteten som utføres, er det rimelig å anta at utfordringen beskrevet i kapittel 6.4.2 om at brukeren må føle at de får noe igjen for det for å benytte seg av metainformasjon, vil bli møtt.

Ved lagring av informasjon i Exchange direkte fra Word må teknikker som Document Management Extensions (DME) benytter seg av bli brukt. DME tilbyr setting av noen parametre, men ikke nok til å implementere parameterne definert i kapittel 7. For å vise at det er mulig å lagre direkte i Exchange trekkes DME her frem som et eksemplel. Måten dataene er lagret på, ved å legge til dokumentet som et vedlegg og ha metadataene i meldingen, vil ikke være hensiktsmessig i løsningen som skisseres her, da dette vil medføre at tilgang fra Internett ikke vil la seg gjennomføre i samme grad som ved å benytte Outlook sitt mail-objekt. Lagring fra Word må derfor være opp mot et Outlook documet-objekt. Dialogboksen for lagring må endres slik at den tvinger brukeren til å sette de ønskede parameterne.

Diskusjonen i dette kapitlet har vist at det er mulig å benytte forms i Outlook eller dialogbokser i Word for å tvinge brukeren til å sette ulike parametre. Disse kan velges blant et utvalg ved å ha drop-down bokser koblet mot en sentral database. Parameterne i formene eller dialogboksene kan også settes som et resultat av identifisering av kontekst, som beskrevet i forrige kapittel.

10.2.4 Søk etter/ arbeide med informasjon

Dialogboksene beskrevet over kan benyttes ved søk utført fra Word i Exchange databasen. Ved å koble dialogboksen opp mot et ASP-script som kjøres på Exchange serveren er det mulig å foreta søk på ulike parametre, og flere parametre samtidig. Dette scriptet må kunne gå igjennom de ulike tilgjengelig folderne på serverne som er aksesserbare og for hvert dataelement sjekke de aktuelle property'ene.

I Outlook er det mulig å benytte seg av find funksjonen. Denne er imidlertid for uoversiktitlig i den formen som følger med Outlook. Det må derfor utvikles et nytt grensesnitt til denne funksjonen hvor de ulike parameterne som er beskrevet i kapittel 10.1 kan settes. Parameterne rolle og kategori må som beskrevet i forrige kapittel kunne velges fra et forhåndsdefinert utvalg, og det må være mulig å benytte den lagrede konteksten for å sette de ulike parameterne.

Ved å søke på informasjon basert på disse parameterne vil spesifiseringen bli mer nøyaktig enn hvis ikke disse kunne blitt spesifisert. Dette forutsetter imidlertid at informasjonen er lagret som angitt i forrige kapittel slik at parameterne er satt korrekt. Parameterne vil ved arbeide med informasjon kunne settes slik at bare informasjonen som er relevant for konteksten vises. Dermed unngår brukeren å oppleve "information overload". Dette vil gjøre at brukeren oppnår noe ved å benytte metainformasjon (omtalt som en praktisk utfordring i kapittel 6.4.2), og dermed øker sannsynligheten for at brukeren vil benytte metainformasjonen (f. eks. sette parametre).

10.3 Automatisering av informasjonsorganisering

Bruk av regler kan spesielt benyttes for å behandle innkommende mail, men også ved opprettelse av nye document-objekter i en folder. Funksjonen organize gir tilgang til både bruk av regler samt noen andre muligheter. Det er spesielt to funksjoner som er attraktive ved kontekstbasert arbeids- og informasjonsorganisering: muligheten for å flytte dataelementer eller vise mailinformasjonen i en annen farge ved spesielle parametre.

Parametre som kan være aktuelle å sjekke ved innkommende mail er opprettet av (sender), rolle, kategori, organisasjon og navn på dataelement (subject). Mail kan da enten velges å sende til en egen folder, f. eks. mail som tilhører et spesielt fag, eller vise mailinformasjonen i en spesiell farge. Mail kan da organiseres i foldere etter kontekst, eller fargene kan brukes for raskt å visuelt gjenkjenne ulike mail basert på kontekster.

Rules kan også benyttes for å flytte dataelementer som legges i en generell folder. Et eksempel kan tenkes å være et fag hvor alle innleveringer legges i en spesiell folder. Dette gir enkel oversikt for elevene. Læreren kan da lage en regel som sier at dokumenter laget av spesielle personer, har en spesiell kategori eller har et spesielt navn (subject) skal flyttes til ønsket folder. Det kan f. eks. være foldere for de ulike oppgavene.

Andre muligheter for informasjonsorganisering gir Event Scripting Agent. De mest interessante mulighetene er å kunne kjøre et script er ved lagring eller endring av et dataelement, eller ved en spesiell tid, f. eks. en gang i døgnet. Tjenester som ønskes utført er da inspeksjon av dataelementer i den aktuelle folderen og en eller annen handling utført på disse.

Et eksempel kan være en videreføring av eksemplet i forrige avsnitt. Sett at læreren automatiserte flyttingen av oppgaver basert på hvilken øving det var. I hver folder for de ulike øvingene kunne det kjøres et script som for hvert nye dokument som kom inn til denne folderen fant navnet på den som stod som eiere av filen. Dette kan skrives ut som en tekstfil, og legges et sted på harddisken. Et annet ASP-script som ble kjørt regelmessig på Exchange serveren kunne igjen sjekke de ulike filene og legge ut en web-side basert på tekstfilene over hvor mange oppgaver hver enkelt student hadde levert.

For å håndtere ulike versjoner kan et script lages slik at det kjøres en gang i døgnet. Ved inspeksjon av dataelementene kan det sjekke om det er et dokument, og hvilken versjon dokumentet er. Hvis det er definert som ferdig versjon, kan det flyttes til et annet lager, hvor all ferdig f. eks. dokumentasjon for et prosjekt legges.

Som det delvis er kommet inn på tidligere er det som utprøvingen i kapittel 9.1 viste mulig å sende en mail fra en mailboks basert på spesielle hendelser. En slik hendelse kan være at det som det er nevnt i kapittel 10.2.3 ønskes å ha en mailboks tilknyttet en rolle. Et script kan da lages for denne mailboksen som ved innkommende mail sender mailen videre til den som for tiden fyller den aktuelle rollen. Da kan også ulike parametre settes,

som f. eks. hvilken rolle mailen som sendes videre assosieres med for den som mottar mailen.

Diskusjonen i dette kapitlet viser at regler kan benyttes for behandling av dataelementer. Det kan sjekkes parametre ved f. eks. innkommende mail, og dermed bestemme kontekst og videre behandling av mailen. Event Scripting Agent gir større muligheter for å organisere dataelementer avhengig av kontekst. Denne funksjonen kan flytte dataelementer basert på parametre, eller brukes for å håndtere mail forbundet med roller, for dermed å gi enda bedre arbeids- og informasjonsorganisering basert på den konteksten som brukeren er i ved behandling av mailen.

10.4 Bruk av web for aksess til Exchange

Exchange tilbyr som det er beskrevet i kapittel 8.1.1 mulighet for aksess via HTTP. Dette gir mulighet for å lage en webside for å vise data lagret i Exchange. Ved å sette opp en Internet Information Server som beskrevet i 9.2.3 og lage et ASP-script som benytter CDO for å aksessere Exchange, er dette mulig.

Outlook Today, som kommer med Outlook 2000, er egentlig en HTML-side som viser informasjon som ligger inne i Exchange databasen og som er av interesse for brukeren. Denne siden er generert for å gi personlig informasjon. Basert på de parameterne som er implementert i Office 2000 og Exchange, diskutert i kapittel 10.1, er det mulig å generere en HTML-side som gir relevant informasjon basert på ulike kontekster. Denne siden vil, fordi den er i HTML-format, være mulig å plassere på en IIS-server for å kunne tilby ønsket informasjon uansett hvor brukeren måtte befinne seg.

I Outlook kan denne siden assosieres med en folder, en mulighet kalt Folder Homepage. Dette vil si at ved å klikke på en folder, kommer bildet av den aktuelle web-siden opp i det store vinduet til høyre i grensesnittet til Outlook. En HTML-side for å vise informasjon tilsvarende Outlook Today, men med utgangspunkt i en spesiell kontekst, krever at parameterne som bestemmer konteksten må kunne settes.

Det har ingen hensikt å lage en ny Outlook Today, da denne allerede finnes. Det er mer hensiktsmessig å lage en side som kun viser informasjon av samme type som i Outlook Today, men basert på ulike kontekster. En slik side kan programmeres slik at den alltid ber brukeren fylle i kategori og/eller rolle ved aksessering. Valgene kan være i form av drop-down bokser. Disse kan være konstruert slik at ved valg av enten kategori eller rolle går scriptet gjennom aktuelle foldere og finner dataelementer med sammenfallende parametre. Hvis bare den ene velges først vil valg av den andre ytterligere spesifisere konteksten.

Ved aksess av web-siden er det hensiktsmessig å programmere denne slik at den sjekker om det finnes en temp-fil med informasjon om arbeidskonteksten. Denne tempfilen er presentert i kapittel 10.2.1. Dette kan tenkes naturlig fordi en bruker som jobber f. eks. med mail tilhørende en spesiell kontekst kan ha et behov for raskt å sjekke status for ulike

oppgaver eller dokumenter i forbindelse med denne konteksten, før brukeren svarer på en mail.

Andre parametre enn kategori, rolle og bruker er ikke nødvendig, da den typen informasjon som finnes i Outlook Today ikke trenger ytterligere spesifikasjon for å kunne vises avhengig av konteksten. Det er imidlertid mulig å tilby annen informasjon, som f. eks. hvilke typer dataelementer som ligger lagret i en folder eller hvor mange ferdige versjoner av dokumenter det finnes i en folder. Det er m.a.o. mange ulike typer informasjon som kan vises v.h.a. en slik web-side, men spesielt er at den benyttes for å vise informasjon basert på kontekst, og ikke bare person eller bruker pålogget systemet.

Ved å benytte en webside er det også mulig å gi en transparent presentasjon av informasjon uavhengig av informasjonsdomene. Exchange tilbyr som det er påpekt tidligere ulike Internett-baserte tilganger. Det gjør også andre databaser. Som det er påpekt i kapitell 8.2.2, er det v.h.a. ASP mulig å aksessere ulike databaser forutsatt at disse har en webtilgang. Da vil det være mulig å ytterligere modifisere websiden som er skissert tidligere i dette kapitlet. Identifikasjon av konteksten er allerede diskutert. Ved å programmere løsningen slik at den sender en SQL-søkestreng basert på parameterne som bestemmer konteksten, er det mulig å sjekke dataelementer i flere ulike typer databaser basert på den aktuelle konteksten. Resultatet av søket blir visualisert v.h.a. websiden.

Diskusjonen over viser at en Folder Homepage kan benyttes for å søke etter eller presentere informasjon basert på konteksten til brukeren. Konteksten må enten spesifiseres av brukeren selv, eller ved at systemet sjekker en fil for å kunne lese aktuell kontekst. Det er også mulig å tilby kontekstbasert presentasjon av informasjon fra flere ulike informasjonsdomener.

Konklusjon

Begrepet kontekstbasert arbeids- og informasjonsorganisering fantes ikke som frittstående begrep i den teorien som ble undersøkt. Det var heller ikke mulig å finne etablert forskning eller fagområder som hadde utgangspunkt tilsvarende denne oppgaven. De ulike fagområdene behandlet i varierende grad problemstillinger relatert til kontekstbasert arbeids- og informasjonsorganisering. Knowledge Management ble funnet som det fagområdet som hadde mest fellestrekk med begrepet. De andre fagområdene berørte andre problemstillinger med relevans for denne oppgaven, og kunne derfor også benyttes for å belyse problemstillingen i denne oppgaven.

Metainformasjon viste seg å være velegnet som metode for kontekstbasert arbeids- og informasjonsorganisering. Arbeidskonteksten kunne klassifiseres på grunnlag av noen parametre som ble diskutert og valgt, og disse parameterne kunne implementeres v.h.a. metainformasjon. Ved å ta utgangspunkt i Word-filer og Mail- og Document-objektet fra Outlook 2000 objekt-modellen, ble det vist at det er mulig å implementere de parameterne for klassifisering av kontekster som ble funnet, i Office 2000 og Exchange.

Ved å lage web-applikasjoner basert på ASP, CDO og VBScript ble det funnet at det var mulig å få tilgang til flere ulike informasjonsdomener fra en web-applikasjon. En slik applikasjon kunne få parametre fra brukeren ved aksess, og dermed presentere informasjon basert på ulike kontekster. Denne applikasjonen kunne assosieres med en Folder Homepage i Outlook 2000. Dermed var det mulig å gi tilgang til informasjon fra flere ulike informasjonsdomener basert på en kontekst, uten at brukeren måtte gå ut av Outlook 2000 for å finne denne informasjonen. Outlook 2000 var det programmet i Office 2000 som best egnet seg for kontekstbasert arbeids- og informasjonsorganisering. Programmet hadde også sammen med Exchange mulighet for å manipulere dataelementer i foldere avhengig av ulike kontekster, basert på ulike begivenheter.

Word 2000 kunne benyttes sammen med Exchange for lagring av dokumenter. Document Management Extensions (DME) viste at denne koblingen var mulig. For at Word 2000 skal kunne benytte kontekstbasert arbeids- og informasjonsorganisering, er det nødvendig med utvikling av en løsning tilsvarende DME, men med mulighet for å benytte Document-objektet og setting av parameterne for klassifisering av ulike kontekster ved lagring og åpning av dokumenter.

Ved å benytte kontekstbasert arbeids- og informasjonsorganisering vil brukeren få bedret muligheten for å få visualisert eller finne kun den informasjonen som er relevant for den aktuelle konteksten. Dette forutsetter at brukeren setter de ulike parameterne ved lagring av dataelementer. Dette kan sikres ved å benytte forms i Outlook 2000 eller en løsning som er lignende DME i Word 2000.

Eventuelt videre arbeid basert på denne oppgaven bør først konsentreres om utvikling av en prototyp for å kunne utføre praktiske tester og målinger av effekt ved kontekstbasert arbeids- og informasjonsorganisering. Praktiske erfaringer vil sannsynligvis avdekke nye problemstillinger som ikke har blitt behandlet i denne oppgaven. Spesielt må de

parameterne for klassifisering av kontekster som er funnet i oppgaven prøves ut, i et miljø tilsvarende det som er definert i avgrensningene for oppgaven, for å kartlegge om de dekker de kontekstene de er tiltenkt og om de virkelig lar seg implementere ved praktisk arbeid med informasjon.

Arbeidet med oppgaven gav mange nyttige erfaringer. Det ble erfart at søk blant relaterte fagområder til en problemstilling kan gi tilgang på andres erfaringer og refleksjoner, som igjen kan føre til at en ser nye muligheter og andre innfallsvinkler til problemstillingen. Ved arbeide med å utvikle en prototyp må mye arbeid legges ned i spesifikasjonen av hva som ønskes utviklet. Under diplomarbeidet ble denne delen av arbeidet viet for liten oppmerksomhet i begynnelsen av arbeidsperioden. Dette var et resultat av at jeg ikke hadde gode nok metodekunnskaper om utvikling av prototyper. De erfaringene som er gjort under arbeidet med diplomoppgaven må derfor kunne sees som verdifulle, selv om de ikke var gitt i oppgaven som et mål i seg selv.

Referanser

1. *Knowledge Management: An Industry Perspective Sponsored by BackWeb Technologies*, Knowledge Management World, Tilgjengelig: http://www.kmworld.com/newestlibrary/1998/backwebwp/backwebwp.cfm [1999, 25.03]

2. Gerber, B. (1997) *Mastering Exchange Server 5*, Sybex Inc., Alameda, CA

3. Grudin J. & Poltrock S. E. (1996) *Computer-Supported Cooperative Work and Groupware*

4. Klein, M.R. & Methlie, L.B. (1995) *Knowledge-based decision support systems*. Chichester: John Wiley. Kapittel 4.

5. Christensen G. E. (1997) *Strategisk omstilling med IT*, Praktisk Økonomi, 13:1, 90-101

6. Kim K.H. & Paik S. K. (1996) *Practical Experiences and Requirements on Workflow*, Coordination Technology for Collaborative Applications, Springer, Berlin Heidelberg, Germany 1998 : Conen W. & Neumann G. 145-160

7. Thomas Koulopoulos, Knowledge Management World, *Knowledge Management,Toward creating the "knowing enterprise"*, Tilgjengelig: http://www.kmworld/newestlibrary/1997/KMdelphi/KnowMgmtWPTOC.cfm [1999, 25.03]

8. Grudin J. (1994) *Computer-Supported Work: History and Focus*, IEEE Computer, mai 1994, 19-26

9. Venkatraman, N. (1994) *IT-Enabled Business Transformation: From Automation to Business Scope Redefinition.* Sloan Management Review, Winter. 73-87.

10. Singh K. J *Concurrent Engineering*

11. *Getting Started with Knowledge Management*, Excalibur Technologies, Tilgjengelig: http://www.excalibur.be/GB/WPapers/knowmgt.htm [1999, 26.03]

12. Beyer H. & Holtzblatt K. (1999) *Contextual Design* ACM 1072-5220/99/0100

13. Beyer H. & Holtzblatt K. (1995) *Apprenticing With the Costumer*, Communications of the ACM, 38:5, 45-52

14. Beyer H. & Holtzblatt K. (1993) *Making customer-centered design work for team,*, Communications of the ACM, 36:10, 92-103

15. *Representing Work for the Purpose of Design*, InContext Enterprise Inc.,Tilgjengelig: http://www.incent.com/papers.indx/WorkModeling.html [1999, 12.04]

16. Endsley, M. R. & Robertson M. M., *Team situation awareness in aviation maintenance*, Tilgjengelig: http://www.hfskyway.com/hfami/mtng10/endsley.htm [1999, 12.04]

17. Bryson S. et al (1996) *Strategic Directions in Human Computer Interaction*, Computing Surveys of ACM, eds. Myers B. et al, Tilgjengelig: http://www.cs.cmu.edu/~bam/nsfworkshop/hcireport.html [1999, 12.04]

18. Schiefloe P. M. & Syvertsen T. G. (1996), *Coordination in Knowledge-Intensive Organizations*, Coordination Technology for Collaborative Applications, Springer, Berlin Heidelberg, Germany 1998 : Conen W. & Neumann G. 9-23

19. Francis et al. (1998) *Active Server Pages 2.0*, Wrox Press Ltd., Canada

20. Cole P. *The Impact of Group Context on Patterns of Groupware Groupware Use: A Study of Computer Conferencing as a Medium of Work Group Communication and Coordination*, Center For Coordination Science MIT, http://ccs.mit.edu/CCSWP182.html [1999, 21.04]

21. Holowczak R. D. & Li W. S. *A Survey on Attribute Correspondence and Heterogeneity Metadata Representation*, IEEE, Tilgjengelig: http://www.computer.org/conferen/meta96/li/paper.html [1999, 12.04]

22. Schatz B. & Chen H. (1996), *Building Large Scale Digital Libraries*, IEEE Computer Society, Tilgjengelig: http://www.computer.org/computer/dli/ [1999, 12.04]

23. Wilensky R. (1996), *Toward Work-Centered Digital Information Services*, IEEE Computer Society, Tilgjengelig: http://www.computer.org/dli/r500370/r50037.htm [1999, 12.04]

24. Smith T. R. (1996), *A Digital Library for Geographically Referenced Materials*, IEEE Computer Society, Tilgjengelig: http://www.computer.org/computer/dli/r50054/r50054.htm [1999, 12.04]
25. Schatz B. (1996), *Federating Diverse Collections of Scientific Literature*, IEEE Computer Society, Tilgjengelig: http://www.computer.org/computer/dli/ [1999, 12.04]
26. Newton J. (1996), *Application of Metadata Standards*, IEEE Computer Society, Tilgjengelig: http://www.computer.org/conferen/meta96/newton/paper.html [1999, 21.04]
27. Heiler S. (1996), *Using Metadata to Address Problems of Semantic Interoperability in Large Object Systems*, IEEE Computer Society, Tilgjengelig: http://www.computer.org/conferen/meta96/miller/paper.html [1999, 21.04]
28. Keller T. & Jones D. (1996), *Metadata: The Foundation of Effective Experiment Management*, IEEE Computer Society, Tilgjengelig: http://www.computer.org/conferen/meta96/keller/metadata.html [1999, 21.04]
29. Brown P. et al (1996), *Metadata for Balanced Performance*, IEEE Computer Society, Tilgjengelig: http://www.computer.org/conferen/meta96/brown/all.html [1999, 21.04]
30. Callahan S. et al (1996), *Dataset Publishing – a Means to Motivate Metadata Entry*, IEEE Computer Society, Tilgjengelig: http://www.computer.org/conferen/meta96/callahan/callahan.html [1999, 21.04]
31. Griffioen J. et al (1996), *Automatic and Dynamic Identification of Metadata in Multimedia*, IEEE Computer Society, Tilgjengelig: http://www.computer.org/conferen/meta96/adams/paper.html [1999, 21.04]
32. Emanuelson P. & Hedlund I. (1996), *a Metadata Service for Meteorological Information*, IEEE Computer Society, Tilgjengelig: http://www.computer.org/conferen/meta96/par/paper.html [1999, 21.04]
33. Wãchter J. (1996) *GEOLIS – Innovative Geoscientific Information Management*, IEEE Computer Society, Tilgjengelig: http://www.computer.org/conferen/meta96/waechter/geolis.html [1999, 21.04]
34. Lownsbery B & Newton H. (1996), *The Key to Enduring Access: Multi-organizational Collaboration on the Development of Metadata for Use in Archiving Nuclear Weapons Data*, IEEE Computer Society, Tilgjengelig: http://www.computer.org/conferen/meta96/bel/enduring.html [1999, 21.04]
35. Foresman T. W. (1996), *Metadata Myth: Misunderstanding the Implications of Federal Metadata Standards*, IEEE Computer Society, Tilgjengelig: http://www.computer.org/conferen/meta96/wiggins/foresman_final.html [1999, 21.04]
36. Cleary J. et al (1996), *MetaData for Database Mining*, IEEE Computer Society, Tilgjengelig: http://www.computer.org/conferen/meta96/holmes/DataBaseMining.html [1999, 21.04]
37. Breiteneder C. J. et al (1996), *Metadata Mining in Legacy Data Sets*, IEEE Computer Society, Tilgjengelig: http://www.computer.org/conferen/meta96/breitender/Meta.html [1999, 21.04]
38. 80-20 Software, *Document Management Extensions Whitepaper*, Tilgjengelig: http://80-20.com/product/DMEPWDOC.zip [1999, 15.03]
39. Objektmodell for Outlook 2000, Tilgjengelig: http://www.microeye.com/images/OL2Kmap.gif [1999, 20.05]
40. Microsoft, *Outlook 2000 Product Enhancement Guide*, Tilgjengelig: http://www.microsoft.com/office/2000/Outlook/documents/OutlkPEG.doc [1999 05.04]
41. Rizzo, T. (1999), *Programming Microsoft Outlook and Microsoft Exchange*, Microsoft Press, Redmond, Wash.
42. CDO-hjelpfil, Tilgjengelig: http://www.cdolive.com./download/cdo.zip [1999 02.02]
43. CDOLive (1998) Event Scripting Agent Example, Tilgjengelig: http://www.cdolive.com/sample8.htm [1999, 19.02]

Vedlegg 1

Default.htm

Your browser should pull you to AuthLogon.asp. If not, feel free to click on through.

Global.asa

```
'<!--Copyright (c) Microsoft Corporation 1993-1997. All rights reserved.-->
<SCRIPT LANGUAGE="VBScript" RUNAT="Server">
Sub Application_OnStart
   On Error Resume Next
   Set objRenderApp = Server.CreateObject("AMHTML.Application")
   If Err = 0 Then
      Set Application("RenderApplication") = objRenderApp
   Else
      Application("startupFatal") = Err.Number
      Application("startupFatalDescription") = "Failed to create application object<br>" & Err.Description
   End If
   Application("hImp") = Empty
   Err.Clear
End Sub

Sub Application_OnEnd
   Set Application("RenderApplication") = Nothing
End Sub

Sub Session_OnStart
   On Error Resume Next
   ' this is a handle to the security context.  It will be set to the correct
   ' value when an Active Messaging session is created.
   Session("hImp") = Empty
   Set Session("AMSession") = Nothing
End Sub


' While calling the Session_onEnd event, Denali doesn't call us in the right
' security context. Workaround: current security context in stored in Session
' (look at logon.asp) and then gets restored in Session_onEnd event handler.
'
' All ActiveMessaging and ActiveMessagingRenderingLibrary objects stored in the session object
' need to be explicitly set to Nothing between the two objRenderApp.Impersonate calls below
Sub Session_OnEnd
   On Error Resume Next
   set objRenderApp = Application("RenderApplication")
   hImp = Session("hImp")
   If Not IsEmpty(hImp) Then
      objRenderApp.Impersonate(hImp)
   End If

   ' Do our cleanup- set all AM and AMHTML objects inside the session to Nothing
   ' The AM session is a little special because we need to do the Logoff on it.
   Set objOMSession = Session("AMSession")
   If Not objOMSession Is Nothing Then
      Set Session("AMSession") = Nothing
      objOMSession.Logoff
      Set objOMSession = Nothing
   End If
End Sub
</SCRIPT>
```

AuthLogon.asp

```
<!--#include file="AuthLogon.inc"-->
<!--#include file="SendMail.inc"-->
<%
'<!--Copyright (c) Microsoft Corporation 1993-1997. All rights reserved.-->

   On Error Resume Next

   Dim objAMSession

   ' This should occur at the top of the script.  If the AM session does not
   ' exist and the URL does not contain server and mailbox data, a logon form will be
   ' emited and returned for the user to fill out.
   CheckAMSession                    'AuthLogon.inc

   ' This should occur near the top of the script as well, before anything that may cause
   ' any text to be emited to the browser.  If the user is not already authenticated,
   ' this function will return error 401 (Access denied) and prompt for credentials.
   BAuthenticateUser         'AuthLogon.inc

   ' After calling CheckAMSession, the AM session can be retrieved from
   ' Session("AMSession").  CheckAMSession may have failed to create a session, though,
   ' so check whether objAMSession is Nothing and if so cleanup, inform the user, etc.
   Set objAMSession= Session( "AMSession")

   if objAMSession Is Nothing Then
      ' CheckSession was unable to retrieve or create a session
      Response.Write( "GetAMSession returned nothing!<br>")
   End If

   GetMessageInfo objAMSession   'SendMail.inc
%>

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 3.2//EN">
<HTML>
<HEAD>
</HEAD>
<BODY>

<a href="AuthLogon.asp">AuthLogon.asp</a><br>
<a href="AuthLogoff.asp">AuthLogoff.asp</a><br>

</BODY>
```

AuthLogin.inc

```
<%
' AuthLogin.inc.  VBScript methods to create and check an ActiveMessaging session
'<!--Copyright (c) Microsoft Corporation 1993-1997. All rights reserved.-->
'============================
' ReportError
'
'============================
Function ReportError( bstrContext)
    ReportError= False
    If Err.Number <> 0 Then
        Response.Write( "Error: " & bstrContext & " : " & Err.Number & ": " & Err.Description & "<br>")
        Err.Clear
        ReportError= True
    End If
End Function


'============================
' CheckAMSession
'
' Checks for and returns the AM session in the Sesion object.  If not found,
' calls NoSession().
' Call this before emiting any HTML or any redirects, authentication, etc. may not work.
' Returns: True if session exists or can be created
'============================
Public Function CheckAMSession
    On Error Resume Next
    Dim amSession
    CheckAMSession= False
    Set amSession= Session( "AMSession")
    If amSession Is Nothing Then
        NoSession
        Set amSession= Session( "AMSession")
    End If

    If Not amSession Is Nothing Then
        CheckAMSession= true
    End If
End Function



'============================
' NoSession
'
' Called when the AM session can not be found.  Either creates a session or gets
' more info from the user.  Returns only if the session was created.
'============================
Sub NoSession
    On Error Resume Next
    Dim strLogonID
    Dim bstrServer
    Dim bstrMailbox
    Dim bstrProfileInfo
    Dim objAMSession1
    Dim objRenderApp
    Dim objInbox


    ' must be authenticated to sucessfully logon
```

```
    BAuthenticateUser
    Err.Clear
    Set objAMSession1 = Server.CreateObject("MAPI.Session")
    If Not ReportError( "create MAPI.Session") Then
        set objRenderApp = Application( "RenderApplication" )

        ' Construct the ActiveMessaging profile from server and mailbox name
        'Set bstrServer equal to an exchange server on the same site as the expected user
        bstrServer = "Diplom"

        'Retrieve bstrMailbox from the LOGON_USER server variable
        'Get Logon_User and cut the string to just the UserID
        strLogonID = Request.ServerVariables("Logon_User")
        bstrMailbox = Right(strLogonID, Len(strLogonID) - _
                InStr(strLogonID, "\"))

        bstrProfileInfo = bstrServer + vbLF + bstrMailbox
        Err.Clear

        objAMSession1.Logon "", "", False, True, 0, True, bstrProfileInfo
        If Not ReportError( "ActiveMessaging Logon") Then
            ' To ensure that we are really logged on, we need to try retrieving
            ' some data.
            Err.Clear
            Set objInbox = objAMSession1.Inbox
            If ReportError( "Get Inbox") Then
                ' The logon is no good.  We'll do a little cleanup here.
                objAMSession1.Logoff
                Set objAMSession1 = Nothing
            End If

            ' This will be retrieved back up in CheckSession
            ' Note that if the logon failed this is set to 'Nothing'
            Set Session("AMSession") = objAMSession1

            ' Need this to recreate the proper security context in Session_OnEnd
            Session("hImp") = objRenderApp.ImpID
        End If    'objAMSession1.Logon
    End If   'Server.CreateObject()
End Sub

'============================
' AuthenticateUser

' Ensures user is authenticated
' Note that this implies that Basic authentication is enabled on the IIS server
'============================
Public Function BAuthenticateUser
    On Error Resume Next
    BAuthenticateUser = False
    bstrAT = Request.ServerVariables("AUTH_TYPE")
    If InStr(1, "_BasicNTLM", bstrAT, vbTextCompare) < 2 Then
        Response.Buffer = TRUE
        Response.Status = ("401 Unauthorized")
        Response.End
    Else
        BAuthenticateUser = True
    End If
End Function

%>
```

AuthSendMail.inc

```
<%
' AuthSendMail.inc.  VBScript methods to create and check an ActiveMessaging session
'<!--Copyright (c) Microsoft Corporation 1993-1997. All rights reserved.-->
'============================
' ReportError
'
'============================
Function ReportError( bstrContext)
    ReportError= False
    If Err.Number <> 0 Then
        Response.Write( "Error: " & bstrContext & " : " & Err.Number & ": " & Err.Description & "<br>")
        Err.Clear
        ReportError= True
    End If
End Function


'============================
' CheckAMSession
'
' Checks for and returns the AM session in the Sesion object.  If not found,
' calls NoSession().
' Call this before emiting any HTML or any redirects, authentication, etc. may not work.
' Returns: True if session exists or can be created
'============================
Public Function CheckAMSession
    On Error Resume Next
    Dim amSession
    CheckAMSession= False
    Set amSession= Session( "AMSession")
    If amSession Is Nothing Then
        NoSession
        Set amSession= Session( "AMSession")
    End If

    If Not amSession Is Nothing Then
        CheckAMSession= true
    End If
End Function



'============================
' NoSession
'
' Called when the AM session can not be found.  Either creates a session or gets
' more info from the user.  Returns only if the session was created.
'============================
Sub NoSession
    On Error Resume Next
    Dim strLogonID
    Dim bstrServer
    Dim bstrMailbox
    Dim bstrProfileInfo
    Dim objAMSession1
    Dim objRenderApp
    Dim objInbox
```

```
    ' must be authenticated to sucessfully logon
    BAuthenticateUser
    Err.Clear
    Set objAMSession1 = Server.CreateObject("MAPI.Session")
    If Not ReportError( "create MAPI.Session") Then
        set objRenderApp = Application( "RenderApplication" )

        ' Construct the ActiveMessaging profile from server and mailbox name
        'Set bstrServer equal to an exchange server on the same site as the expected user
        bstrServer = "Diplom"

        'Retrieve bstrMailbox from the LOGON_USER server variable
        'Get Logon_User and cut the string to just the UserID
        strLogonID = Request.ServerVariables("Logon_User")
        bstrMailbox = Right(strLogonID, Len(strLogonID) - _
                InStr(strLogonID, "\"))

        bstrProfileInfo = bstrServer + vbLF + bstrMailbox
        Err.Clear

        objAMSession1.Logon "", "", False, True, 0, True, bstrProfileInfo
        If Not ReportError( "ActiveMessaging Logon") Then
            ' To ensure that we are really logged on, we need to try retrieving
            ' some data.
            Err.Clear
            Set objInbox = objAMSession1.Inbox
            If ReportError( "Get Inbox") Then
                ' The logon is no good.  We'll do a little cleanup here.
                objAMSession1.Logoff
                Set objAMSession1 = Nothing
            End If

            ' This will be retrieved back up in CheckSession
            ' Note that if the logon failed this is set to 'Nothing'
            Set Session("AMSession") = objAMSession1

            ' Need this to recreate the proper security context in Session_OnEnd
            Session("hImp") = objRenderApp.ImpID
        End If    'objAMSession1.Logon
    End If   'Server.CreateObject()
End Sub

'============================
' AuthenticateUser

' Ensures user is authenticated
' Note that this implies that Basic authentication is enabled on the IIS server
'============================
Public Function BAuthenticateUser
    On Error Resume Next
    BAuthenticateUser = False
    bstrAT = Request.ServerVariables("AUTH_TYPE")
    If InStr(1, "_BasicNTLM", bstrAT, vbTextCompare) < 2 Then
        Response.Buffer = TRUE
        Response.Status = ("401 Unauthorized")
        Response.End
    Else
        BAuthenticateUser = True
    End If
End Function
```

```
'==============================
'Get message information and send
'
' Finds a server and mailbox name to use to login with.  This sample returns
' a form to the user requesting the information.  By replacing this subroutine,
' the data could come from anywhere else (a database, hardcoded, etc.)
'==============================
Sub GetMessageInfo(ByRef objAMSession)
    Dim objOutbox
    Dim objMessage
    Dim objRecipient
    Dim strLogonID
    Dim bstrMailbox
    Dim bstrServer
    Dim strRecipient
    Dim strSubject
    Dim strMessage

    'Retrieve bstrMailbox from the LOGON_USER server variable
    'Get Logon_User and cut the string to just the UserID
    strLogonID = Request.ServerVariables("Logon_User")
    bstrMailbox = Right(strLogonID, Len(strLogonID) - _
      InStr(strLogonID, "\"))
    bstrServer = "Diplom"
    strRecipient = Request.QueryString( "txtRecipient")
    If strRecipient = "" then
        Response.Write("<html><head><title> Please fill in all fields for your message. </title></head>")
        Response.Write("<body><FORM METHOD=GET WINDOW=_self><TABLE>")
        Response.Write("<tr><td>Recipient:</td><td><input TYPE=TEXT SIZE=20
NAME=""txtRecipient""></td></tr>")
        Response.Write("<tr><td>Subject:</td><td><input TYPE=TEXT SIZE=20
NAME=""txtrSubject""></td></tr>")
        Response.Write("<tr><td> Message:</td><td><textarea name=""txtMessage"" rows=""13""
cols=""50""></textarea>")
        Response.Write("</TABLE><P><INPUT TYPE=""SUBMIT"" VALUE=""OK""><INPUT
TYPE=""RESET"" VALUE=""Reset""></FORM>")
        ' nothing after the next line will be executed
        Response.End
    Else
        strSubject= Request.QueryString( "txtSubject")
        strMessage = Request.QueryString( "txtMessage")
        Set objOutbox = objAMSession.Outbox
        Set objMessage = objOutbox.Messages.Add
        Set objRecipient = objMessage.Recipients.Add
        objRecipient.Name =  Request.QueryString( "txtRecipient")
        'objRecpient.Type = 1
       ' Response.Write("AuthSendMail.inc - objRecipient.Type = " & objRecipient.Type & "<br>")
       ' Response.Write("AuthSendMail.inc - Resolving ojRecipient.Name<br>")
        objRecipient.Resolve
        objMessage.Subject = Request.QueryString( "txtSubject")
        objMessage.Text =  Request.QueryString( "txtMessage")
        objMessage.Send
        Response.Write("AuthSendMail.inc - Message Sent<br>")
    End If

    Set objRecipient  = Nothing
    Set objMessage = Nothing
    Set objOutbox = Nothing
End Sub
%>
```

SendMail.inc

```
<%
' SendMail.inc.  VBScript methods to retrieve information from a user and send mail
'<!--Copyright (c) Microsoft Corporation 1993-1997. All rights reserved.-->
'==============================
'Get message information and send
'
' Finds a server and mailbox name to use to login with.  This sample returns
' a form to the user requesting the information.  By replacing this subroutine,
' the data could come from anywhere else (a database, hardcoded, etc.)
'==============================
Sub GetMessageInfo(ByRef objAMSession)
    Dim objOutbox
    Dim objMessage
    Dim objRecipients
    Dim objRecipient
    Dim strLogonID
    Dim bstrMailbox
    Dim bstrServer
    Dim strRecipient
    Dim strSubject
    Dim strMessage

   On Error Resume Next
    strRecipient = Request.QueryString( "txtRecipient")
    If strRecipient = "" then
        Response.Write("You are currently logged into the  mailbox for: <b>" & objAMSession.CurrentUser &
"</b><br>")
        Response.Write("Please fill in all fields for your message.<br>")
        Response.Write("<html><head><title></title></head>")
        Response.Write("<body><FORM METHOD=GET WINDOW=_self><TABLE>")
        Response.Write("<table border=""0""> ")
        Response.Write("<tr><td valign=""top"">Recipient:</td> ")
        Response.Write("<td valign=""top""><input type=""text"" size=""20"" ")
        Response.Write("name=""txtRecipient"")></td></tr> ")
        Response.Write("<tr><td valign=""top"">Subject:</td> ")
        Response.Write("<td valign=""top""><input type=""text"" size=""20"" ")
        Response.Write("name=""txtSubject""></td></tr> ")
        'Additional stuff by KHTM
        Response.Write("<tr><td valign=""top"">Categories:</td> ")
        Response.Write("<td valign=""top""><input type=""text"" size=""20"" ")
        Response.Write("name=""txtCategories""></td></tr> ")
        'End AddStuff
        Response.Write("<tr><td valign=""top"">Message:</td> ")
        Response.Write("<td><textarea name=""txtMessage"" rows=""13"" cols=""52""></textarea></td></tr> ")
        Response.Write("</table> ")
        Response.Write("<p><input type=""submit"" name=""B1"" value=""Submit""><input type=""reset""
name=""B2"" value=""Reset""></p> ")
        Response.Write("</form>")
        'Additional Stuff By KHTM
        Set objInbox = objAMSession.Inbox
        'Set objMsgColl = objInbox.Messages
        'Set intMessCount = objMsgColl.Count
        intMessCountUnread = objAMSession.Inbox.Fields(&H36030003)
        intMessCountTotal = objAMSession.Inbox.Fields(&H36020003)
        intMessCountTotal2 = objAMSession.Inbox.Messages.Count
        Response.Write("Number of unread messages: " & intMessCountUnread & "<Br>")
        Response.Write("Total number of messages: " & intMessCountTotal & " (Using Hex-code)<Br>")
        Response.Write("Total number of messages: " & intMessCountTotal2 & " (Using Count
property)<Br><Br>")
        Response.Write("</body></html>")
```

```
        ' nothing after the next line will be executed
        Response.End
    Else
        strSubject= Request.QueryString( "txtSubject")
        strMessage = Request.QueryString( "txtMessage")
        Set objOutbox = objAMSession.Outbox
        Set objMessage = objOutbox.Messages.Add
        Set objRecipients = objMessage.Recipients
        Set objRecipient = objRecipients.Add
        objRecipient.Name =  Request.QueryString( "txtRecipient")
        err.clear
        objRecipient.Resolve
        if err.number= -2147221233  then
           response.write("Recipient not found,please use your back button to reload your page<br><br>")
       else
            objMessage.Subject = Request.QueryString( "txtSubject")
            objMessage.Text =  Request.QueryString( "txtMessage")
            objMessage.Categories = Request.QueryString( "txtCategories")
            objMessage.Send
            Response.Write("The following message was sent<br>")
            Response.Write("Sender: " &  objAMSession.CurrentUser & "<br>")
            Response.Write("Recipient: " & Request.QueryString("txtRecipient") &  "<br>")
            Response.Write("Subject:  " &  Request.QueryString("txtSubject") & "<br>")
            Response.Write("Categories:  " &  Request.QueryString("txtCategories") & "<br>")
            Response.Write("Message:  " &  Request.QueryString("txtMessage") &  "<br>")
       end if
    End If
    Set objRecipient  = Nothing
    Set objMessage = Nothing
    Set objOutbox = Nothing
End Sub
%>
```

AuthLogoff.asp

```
<%
'<!--Copyright (c) Microsoft Corporation 1993-1997. All rights reserved.-->
%>
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 3.2//EN">
<HTML>
<HEAD>
</HEAD>
<BODY>

Abandoning Denali session.  This will logoff AM session.<br><br>

<%
    ' This will trigger Session_OnEnd in global.asa.  It would also be possible
    ' to explicitly logoff here and set all Session objects to Nothing
    Session.Abandon
%>

<a href="Authlogon.asp">AuthLogon.asp</a>


</BODY>
</HTML>
```

Vedlegg 2

```
<SCRIPT RunAt=Server Language=VBScript>

'THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT
'WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED,
'INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES
'OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR
'PURPOSE

'---------------------------------------------------------------------------
'
' NAME: MbxReport
'
' FILE DESCRIPTION: Creates a report of all mailboxes on the same Exchange Server
'                   and send it to the system administrator.
'
' Copyright (c) CdoLive 1998. All rights reserved.
'               Http://www.cdolive.com
'               Mailto:samples@cdolive.com
'
' Portions:
' Copyright (c) Microsoft Corporation 1993-1997. All rights reserved.
'
'---------------------------------------------------------------------------

Option Explicit

'---------------------------------------------------------------------------
'             Global Variables
'---------------------------------------------------------------------------

Dim g_bstrDebug                                                 ' Debug String

'---------------------------------------------------------------------------
'             CONSTANTS
'---------------------------------------------------------------------------

Dim g_Const_MBX
Dim g_Const_ReportFile
Dim CdoPR_ACCOUNT
Dim CdoPR_MAILBOX_SIZE
Dim CdoPR_EMAIL
Dim CdoPR_CONTENT_COUNT
Dim CdoPR_CONTENT_UNREAD
Dim PR_EMS_AB_HOME_MTA
Dim PR_LAST_LOGON_TIME
Dim PR_LAST_LOGOFF_TIME

' Enter the display name of the mailbox which you want have send the report to
g_Const_MBX = "Service"

' Enter the path and filename of the report file
' if you use the extension csv it can be opened with Microsoft Excel
g_Const_ReportFile = "c:\temp\report.csv"

' MAPI properties not documented
CdoPR_MAILBOX_SIZE   = &HE08001E                ' Size of the mailbox in byte
CdoPR_EMAIL          = &H39FE001E               ' SMTP address of the mailbox
```

```
' MAPI properties which are documented
CdoPR_ACCOUNT       = &H3A00001E              ' Alias
CdoPR_CONTENT_COUNT  = &H36020003             ' Count of all messages of the mailbox
CdoPR_CONTENT_UNREAD = &H36030003                     ' Count of unread messages of the inbox
PR_EMS_AB_HOME_MTA   = &H8007001E             ' Home MTA needed to get the home server name

' MAPI properties cannot be accessed with this script
PR_LAST_LOGON_TIME   = &H66A20040             ' Last logon time
PR_LAST_LOGOFF_TIME  = &H66A30040             ' Last logoff time

'--------------------------------------------------------------------------
'           EVENT HANDLERS
'--------------------------------------------------------------------------

' DESCRIPTION: This event is fired when a new message is added to the folder
Public Sub Folder_OnMessageCreated
        'Not used
End Sub

' DESCRIPTION: This event is fired when a message in the folder is changed
Public Sub Message_OnChange
        'Not used
End Sub

' DESCRIPTION: This event is fired when a message is deleted from the folder
Public Sub Folder_OnMessageDeleted
        'Not used
End Sub

' DESCRIPTION: This event is fired when the timer on the folder expires
Public Sub Folder_OnTimer
Dim objSession                                  ' Session
Dim objReportMsg                                ' Report message
Dim objFolderOutbox                             ' Outbox folder
Dim objRecipient                        ' Recipient
Dim objUserSess                                 ' User session
Dim objRecipientList                            ' Recipients address list
Dim objUserList                                 ' Address entries of the recipients
Dim objUser                                     ' Single user
Dim objFileSystem                               ' Scripting Filesystem object
Dim objFile                                     ' File object
Dim objAttachment                               ' Attachment object

Dim intCounter                                  ' Counter
Dim strProfileInfo                      ' MAPI profile information
Dim strReportText                               ' Report with results
Dim strUserID                                   ' Session ID of the user MAPI session
Dim strUserName                                 ' Username
Dim intUserCount                        ' Count of messages
Dim intUserUnread                               ' Count of unread messages
Dim intUserSize                                 ' Mailbox size
Dim strUserMail                                 ' E-mail address
Dim strUserHomeServer                           ' Mailbox home server
Dim strAgentID                                  ' Session ID of the agent MAPI session
Dim strAgentHomeServer                          ' Agent home server

' Initialize objects
Set objSession = Nothing
Set objReportMsg = Nothing
Set objFolderOutbox = Nothing
Set objRecipient = Nothing
Set objUserSess = Nothing
```

```
Set objRecipientList = Nothing
Set objUserList = Nothing
Set objUser = Nothing
Set objFileSystem = Nothing
Set objFile = Nothing
Set objAttachment = Nothing

' Set header information of the report textfile in CSV format for use in conjunction with Microsoft Excel
strReportText = "Alias;E-mail Address;Total Mailbox Size,Total Mailbox Messages;Unread Messages at the
Inbox" & Chr(13)

' Clear error buffer
Err.Clear

' Get session informationen
Set objSession = EventDetails.Session

' No errors detected ?
If Err.Number = 0 Then

' Write some logging
Call DebugAppend(objSession.CurrentUser & " MbxReport - Proccessing startet", False)

' Get session ID of the current logged on MAPI session
strAgentID = objSession.CurrentUser.ID

' Get home MTA property of the current session and extract the home server
strAgentHomeServer = objSession.GetAddressEntry(strAgentID).Fields(PR_EMS_AB_HOME_MTA)
strAgentHomeServer = Mid(strAgentHomeServer, InStr(1, strAgentHomeServer,
"/cn=Configuration/cn=Servers/cn=") + Len("/cn=Configuration/cn=Servers/cn="),255)
strAgentHomeServer = Left(strAgentHomeServer, InStr(1, strAgentHomeServer, "/") -1)

' Get global address list
On Error Resume Next
Set objRecipientList = objSession.GetAddressList(0)
' No errors detected ?
If Err.Number = 0 Then

' Getting userlist, write logging
Call DebugAppend("Getting userlist", False)

' Get collection of address entries
On Error Resume Next
Set objUserList = objRecipientList.AddressEntries
Else

' Could not get global address list, write logging
Call DebugAppend("Error - Could not get global address list", True)
End If

' Check if userlist is not empty
If Not objUserList Is Nothing Then

' Userlist successfully found, write logging
Call DebugAppend("Userlist successfully found", False)

' Loop through the userlist
For Each objUser In objUserList

' Check if the object is a mailbox
If objUser.DisplayType = 0 Then
```

```
' Mailbox found, write logging
Call DebugAppend("Mailbox found: " & objUser.Name, False)

' Get home MTA property of the user and extract the home server
                                                    strUserHomeServer =
objUser.Fields(PR_EMS_AB_HOME_MTA)
                                            strUserHomeServer = Mid(strUserHomeServer, InStr(1,
strUserHomeServer, "/cn=Configuration/cn=Servers/cn=") + Len("/cn=Configuration/cn=Servers/cn="),255)
                        strUserHomeServer = Left(strUserHomeServer, InStr(1, strUserHomeServer, "/") -1)

' Mailbox is on the same server the scirpt is running?
If Trim(UCase(UserServerName)) = Trim(UCase(strAgentHomeServer)) Then

' Agent is running on the mailbox home server, write logging
Call DebugAppend("Agent is running on the mailbox home server", False)

' Create MAPI session
Set objUserSess = Nothing
On Error Resume Next
Set objUserSess = CreateObject("MAPI.Session")

' No errors detected ?
If Not objUserSess Is Nothing Then

' Logging on with user, write logging
Call DebugAppend("Logging on with user: " & objUser.Name, False)

' Create MAPI profile and logon to the user mailbox
Err.Clear
strProfileInfo = strUserHomeServer & Chr(10) & objUser.Fields.Item(CdoPR_ACCOUNT).Value
On Error Resume Next
objUserSess.Logon "", "", False, True, 0, False, strProfileInfo

' No errors detected ?
If Err.Number = 0 Then

' Logon successfully, write logging
Call DebugAppend("Logon successfully with user: " & objUserSess.CurrentUser, False)

' Initialize variables
intCounter = 1
intUserSize = ""
intUserCount = ""
intUserUnread = ""
' Get user ID, name and SMTP address
strUserID = objUserSess.CurrentUser.ID
strUserName = objUserSess.CurrentUser
strUserMail = objUserSess.GetAddressEntry(strUserID).Fields(CdoPR_EMAIL)

' Loop through the infostores
For intCounter = 1 To objUserSess.InfoStores.Count

' Check if we have not hit the public information store
' Note that the name of the public information store is not language independent
' You must change it to the language of the Exchange Server which you use
If Trim(UCase(objUserSess.Infostores.Item(intCounter).Name)) <> "PUBLIC FOLDERS" Then

' Getting mailbox information, write logging
Call DebugAppend("Getting mailbox information", False)

' Getting the all message count of the mailbox
intUserCount = objUserSess.Infostores.Item(intCounter).Fields(CdoPR_CONTENT_COUNT)
```

```
' Getting unread message count of the inbox
intUserUnread = objUserSess.Inbox.Fields(CdoPR_CONTENT_UNREAD)
' Get size of the mailbox
intUserSize = Int((objUserSess.Infostores.Item(intCounter).Fields(CdoPR_MAILBOX_SIZE))  / 1024)
If Trim(intUserSize) <> "" Then
intUserSize = intUserSize & " Kbytes"
Else
intUserSize = "N/A"
End If
' Create report data, write logging
Call DebugAppend("Write report data", False)
' Add the results to the report text
strReportText = strReportText & strUserName & ";" & strUserMail & ";" & intUserSize & ";" & intUserCount &
";" & intUserUnread & Chr(13)
End If
Next

' Close MAPI session, write logging
Call DebugAppend("Close MAPI session", False)

' Logoff from the mailbox
On Error Resume Next
objUserSess.Logoff
Set objUserSess = Nothing
Else

' Could not log on, write logging
Call DebugAppend("Error - Could not log on on with user: " & objUser.Name, True)
End If
Else

' Could not create MAPI session, write logging
Call DebugAppend("Error - Could not create MAPI session", True)
End If
End If
End If
Next

' Clear error buffer
Err.Clear

' Check if report is not empty
If Trim(strReportText) <> "" Then

' Get outbox folder
On Error Resume Next
Set objFolderOutbox = objSession.Outbox

' No errors detected ?
If Not objFolderOutbox Is Nothing Then

' Create Scripting FileSystem object
On Error Resume Next
Set objFileSystem = CreateObject("Scripting.FileSystemObject")

' No errors detected ?
If Not objFileSystem Is Nothing Then

' Create file, if already exists overwrite it
On Error Resume Next
Set objFile = objFileSystem.CreateTextFile(g_Const_ReportFile, True)
```

```
' No errors detected ?
If Not objFile Is Nothing Then

' Write report to file
objFile.Write strReportText

' Close file
objFile.Close
End If
End If

' Create new message
On Error Resume Next
Set objReportMsg = objFolderOutbox.Messages.Add

' No errors detected ?
If Not objReportMsg Is Nothing Then

' Add attachment to the message
On Error Resume Next
Set objAttachment = objReportMsg.Attachments.Add

' No errors detected?
If Not objAttachment Is Nothing Then

' Add the report as attachment at the end of the message
objAttachment.Position = -1
objAttachment.Type = 1
objAttachment.Source = g_Const_ReportFile
objAttachment.ReadFromFile = g_Const_ReportFile
End If

' Set subject
objReportMsg.Subject = "Mailbox Report created: " & Date

' Enter your message text here
objReportMsg.Text = "Attached is your scheduled mailbox report" & Chr(10)

' Set recipient
Set objRecipient = objReportMsg.Recipients.Add
objRecipient.Name = g_Const_MBX

' Resolve recipient against the Exchange Directory
Err.Clear
objRecipient.Resolve

' Error detected ?
If Err.Number <> 0 Then

' Could not resolve recipient, write logging
Call DebugAppend("Error - Could not resolve recipient: " & g_Const_MBX, True)
Else

' Sent message
On Error Resume Next
objReportMsg.Update
objReportMsg.Send

' Error detected ?
If Err.Number <> 0 then
```

```
' Could not sent message, write logging
Call DebugAppend("Error - Could not sent message", True)
Else

' Message successfully sent
Call DebugAppend("Success - Message sent to: " & g_Const_MBX, False)
End If
End If
Else

' Could not create new message
Call DebugAppend("Error - Failed to create message", True)
End If
Else

' Could not get outbox
Call DebugAppend("Error - Could not get outbox", True)
End If
Else

' Report is empty, write logging
Call DebugAppend("Error - Report is empty", False)
End If
Else

' Could not get userlist, write logging
Call DebugAppend("Error - Could not get userlist", True)
End If
Else

' Check for any possible sys errors
Call DebugAppend("Undefinied Error detected", True)
End If

' Write some logging
Call DebugAppend("MbxReport - Processing finished", False)

' Clear objects
Set objSession = Nothing
Set objReportMsg = Nothing
Set objFolderOutbox = Nothing
Set objRecipient = Nothing
Set objUserSess = Nothing
Set objRecipientList = Nothing
Set objUserList = Nothing
Set objUser = Nothing
Set objFileSystem = Nothing
Set objFile = Nothing
Set objAttachment = Nothing

' Write results to the Scripting Agent log
Script.Response = g_bstrDebug
End Sub

'-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
'            PRIVATE FUNCTIONS/SUBS
'-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

'-----------------------------------------------------------------------------
'  Name: DebugAppend
'  Area: Debug
'  Desc: Simple Debugging Function
```

```
'   Parm: String Text, Bool ErrorFlag
'--------------------------------------------------------------------------------

Private Sub DebugAppend(bstrParm,boolErrChkFlag)

	If boolErrChkFlag = True Then
		If Err.Number <> 0 Then
			g_bstrDebug = g_bstrDebug & bstrParm & " - " & cstr(Err.Number) & " " &
Err.Description & vbCrLf
			Err.Clear
		End If
	Else
		g_bstrDebug = g_bstrDebug & bstrParm & vbCrLf
	End If

End Sub

</SCRIPT>
```