UNIVERSITETET I OSLO
Institutt for informatikk

En flerspråklig bibliografiprosessor

Marius L. Jøhndal

Hovedoppgave

25. juli 2003

Forord

Denne hovedoppgaven ble påbegynt høsten 2002 på Institutt for informatikk ved Universitet i Oslo. Den er veiledet av Dag Langmyhr ved studieretningen databehandling. Oppgaven beskriver et prosjekt som studerer forutsetningene for, kravene til og utformingen av en bibliografiprosessor for vitenskapelig forfatterskap og flerspråklige bibliografier. Målsetningen har vært å beskrive de spesielle hensyn man må ta for å kunne behandle flerspråklige bibliografier, og å utvikle en slik bibliografiprosessor.

Flere har bidratt til denne oppgaven på forskjellige måter. Først og fremst ønsker jeg å takke Dag F. Langmyhr for veiledning og Fredrik Skribeland for korrekturlesning.

Blindern, 25. juli 2003
Marius L. Jøhndal

Innhold

Figurer

Tabeller

Kapittel 1

Innledning

> «It is more complicated than you think.»
>
> – The Eighth Networking Truth, RFC 1925

Dokumentproduksjon er en viktig del av mange menneskers hverdag. Programvare for dokumentproduksjon er først og fremst innrettet mot tekstbehandling og tilbyr derfor funksjoner som stavekontroll, linjebrytning, automatisk formatering og så videre.

For vitenskapelig publikasjon er det også behov for hjelpemidler som kan holde orden på bibliografier. Bibliografier er en essensiell del av de fleste former for vitenskapelige publikasjoner. Kravene til at påstander skal være etterprøvbare, at sitater skal kunne gjenfinnes i full form, at resultater skal kunne tilbakeføres til sine opphavsmenn, og, ikke minst, at oppdagelser og ideer skal krediteres sine skapere, sikres blant annet ved høye krav til konsise, presise og utfyllende bibliografier. Bibliografisk programvare bør derfor både ta vare på de bibliografiske opplysningene og sørge for at disse tilknyttes dokumenter på en korrekt måte.

Slik programvare finnes allerede. Et slikt program er BibTeX, og fordelene ved dette beskrives slik av en meget flittig bruker:

> «The major benefits of using BibTeX are the potential for data reuse, the separation of form and content [...], and the many stylistic variants of the typeset bibliography.» (Nelson Beebe, *Bibliography prettyprinting and syntax checking* [4].)

Mye arbeid er nedlagt i ulike bibliografiske prosjekter for å finne mønstre som gjelder for alle former for bibliografier. Man har gjort generaliseringer som skulle gjelde alle bibliografiformer, bare for, altfor ofte, å finne ut at disse generaliseringene ikke

stemmer – noen ganger bare for noen få bibliografityper, andre ganger for store grupper av bibliografier, og, en sjelden gang, for alle bibliografier bortsett fra den som man i øyeblikket ser på. Som regel er denne bibliografien på engelsk.

Tradisjonell bibliografisk programvare er laget for engelskspråklige bibliografier og fungerer derfor sjelden optimalt når den brukes for andre språk. I vitenskapelige arbeider produsert utenfor Storbritannia og USA er det dessuten ikke uvanlig at det refereres kilder på flere språk i samme bibliografi, og slik oppstår behovet for bibliografisk programvare for flerspråklige bibliografier.

For flerspråklige bibliografier er det vanskeligere å sikre gjenbruk av data, siden disse må foreligge på flere språk, skillet mellom form og innhold blir mindre klart, og det finnes en lang rekke nye stilistiske varianter som man må ta hensyn til.

1.1 Problemdefinisjon

Dagens programvare for formatering av bibliografier for bruk i typesettings- og dokumentprosesseringssystemer fungererer lite tilfredsstillende for flerspråklige bibliografier og for bibliografiske stiler som brukes utenfor engelskspråklige land. Eksisterende dataformater, programmer og modeller for bibliografiske data må tilpasses eller redefineres, men hvordan dette skal gjøres, er dårlig forstått.

1.2 Målsetning

Hensikten med prosjektet har vært å se på hvilke konsekvenser det har for den tradisjonelle ideen om en bibliografiprosessor når denne skal tilpasses flerspråklige bibliografier. Dette er realisert ved å utvikle en bibliografiprosessor og et dataformat for flerspråklige bibliografier, og å se på hvordan disse skiller seg fra eksisterende programmer og formater.

1.3 Forutsetninger

Det forutsettes at leseren har grunnleggende kjennskap til XML, XSLT, SAX, DOM og XPath, og det gis ingen bakgrunn for eller forklaring av detaljer knyttet til dette. Tilsvarende fordres det kunnskap om prinsippene bak TEX, oppmerkingsspråk og dokumentasjonsprosjekter som DocBook og TEI. Problemstillinger knyttet til Unicode og internasjonalisering forklares derimot, da jeg antar at få lesere har detaljerte kunnskaper om dette.

1.4 Terminologi og definisjoner

Det forekommer endel lingvistisk og typografiske terminologi i teksten, særlig i forbindelse med eksempler, og disse begrepene er forklart i fotnoter i den løpende teksten første gang de benyttes. Øvrige informatikkbegreper forklares ikke.

I teksten har jeg så langt mulig benyttet etablert norsk terminologi for tekniske begreper. I de tilfellene der den norske oversettelsen ikke er særlig utbredt eller der det kan råde tvil om den nøyaktige avgrensningen av begrepet, oppgir jeg ved første bruk det norske begrepet etterfulgt av det tilsvarende engelske begrepet i parenteser. Når oversettelser overhodet ikke finnes, har jeg benyttet engelske begreper, og disse er kursiverte.

1.5 Eksempler

Under diskusjonen av egenskaper ved bibliografier er eksempler flittig benyttet som illustrasjoner. Jeg har funnet at dette er den beste måten å presentere emnet på, da dette reflekterer den induktive arbeidsmåten som må benyttes ved innsamling av bibliografiske data. I den grad tiden har tillatt dette, har jeg benyttet virkelige eksempler tatt fra fagbøker og tidsskrifter fra et bredt fagområde og fra flere språk. Referanser til disse kildene er samlet i en separat liste etter oppgavens referanseliste.

1.6 Kapittelinndeling

Kapitlene 2 og 3 er ment å gi en oversikt over bakgrunnen for prosjektet. Kapittel 2 behandler ulike bibliografiske teknikker fra bibliografisk, lingvistisk og typografisk perspektiv, mens kapittel 3 beskriver eksisterende bibliografisk programvare og dataformater. Kapittel 4 gir en grundig innføring i Unicode og relevante problemstillinger som vil være av betydning senere.

Kapittel 5 beskriver den overordnede organiseringen av en flerspråklig bibliografiprosessor. Detaljene ved prosessoren utsettes; først beskrives et egnet dataformat i kapittel 6. I kapitlene 7–9 diskuteres så de enkelte komponentene av prosessoren i detalj.

Kapittel 10 omtaler den konkrete implementasjonen av bibliografiprosessoren, mens kapittel 11 avslutter med en vurdering av prosjektet.

Kapittel 2

Bibliografier på ulike språk

De fleste vitenskapelige tidsskrifter og forlag stiller idag egne krav til innholdet i og den visuelle utformingen av bibliografier. De faktiske opplysningene om bøker, tidsskrifter og andre media som det vises til, har begrenset variasjonsbredde, da denne informasjonen er knyttet til tradisjonene innen publikasjon og katalogisering av trykt materiale. Variasjonene innen presentasjonen av informasjonen er derimot svært stor.

Mye av denne variasjonen er knyttet til ulike tradisjoner og ulike behov innen forskjellige fagmiljøer. Det vil ikke være praktisk å foreta en uttømmende gjennomgang av alle disse; isteden vil vi først se på de vanligste teknikkene for å knytte de bibliografiske opplysningene til teksten og forsøke å klassifisere disse teknikkene. Deretter vil vi se på hvilke bibliografiske opplysninger det er vanlig å inkludere i bibliografier og deres forhold til ulike publikasjonstyper. Avslutningsvis skal vi se på hvordan disse teknikkene varierer mellom ulike språk og på spesielle problemstillinger som oppstår i forbindelse med ulike språk.

2.1 Siteringsteknikker og bibliografistiler

En **referanse** inneholder alle opplysninger som er nødvendige for å lokalisere et bestemt verk. Dette omfatter for eksempel alle forfatteres og redaktørers fulle navn, verkets tittel og tid og sted for utgivelse. En **referanseliste** er en liste med referanser og er gjerne samlet bakerst i et verk eller et bidrag i et verk. En **sitering** er en referanse i kortform eller en nøkkel satt parentetisk i den løpende teksten eller plassert i en note til teksten, gjerne forutgått eller etterfulgt av en kort tekst som modifiserer eller kommenterer siteringen.

Nesten enhver rasjonell teknikk som er klar og konsistent, kan brukes til for siteringer og referanser, men lesere er tjent med å møte kjente teknikker [69: 15.1], og utgivere krever derfor at forfattere følger kjente, og forholdsvis standardiserte,

Forfatter-dato-systemer	Siteringer i tekst med forfatter og dato. Referanselister ordnet etter forfatter og dato.
Nøkkel-systemer	Siteringer i tekst med nøkkel. Referanselister ordnet etter nøkler.
Forfatter-tittel-systemer	Siteringer i tekst eller i noter med enten hele referansen eller en forkortet form basert på forfatter og tittel. Hvis egen referanseliste brukes, er denne ordnet etter forfatter og tittel.

Tabell 2.1: Klassifikasjon av bibliografistiler.

systemer.

De fleste stilguider og standarder omtaler siteringsteknikker og bibliografistiler under ett og inndeler dem i tre hovedgrupper, dog med nokså ulike navn:

- Et system kalt *author-date* [69, 1], *first element and date method* [46] eller *name and date system* [9]: Siteringene er plassert parentetisk i teksten og inneholder forfatternavn og årstall. De viser til en referanseliste et annet sted i verket, og denne er ordnet etter forfatter og årstall. Jeg vil kalle disse systemene for **forfatter-dato-systemer**.

- Et system kalt *numbered references* [1], *author-number* [69], *numeric references method* [46] eller *numeric system* [9]: Siteringene er plassert parentetisk i teksen og inneholder en nøkkel, oftest et nummer. Denne nøkkelen kan brukes til å finne den tilsvarene referansen i en referanseliste plassert et annet sted i verket. Jeg vil benytte betegnelsen **nøkkel-systemer** om alle slike systemer.

- Et system kalt *documentary-note* [1], *humanities style* [1], *author-title* [69] eller *running notes* [9, 46]: Systemet finnes i to hovedvarianter, en variant med siteringene i teksten og en med siteringene i noter. Siteringene er enten fulle eller forkortede former av selve referansene. Systemet kan brukes både med og uten separate referanselister, de er i så fall ordnet etter forfatter og tittel. For disse systemene vil jeg benytte betegnelsen **forfatter-tittel-systemer**.

I tillegg kan man skille ut varianter som ikke benytter siteringer i teksten, men oppgir referanser i verkets forord eller lignende [56], men disse blir såpass spesielle og lite egnet for en bibliografiprosessor, at vi ser bort fra dem her.

La oss nå se nærmere på noen detaljer ved disse tre hovedsystemene. Dersom ikke annet er angitt, er opplysningene og eksemplene hentet fra [69] eller [1].

2.1.1 Forfatter-dato-systemer

Forfatter-dato-systemer har vært vanligst i natur- og samfunnsvitenskapelige fag, men er blitt mer utbredt også i humaniora [56]. Blant organisasjoner som benytter stiler

som denne, er American Psychological Association [2], og den anbefales også av flere for bruk i norske sammenhenger [44, 6, 71].

I den antagelig vanligste varianten av forfatter-dato-systemer, Harvard-systemet, gjøres referansene ved forfatters fulle etternavn og årstall omsluttet av parenteser eller klammer:

(Smith 1979)

Dette systemet ble først beskrevet i [1], men er aldri standardisert, og det finnes endel variasjoner:

Tegnsetting Det brukes enten parenteser eller klammer, og forfatternavnet skilles fra årstallet enten ved et enkelt mellomrom eller ved komma og mellomrom:

[Smith 1979]

(Smith, 1979)

Grupperte siteringer Grupperte siteringer kan ordnes kronologisk eller alfabetisk. Skilletegnet som benyttes mellom henvisningene, er vanligvis et semikolon eller et komma:

(Smith 1979; Johnson 1980)

(Johnson 1980, Smith 1979)

Flere medforfattere Ved flere medforfattere velger noen stiler å bruke kun det første forfatternavnet, andre lister opp alle, mens en tredje gruppe oppgir et bestemt antall navn (gjerne tre eller seks) og tilføyer *m.fl.*, *et al.*, *and others* og så videre. En fjerde mulighet er en fullstendig opplisting ved første sitering og en nedkortet utgave senere:

(Smith, Johnson, Kelley et al. 1981)

(Smith m.fl. 1981)

(Smith et al. 1981)

(Smith 1981)

Innen slike opplistinger varierer tegnsetningen og om tegnet *&* eller konjunksjonen *og* i en egnet språkform benyttes:

(Smith, Johnson, and Kelley 1982)

(Smith, Johnson og Kelley 1982)

(Smith, Johnson & Kelley 1982)

Forfattere med samme etternavn Når flere verker har forfattere med samme etternavn, benyttes fornavn eller initialer for å skille forfattere fra hverandre:

(J. F. Smith 1979; K. G. Smith 1979)

Samme forfatter og år Det skilles mellom siteringer av verker av samme forfatter utgitt samme år ved at små bokstaver fra det latinske alfabetet tilordnes i referanserekkefølge og plasseres etter årstallet. Hvorvidt det første verket i en slik liste skal suffigeres med en bokstav, om bokstaven kursiveres, og hvordan flere henvisninger til slike verker i én sitering håndteres, varierer:

(Smith 1979a)

(Smith 1979*b*)

(Smith 1979, 1979b)

(Smith 1979a, 1979b)

(Smith 1989a,b)

Inkorporerte siteringer Hvis forfatternavnet forekommer i teksten, kan siteringer gjøres ved årstallet alene:

Denne sammenhengen er vist av Smith (1979).

Siteringsavgrensninger Angivelse av sidetall, kapitler, avsnitt og lignende gjøres alltid innenfor siteringens parenteser eller klammer, men med ulik bruk av skilletegn og forkortelser:

(Smith 1979: 16–19)

(Smith 1979, 16–19)

(Smith 1979 pp. 16–19)

Forfatterløse verker For verker med korporativ forfatter, uten forfatter, flerbindsverker, antologier, essaysamlinger og lignende må en annen opplysning brukes istedenfor forfatters etternavn. Valget av dette avhenger av hva som velges som katalogiseringsord i referanselisten.

2.1.2 Nøkkel-systemer

Nøkkel-systemer har stort sett vært begrenset til tidsskrifter [69: 15.1] – mye på grunn av at systemet kan være arbeidskrevende å bruke for større publikasjoner [1: 15.3] – men systemet er populært innen flere naturvitenskapelige områder og i sammenhenger der litteraturlistene er små.

Blant nøkkel-systemene er det to undertyper: De systemene som benytter referanselister ordnet alfabetisk etter forfatternavn og de som benytter referanselister ordnet i siteringsrekkefølge [5]. Sistnevnte kalles for **Vancouver-systemet**, og er standard innen blant annet medisin [56]. Det kan illustreres ved dette tekstutdraget (fra [93]):

> «The reader is referred to [1] for a bibliography [. . .] Consequently, membrane (2D) models giving a realistic prediction of the dog bone defect have been developed by d'Halewyn *et al.* [2] for a Newtonian fluid and by Debbaut *et al.* [3] for a viscoelastic fluid.»

Legg merke til at numrene øker for hver sitering, og at dette fører til at referanselisten ikke er sortert etter forfatter:

> 1. D. Silagy, Y. Demay and J.F. Agassant, 'Etude de la stabilité de l'étirage d'un fluide Newtonien', *C. R. Acad. Sci. Paris II*, **322**, 283–289 (1996).
> 2. S. d'Halewyn, Y. Demay and J.F. Agassant, 'Numerical simulation of the cast film process', *Polym. Eng. Sci.*, **30**, 335–340 (1990).
> 3. B. Debbaut and J.M. Marchal, 'Viscoelastic effects in film casting', *Z. Angew Math. Phys.*, **46**(special issue), 679–698 (1995).

I andre varianter av nøkkel-systemene alfabetiseres referansene etter forfatter, og flere referanser med samme forfatter ordnes kronologisk (eksempel fra [84]):

> [4] L. Boxer and R. Miller, "Parallell Dynamic Computational Geometry," Technical Report TR 87-11, State University of New York, Buffalo, 1987.
>
> [5] L. Boxer and R. Miller, "Common Intersections of Polygons," *Information Processing Letters*, vol. 33, no. 5, pp. 249–254, 1988.

I mange av nøkkel-stilene settes nøklene i klammer eller parenteser (fra [84]):

> «The problem of finding the upper envelope of segments is fundamentally important in computational geometry and has applications in visibility, motion planning, convex hulls, construction of arrangements, and polygon containment [2], [5], [12], [17].»

Enkelte stiler, særlig Vancouver-stiler, benytter isteden en annen skrifttype (fra [82]):

> «Such a proof is unavailable for gapped local alignments, but computational experiments strongly suggest that the same type of distribution applies[10].»

eller et nøkkelord etterfulgt av et nummer (fra [91]):

> «For other array-based assays, such as pre-spotted filter arrays and Affymetrix GeneChips™ (REF. 2), the researcher has little, if any, control over the probe content of the chip.»

Grupperte siteringer med numre kan nedkortes når numrene er sekvensielle, for eksempel:

> [1, 4, 5, 6]
>
> [1, 4–6]

Siteringsavgrensninger kan også gjøres på ulike måter:

> [1, s. 14]

[1: 14]

Varianter som benytter andre nøkler enn numre, er sjeldne, men forekommer for eksempel med egne nummersekvenser for hver forfatter [10]. En annen mulighet er at nøkkelen består av en forkortet utgave av forfatteres etternavn og årstall [68], for eksempel (fra [87]):

> «This collection therefore constitutes a comparative case study in the use of formal methods, and adds to the growing body of work of this kind (e.g., [ABL96, BoG00, BMS96, FrH01, LeL95]) which serve an important purpose in broadening awareness of the scope and possibilities of formal methods.»

Nøklenes sammensetning avhenger av antall forfattere i angitt i referansen:

> [Bog00] Boerger, E. and Gotzheim, R.: Requirements engineering case study: light control. *Journal of Universal Computer Science*, 6(7): 2000.
>
> [BMS96] Broy, M., Merz, S. and Spies, K.: *Formal Systems Specification: The RPC-Memory Specification Case Study*. Number 1169 in Lecture Notes in Computer Science. Springer-Verlag, 1996.

2.1.3 Forfatter-tittel-systemer

Disse systemene er vanlige innen humaniora, og den dominerende moderne varianten i engelskspråklige land er MLA-systemet (se [26]). Dette benytter korte, parentetiske siteringer med forfatters etternavn og, om nødvendig, verkets tittel i teksten (alle eksempler fra [26]):

> (Marcuse 197)
>
> (K. Roemer 123-24)
>
> (Lauter et al. 2425-33)
>
> (Public Agenda Foundation 4)
>
> (Kaku 42; McRae 101-53)

Andre varianter av forfatter-tittel-systemet benyttes med noter, og da plasseres siteringene enten i fotnoter eller i sluttnoter bakerst i verket eller i slutten av et kapittel. Notene inneholder gjerne kommentarer til teksten i tillegg til siteringene, og en note kan gjerne inneholde flere siteringer eller ingen sitering i det hele tatt.

Ved bruk med referanseliste kan hver sitering være en kortform av referansen bygget rundt forfatter og tittel, mens ved bruk uten referanseliste må første sitering inneholde hele referansen [5] (eksempel fra [85] som benytter siteringer i noter og har separat referanseliste):

Forkortelse	Betydning	Bruksområde
op. cit.	*opere citato* «i det siterte verk» eller *opus citatum* «det siterte verk»	Benyttes når man siterer et foregående sitert verk, bortsett fra når dette forekommer i den umiddelbart foregående siteringen. Ledsages av forfatternavn og sidetall.
ibid. eller *ib.*	*ibidem* «på det samme sted»	Refererer til den samme forfatter og det samme verk som i den umiddelbart foregående siteringen. Ledsages oftest av sidetall.
loc. cit. eller *l.c.*	*loco citato* «på det siterte sted»	Benyttes når man siterer samme sted, oftest samme side, som i en foregående sitering. Ledsages av forfatternavn, men kan stå alene hvis siteringen viser til den umiddelbart foregående siteringen.
	passim «her og der»	Angir at det siterte materialet forekommer flere steder igjennom verket.

Tabell 2.2: Latinske forkortelser og uttrykk i forfatter-tittel-systemet. Hentet fra [56], [44] og [26].

> 28. Panofsky, introduction to *Studies in Iconology: Humanistic Themes in the Art of the Renaissance* (1939; reprint, New York, 1962), 30.
>
> [. . .]
>
> 39. Panofksy, "The History of Art as a Humanistic Dicipline," in *Meaning in the Visual Arts* (Garden City, 1955), 16–19.
>
> [. . .]
>
> 41. Panofsky, "The History of Art as a Humanistic Dicipline," 14.
>
> [. . .]
>
> 49. Panofsky, introduction to *Studies in Iconology*, 11, n.3.

Særlig i eldre utgaver av systemet er det en utstrakt bruk av latinske forkortelser (se tabell 2.2 for oversikt), for eksempel (fra [90]):

> 5. *The Collected Works of William Morris*, London, 1915, xxiii, p. 147.
>
> 6. Ibid., 1914, xxii, p. 26.
>
> 7. Mackail, op. cit., ii, p. 99.
>
> 8. *Coll. Works.*, xxii, p. 42; xxiii, p. 173.
>
> 9. Ibid., xxii, pp. 47, 50, 58, 73, 80, etc.

Legg merke til hvordan den sjette noten introduserer et nytt bind av samleverket fra note fem ved hjelp av nummeret på bindet og utgivelsesåret. Når det samme bindet siteres på nytt i note åtte, er årstallet ikke lenger nødvendig. Derimot er det ikke mulig å gjenta bruken av *ibid.* i note åtte, da et annet verk er sitert i mellomtiden.

Hvis referanselister benyttes i forfatter-tittel-systemet, er disse ofte kommenterte, og referansene kan være inndelt etter emner eller en annen relevant klassifikasjon. I

slike sammenhenger kalles listene gjerne **bibliografier**. Et eksempel fra en tematisk bibliografi er gjengitt nedenfor (fra [85]):

> Burckhardt, Jacob. *Der Cicerone: Eine Anleitung zum Genuss der Kunstwerke Italiens* (1855). 4th ed. Ed. Wilhelm Bode. Leipzig, 1879.
>
> ———. *The Civilization of the Renaissance in Italy* (1860). 2 vols. Ed. Benjamin Nelson and Charles Trinkaus. 1929. Reprint. New York, 1958.
>
> ———. *Force and Freedom: Reflections on History* (1871). Ed. and trans. J. H. Nicols. New York, 1943.
>
> ———. *Gesamtausgabe* 14 vols. Berlin, 1929–1934.

I dette verket er siteringene plassert i noter samlet bakerst i verket under overskriften «*Notes*»:

> 20. Jacob Burckhardt, *Force and Freedom: Reflections on History*, ed. and trans. J. H. Nichols (New York, 1943), 80–82. For the complete edition of Burckhardt's work in German, see the fourteen-volume *Jacob Burckhardt: Gesamtausgabe* (Berlin, 1929–1934).
>
> [. . .]
>
> 24. Burckhardt, *Force and Freedom*, 80.

2.2 Bibliografiske opplysninger

I praktisk arbeid med referanselister viser det seg ofte at de fleste referanser enten er bøker eller tidsskriftartikler. Denne observasjonen kan motivere en mer formell oppdeling i publikasjoner som utgis én gang og publikasjoner som publiseres periodisk. ISO 690 [46] og BS 5605 [9] skiller mellom nettopp disse to typene: ***Monograph*** defineres som en publikasjon som er komplett alene eller i et endelig antall adskilte deler, mens ***serial*** er en trykt eller ikke-trykt publikasjon utgitt i suksessive deler, gjerne med numeriske eller kronologiske betegnelser, som skal fortsettes periodisk. I vår diskusjon vil vi benytte begrepene **monografier** og **periodika** (**periodikum** i entall) for disse publikasjonstypene.

Sammenligner man monografier og periodika, ser man at deler av monografier, for eksempel artikler i referanseverker eller bidrag i en antologi, kan likestilles med artikler i periodika. Referanseinformasjonen faller i disse tilfellene inn i samme hovedmønster: Først en angivelse av den hovedansvarlige (vanligvis en eller flere forfattere), deretter tittel og til slutt selve den bibliografiske informasjonen [77]. Den bibliografiske informasjonen avhenger av publikasjonstypen: for monografier angis utgave, utgivelsessted, utgiver og utgivelsesår; for periodika angis sjelden sted eller utgiver, men derimot årgang, eventuelt nummer og dato. Ved deler av monografier og artikler i periodika vil vertsdokumentets samlede bibliografiske opplysninger, altså inkludert tittel, inngå som del av den bibliografiske informasjonen. Som et eksempel på denne modellen, kan følgende referanse i MLA-stil

> Beebe, Nelson. "Bibliography prettyprinting and syntax checking." *TUGboat*, 14 (1993): 395–419.

oppstilles som

Ansvarlig	Nelson F. H. Beebee (forfatter)
Tittel	Bibliography Prettyprinting and Syntax Checking
Bibliografisk informasjon	
Vertsdokument	TUGboat, 14. årg., nr. 4, desember 1993
Avgrensning	Fra side 395–419

mens et kapittel i en bok med referansen (fra [69: 15.3])

> John Shearman, 'The Vatican Stanze: Functions and Decoration', in George Holmes (ed.), *Art and Politics in Renaissance Italy: British Academy Lectures* (Oxford: Clarendon Press, 1993), 185–240.

oppstilles som

Ansvarlig	John Shearman (bidragsforfatter)
Tittel	The Vatican Stanze: Functions and Decoration
Bibliografisk informasjon	
Vertsdokument	
Ansvarlig	George Holmes (redaktør)
Tittel	Art and Politics in Renaissance Italy: British Academy Lectures
Bibl. info.	Oxford, Clarendon Press, 1993
Avgrensning	Fra side 185–240

Denne oppdelingen i hoveddokument og vertsdokument gjenfinnes i mange publikasjonstyper. Det er mulig med flere nivåer, og generelt refereres slike oppdelinger til som **bibliografiske nivåer**.

De fleste publikasjoner vil passe inn i dette enkle mønsteret, men for dem som ikke gjør det, er det naturlig å forsøke å etterligne grunnoppsettet så langt det lar seg gjøre. Elektroniske publikasjoner blir for eksempel en stadig viktigere gruppe, og for disse er særlig publiseringsinformasjonen vanskelig. Anbefalinger varierer, men inkludert er gjerne sponsende organisasjon, versjonsnummer, dato for første versjon, siste oppdatering, dato for aksess og elektronisk adresse [26]. Avgrensning er derimot mindre relevant for elektroniske publikasjoner, da disse sjelden organiseres i enheter som sider eller kapitler.

2.3 Referanselister og bibliografier

Grovt sett brukes betegnelsen **referanseliste** om litteraturlister som kun inneholder opplysninger om siterte verker, mens en **bibliografi** også omfatter annen relevant eller konsultert litteratur. Dette er forbundet med de ulike bruksmønstrene for siteringssystemene – referanselister er naturlig knyttet til forfatter-dato- og nøkkel-systemene, mens bibliografier er knyttet til forfatter-tittel-systemer. Bibliografier kan oppdeles i flere undergrupper etter tematikk, kildetyper og så videre, mens slike oppdelinger vanligvis ikke gjøres for referanselister [69: 15.17].

Det er mulig med flere referanselister i ett dokument. Disse opptrer vanligvis i form av en referanseliste for hvert kapittel i en bok, en referanseliste for hver artikkel i en artikkelsamling eller flere tematisk inndelte referanselister i en bok.

2.4 Språkhensyn

I flerspråklige bibliografier viser det seg at presentasjonsformen til mange av de bibliografiske opplysningene er avhengig av språk. Også typografiske konvensjoner som tegnsetting og skrifttyper, er språkavhengige, og vi vil nå se nærmere på det viktigste av disse avhengighetene.

2.4.1 Nøkkelord og forkortelser

Nøkkelord som «og», «utgave» og «redaktør» forekommer hyppig i referanselister og må finnes i oversettelser:

> Volume 15, number 3, pages 269–273.

> Årgang 15, nummer 3, sidene 269–273.

Merk at disse nøkkelordene ofte er forkortet:

> Vol. 15, no. 3, p. 269–273.

> Årg. 15, utg. 3, s. 269–273.

Flere konkurrerende forkortelsessystemer er ikke uvanlig, og valget av system er stilavhengig. For eksempel foretrekker enkelte engelskspråklige stiler forkortelsen *eds.* for flertallsformen *editors*, mens andre benytter *ed.* for både entalls- og flertallsformen.

2.4.2 Stedsangivelser

Ved angivelse av utgiver oppgis oftest også utgivelsesstedet. Dessverre er stedsnavn språkavhengig, da mange kjente byer har forskjellige navn på ulike språk, for ek-

sempel engelsk *Munich* for tysk *München*. Strengt tatt er stedsnavn også avhengig av historisk sammenheng, da de kan ha skiftet navn gjennom historien [69: 15.28].

Stedsnavn kan også være flertydige, slik som *Cambridge*, som enten er en by i Storbritannia eller i Massachusetts i USA. For å unngå denne flertydigheten, må enkelte stedsnavn kvalifiseres med land og eventuell delstat eller provins, men hvilke stedsnavn som skal kvalifiseres, er språk- og stilavhengig. Dette gjelder også stedsnavn som ikke er flertydige: Stedsnavnet *Bodø* vil ikke trenge kvalifikasjon i en norsk tekst, men i andre tilfeller kan det være nødvendig.

Stedsnavn forkortes også ofte i enkelte språk, for eksempel i denne referansen hvor *Sankt Petersburg* er forkortet til *СПб* (fra [94]):

> 107. Гуревич Л Э, Румянцев А А *Письма в Астрон. журн.* **4** 505 (1978): Горбацкий В Г *Газодинамические неустойчивости в астрофизических системах* (СПб: Изд-во СПбГУ, 1999)

2.4.3 Navn på utgivere og tidsskrifter

I noen referansestiler foretrekkes det at navn på forleggere oppgis i en forkortet form, for eksempel *Cambridge UP* for *Cambridge University Press* [26: 8.5]. Det samme er svært utbredt for tidsskriftnavn (fra [88]):

> 2. Dumas, J. P. & Ninio, J. (1982) *Nucleic Acids Res.* **10**, 197–206.
>
> 3. Wilbur, W. J. & Lipman, D. J. (1983) *Proc. Natl. Acad. Sci. USA* **80**, 726–730.

Hvorvidt slike forkortelser er brukbare, vil variere fra språk til språk.

2.4.4 Datoer

Datoer kan forekomme på mange former, for eksempel *16. Jahrhundert* eller *Spring 1576*, begge språkavhengige, men også ordinære datoangivelser med dag, måned og år, er språkavhengige, og de kan ikke alltid oversettes ord for ord: 17. juni 2000 er 17 հունիսի 2000թ på armensk og *2000. június 17-e* på ungarsk [70].

2.4.5 Tall

Tallord og andre mengdeord kan påvirke påfølgende ord. Dette manifisereres i norsk ved veksling mellom entall og flertall: *1 del*, men *2 deler*. I andre språk skilles det mellom flere former, slik som kasusformer i de slaviske språkene, for eksempel russisk один том «et bind», два тома «to bind» og пять томов «fem bind» [60].

2.4.6 Translitterasjon og oversettelser

I generelle arbeider vil det være behov for å translitterere eller transkribere[1] navn og titler fra språk som benytter fremmede alfabeter [1: 9.86], særlig hvis leseren antas å være ukjent med disse eller hvis dette gjør det enklere å finne arbeidet (eksempel fra [81]):

> **Alemayehu, N.** (1999). *Development of a Stemming Algorithm for Amharic Language Text Retrieval*. Ph.D. Thesis, University of Sheffield.
>
> **Amare, G.** (1990EC). ዘመናዊ የአማረኛ ሰዋሰው በቀላል አቀራረብ. አዲስ አበባ፡ ንግድ ማተሚያ ቤት. (Zemenawi yeamareNa sewasew beqelal aqerareb. Adis Abeba: ngd matemiya bEt.)

Det finnes ofte flere konkurrerende translitterasjonssystemer, for eksempel er Mao Tse-Tung og Mao Zedong mulige translitterasjoner med henholdsvis Wade-Giles-systemet (韋氏 wéishì) og pinyin (拼音 pīnyīn) [67]. Det er også vanlig at ulike systemer benyttes i ulike land, for eksempel ved transkripsjon av kyrillisk:

Russisk	Фёдор Михаилович Достоевский
Engelsk	Fyodor Mikhailovich Dostoyevsky
Tysk	Fjodor Michailowitsch Dostojewski
Norsk	Fjodor Mikhailovitsj Dostojevskij
ISO 9[2]	Fëdor Mihailovič Dostoevskij

I tillegg til ren translitterasjon er det vanlig å oversette titler og eventuell annen informasjon som er skrevet på språk som leseren antagelig ikke forstår (eksempel fra [67]):

> Fu Maoji. 1981. *Naxi-yu tuhua-wenzi "Bai bianfu qu jing ji" yanjiu* [A study of a Naxi pictographic manuscript, "White Bat's Search for Sacred Books"], Vol. I. CAAAL Monograph Series, no. 6. Tokyo: CAAAL.

2.4.7 Tegnsetting

Bruk av tegnsetting og foretrukne tegn varierer mellom språk. For eksempel er det svært varierende hva de foretrukne sitattegnene er (eksempel tatt fra [42]):

Amerikansk engelsk	"Brand of the Werewolf"
Britisk engelsk	'Brand of the Werewolf'
Fransk	« La marque de la bête »
Tysk	„Im Zeichen des Werwolfs"

[1] Strengt tatt er **translitterasjon** en tegnvis tapsfri avbildning fra et skriftsystem til et annet, mens en **transkripsjon** er en lydvis avbildning til et annet språk. I moderne gresk uttales bokstavene η, ι, υ og kombinasjonene ει, οι og υι alle som [i], og man kan derfor finne Ελληνικη Δημοκρατια translitterert *Ellēnikē Dēmokratia* eller transkribert *Elliniki Dimokratia*. I praksis brukes begrepene noe om hverandre, og i denne oppgaven benyttes begrepet translitterasjon i alle sammenhenger.

[2] ISO 9 [45] angir en transliterasjon identisk med eller svært lik den som ofte benyttes av lingvister og slavister. Karakteristisk er bruken av tegn som *š*, *č* og *ž* istedenfor bokstavkombinasjoner som *sj*, *tsj* og *zj*, og markering av harde og bløte tegn ved hjelp av apostrofer, for eksempel *Gogol'* for *Гоголь*.

Legg også merke til at det er mer luft rundt rundt *guillemets* (det vil si tegnene « og ») i den franske oversettelsen av tittelen, enn det ville være i en norsk oversettelse.

2.4.8 Tittelformer

Bruken av versaler[3] i ord varierer i bikamerale[4] skriftsystemer. Utover praksisen med versaler i begynnelsen av ord for egennavn, finnes flere språk som også benytter versaler i begynnelsen av ord fra bestemte ordklasser (for eksempel tysk og eldre dansk-norsk), men også språk som benytter tilsvarende systemer utelukkende for titler og overskrifter (for eksempel engelsk), såkalte **tittelformer**. Dette er først og fremst et fenomen knyttet til romersk-latinsk-skrift, men andre bikamerale skriftsystemer kjenner også tilsvarende distinksjoner. Spesielt kan nevnes at bruk av store forbokstaver i engelske titler ikke benyttes i alle referansestiler, men er mest vanlig i forfatter-tittel-systemer [1: 15.104, 15.73.3].

2.4.9 Alfabetiseringsregler

Referanselister og bibliografier sorteres alfabetisk i de fleste referansesystemene, men ulike språk har ulike regler for alfabetiseringen. Nærliggende eksempler på dette finnes innen de nordiske språk: Svensk alfabetiseringsrekkefølge er *å*, *ä*, *ö*, men i norsk alfabetiseres *å* etter *æ* og *ø*, mens *ä* og *ö* fra svensk vil alfabetiseres sammen med *æ* og *ø* [74].

Ytterligere komplikasjoner oppstår når fremmedspråklige verker opptrer i listene. Skal et tysk verk med tittel som begynner med *ö*, alfabetiseres etter uttale (det vil si som *ø*), etter tysk ordbokkonvensjon (det vil si som *o*) eller som *oe*, slik praksis er i tyske telefonkataloger?

Ikke bare kan bokstavenes innbyrdes rekkefølge være ulik, men det kan finnes bokstavsekvenser som teller som ett selvstendig symbol under alfabetiseringen. For eksempel ble bokstavkombinasjonene *ll* og *ch* inntil nylig behandlet som digrafer[5] i spansk og opptrer derfor som egne symboler i de fleste spanske ordlister.

Tilsvarende problemstillinger finnes i andre alfabet- eller stavelsesskriftsystemer. I systemer som benytter ideografiske tegn,[6] er det ofte flere konkurrerende metoder

[3] **Versaler** (eller **majuskler**) betegner «store bokstaver», og **gemene** (eller **minuskler**) brukes om «små bokstaver» i skriftspråk der dette er et aktuelt skille.

[4] **Bikameral skrift** betegner typer av skrift som skiller mellom versaler og minuskler. Dette skillet finnes i romersk-latinsk, gresk, kyrillisk, armensk og georgisk skrift.

[5] En **digraf** er en gruppe av to tegn som til sammen betegner én språklyd og som oftest også behandles som et selvstendig tegn i sortering, orddeling og ved forkortelser. For eksempel er *dž* en digraf på serbo-kroatisk, og i en ordliste er alle ord som begynner med *dž*, plassert etter samtlige ord som begynner med *d*.

[6] **Ideografiske tegn** brukes her om alle skriftsymboler som gjengir hele begreper istedenfor bestemte språklyder.

for sortering. Disse kan være basert på tradisjonelle tegnrekkefølger, slik som for radikalene i radikal- og strek-sortering,[7] eller på uttale, og man måtte konsultere ordbøker for å sortere disse riktig,

Korrekt sortering kan også kreve mer informasjon enn den som er tilgjengelig gjennom skriften. Sekvensen *aa* i skandinavisk er et eksempel på dette; den skal noen ganger sorteres som *å*, andre ganger som *aa*.

Flerspråklige bibliografier vil også ofte inneholde referanselister med flere skriftsystemer i én liste. Det er ikke opplagt hvordan dette skal håndteres, men to vanlige metoder er enten å sortere referansene som benytter fremmede skriftspråk, for seg eller å sortere dem etter translitterasjon.

Sortering kompliseres videre av at det kan finnes prefikser eller ord som ignoreres under sorteringen, for eksempler artikler.

Det er også ulike syn på betydningen av mellomrom og skilletegn. Ved **ord-for-ord-sortering** avbrytes sorteringen ved slutten av det første ordet. Neste ord teller kun hvis det finnes flere forekomster av det samme ordet. Ved **bokstav-for-bokstav-sortering** fortsetter derimot sorteringen over ordgrenser. I begge tilfeller regnes det meste av tegnsetning som del av et ord. [1: 17.97] illustrerer dette ved å sammenligne bokstav-for-bokstav-sorteringen

newborn
new economics
newlywed
new math
New Testament

med ord-for-ord-sorteringen

new economics
new math
New Testament
newborn
newlywed

2.4.10 Navn

Personnavn er svært viktige i bibliografiske opplysninger og opptrer i flere former både i siteringer og referanselister. Navnetradisjoner er mangfoldige og finnes i mange varianter gjennom historien. Selv «enkle» navnesystemer som vestlige navnesystemer, kan være vanskelige på grunn av partikler som *de*, *von* og *della*. Disse regnes tidvis som

[7] **Radikal- og strek-sortering** betegner en av flere mulige sorteringssystemer i bruk for ulike typer kinesiske tegn i japansk (漢字 kanji), kinesisk (汉字/漢字 hànzì), koreansk (한자/漢字 hanja) og vietnamesisk (chữ Hán og chữ Nôm). Tegnene er sammensatt av et **radikal** og et **fonetikum**, der antallet radikaler er begrenset til et par hundre. Tegnene ordnes først etter en tradisjonsbundet rekkefølge for radikalene, og deretter etter antall streker i tegnets fonetikum.

del av etternavnet, andre ganger som en egen navneenhet. Tradisjonelle europeiske navnesystemer er enda vanskeligere å behandle, for eksempel består spanske familienavn av både fars og mors familienavn, slik som *García Lorca* i *Federico García Lorca*, og de fungerer begge som etternavn. Derimot er et russisk patronym en del av fornavnet, for eksempel *Mikhailovitsj* i *Fjodor Mikhailovitsj Dostojevskij*.

Ved alfabetisering kan det også kreves at navn skal inverteres, det vil si at etternavn skal plasseres foran fornavn for enklere oppslag i listene:

Smith, J. F. & K. G. Johnson

Smith, J.F. & Johnson, K. G.

I asiatiske språk følger fornavnet gjerne familienavnet, og dette har følger for reglene for den inverterte formen av navn.

Kapittel 3

Eksisterende programmer og formater

Det kan være et tidkrevende og vanskelig arbeide for forfattere og redaktører å lage bibliografier, da det finnes et mangfold av bibliografiske stiler, og det er vanskelig å være konsistent når man siterer og fører opp referanser. Å vedlikeholde en samling med referanser for bruk i flere publikasjoner, er heller ingen enkel sak [66].

For å gjøre dette arbeidet enklere, er det allerede utviklet mye bibliografisk programvare og mange ulike dataformater som benyttes for utveksling eller lagring av bibliografiske opplysninger. En omfattende oversikt over bibliografisk programvare ble laget i 1994, og denne omtaler et førtitall ulike produkter for formatering av bibliografiske data [73]. Så vidt jeg har forstått, finnes majoriteten av disse produktene fremdeles, og dessuten er det tilkommet en lang rekke nye *open source*-prosjekter.

Først skal vi se på endel av de mest utbredte formatene for representasjon av referansedatabaser. De eldre av disse ble oftest utviklet for bruk sammen med et bestemt program, mens nyere formater gjerne utvikles som separate prosjekter med sikte på å lage *de facto* standardformater for referansedatabaser. Jeg vil kalle disse formatene for **utvekslingsformater**, da de ofte brukes for utveksling av referansedatabaser, til forskjell fra **internformater**, som kun brukes som internt dataformat i bibliografiprogramvare.

3.1 BibTeX-databaser

BibTeX er en bibliografiprosessor og et filformat laget av Oren Patashnik og Leslie Lamport i 1985 for TeX og LaTeX. Det er svært mye brukt, særlig i naturvitenskapelige miljøer, og selv om det er et nokså primitivt system, fungerer det meget godt – iallfall i angloamerikansk språksammenheng.

Filformatet består av ren tekst i en ikke nærmere bestemt koding. Hver referanse innledes med en angivelse av referansetype (se tabell B.3). Selve referansen består av felter; disse har et navn og en verdi adskilt av et likhetstegn (se figur 3.1 for et eksem-

```
@Book{aho86,
  author = { Alfred V. Aho and Ravi Sethi and Jeffrey D. Ullman },
  title = { Compilers, Principles, Techniques and Tools },
  publisher = "Addisson-Wesley",
  address = { Reading, MA, USA },
  year = 1986
}
```

Figur 3.1: Eksempel på en BibTEX-referanse. *@Book* angir referansetypen. Etter denne følger en unik identifikator, og deretter feltene i vilkårlig rekkefølge. Legg merke til at flere forfattere angis ved å sette *and* mellom navnene deres.

pel). Verdiene kan inneholde TEX-kommandoer, og de kan være omsluttet enten av anførselstegn eller av krøllparenteser. I tillegg har hver referanse en unik identifikator som benyttes i siteringskommandoene til å knytte sammen siteringer og referanser.

For BibTEX versjon 0.99b er det spesifisert 24 ulike felter (se tabell B.1). For hver referansetype er utvalg av disse feltene enten obligatoriske, valgfrie eller ubrukte [61]. Dessverre har mange funnet at dette ikke er tilstrekkelig, og endel uoffisielle tilleggsfelter er derfor oppstått (se tabell B.2).

Svakhetene ved feltrepertoaret viser seg også ved at syntaksen til enkelte felter er for fri. Særlig gjelder dette personers navn, der fordelingen av *whitespace* i BibTEX-filen og implisitte regler om hva som er for-, mellom- og etternavn kan gi uventede resultater. De to BibTEX-referansene

```
@Book{gombrich95a,
  author =       {E.H. Gombrich},
  title =        {The Story of Art},
  publisher =    {Phaidon},
  year =         1995,
  address =      {Oxford},
  edition =      {16th ed.}
}

@Book{gombrich95b,
  author =       {E. H. Gombrich},
  title =        {The Story of Art},
  publisher =    {Phaidon},
  year =         1995,
  address =      {Oxford},
  edition =      {16th ed.}
}
```

er identiske, bortsett fra notasjonen av forfatternavnet. Med stilen `siam.bst` blir disse seende slik ut i en referanseliste:

[1] E. Gombrich, *The Story of Art*, Phaidon, Oxford, 16th ed. ed., 1995.

[2] E. H. Gombrich, *The Story of Art*, Phaidon, Oxford, 16th ed. ed., 1995.

Dette er en sammenblanding av presentasjon og innhold, som også opptrer i forbindelse med konvensjoner knyttet til bruken av strengkonstanter og regler for bruken av store og små bokstaver i titler.

Grammatikken for filformatet er noe uklar, da BibTeX-dokumentasjonen ikke omfatter en fullstendig beskrivelse av den [41]. Det viser seg også at den gjør det vanskelig å utføre kontekstuavhengig leksikalsk analyse, og at det kan oppstå feilkaskader [4].

3.2 Andre feltbaserte tekstformater

I tillegg til BibTeX-formatet finnes en lang rekke andre feltbaserte formater. Disse følger stort sett det samme prinsippet som BibTeX-formatet: Hver referanse består av en mengde felter med et feltnavn og en verdi. Formatene er oftest ikke formelt spesifisert og støtter ikke andre tegnkodinger enn ASCII på en sikker måte.

Programmene *refer*[1] og *Tib*[2] er to eldre bibliografiprosessorer laget henholdsvis for dokumentformateringssystemet `troff` og for typesettingssystemet TeX. Begge programmene leser siteringer fra et dokument og bruker disse til å lage referanselister fra referansedatabaser. Programmene bruker omtrent samme syntaks i referansedatabasene; dette er en gyldig referanse for både *refer* og *Tib*:

```
%A Edsger W. Dijkstra
%T Go To Statement Considered Harmful
%J Communications of the ACM
%V 1
%N 3
%P 147-148
%D 1968
```

Svært likt er formatet RIS. Dette har flere felter enn *refer* og *Tib* (se tabell B.5), det omfatter også referansetyper (se tabell B.4), og det benytter en litt annen syntaks, men ellers er forskjellene små:

```
TY  - BOOK
T1  - Social psychology : exploring universals across cultures
A1  - Moghaddam, Fathali M.
CY  - New York
PB  - W.H. Freeman
Y1  - 1998
N1  - Bibliografi: s. 550-588
SN  - 0-7167-2849-4 (ib.)
```

[1] Del av programmet *groff* skrevet av James Clark. Tilgjengelig fra for eksempel `ftp://prep.ai.mit.edu/pub/gnu/` eller et annet speil av GNU-programmene.

[2] Tilgjengelig fra CTAN under `biblio/tib/`. Skrevet av James C. Alexander.

```
KW  - sosialpsykologi kultur kjønn attribusjon grupper relasjoner
psykologi kulturer krysskulturell
KW  - Social psychology
ER  -
```

RIS brukes fremdeles mye, ettersom mange populære programmer kun kan importere og eksportere RIS-filer i tillegg til sine internformater. Eksempelet ovenfor er tatt fra det norske BIBSYS-systemet, som også kan eksportere RIS-filer.

MARC er et vanlig katalogiseringsformat for biblioteker. Det er et omfattende format, som kan inneholde svært detaljerte bibliografiske opplysninger. Biblioteker har gjerne egne tilpassede versjoner av MARC – de er basert på samme hovedformat, men de tillatte feltene og lovlig innhold er justert. Eksempelet nedenfor er et utdrag fra MARC-posten generert av BIBSYS for verket i forrige eksempel:

```
*001980787246
*008                                      eng
*020  $a0-7167-2849-4$bib.
*082  $c302
*082d $a302
*082kj$a302$b155.8
*082uc$a302
*082uk$a302$b155.8
*082ur$a302
*082uv$a302
*082xc$a302
*082xs$a302
*085a $aCc 680
*100  $aMoghaddam, Fathali M.
*245  $aSocial psychology$bexploring universals across cultures$
cFathali M. Moghaddam
*260  $aNew York$bW.H. Freeman$c1998
*300  $aXXIII, 610 s.$bill.
*500  $aBibliografi: s. 550-588
*650  $aSocial psychology
*691**$asosialpsykologi kultur kjønn attribusjon grupper relasjoner
psykologi kulturer krysskulturell
```

Vi ser at MARC benytter tallkoder istedenfor feltnavn, men at innholdet stort sett er det samme. Den vesentligste forskjellen er at MARC-posten inneholder en lang rekke felter som klassifiserer verket etter Dewey-systemet.[3]

[3] ***Dewey Decimal Classification system*** (DDC) er et system for organisering av kunnskap og er idag i bruk i biblioteker over hele verden.

3.3 Referansedatabaser i XML

I den senere tid er det dukket opp flere nye filformater for referansedatabaser basert på XML. En umiddelbar fordel med dette er at tegnkoding ikke lenger er noe problem, ettersom denne skal angis i XML-dokumenters innledningsdel, og samtidig elimineres de fleste problemer knyttet til spesifikasjon av grammatikker.

Prosjektene som utvikler disse XML-formatene, beskriver gjerne formatene ved å publisere DTD'er, mens dokumentasjonen er sparsom. Jeg vil derfor bruke begrepet *DTD* vekselvis om selve DTD'en og om informasjonsmodellen den beskriver.

DTD'ene kan inndeles i fire grupper på bakgrunn av målsetning for prosjektet og miljøet prosjektet springer ut fra:

- DTD'er som søker å representere BibTeX-databaser mer eller mindre direkte som XML.
- DTD'er som søker å lage komplette bibliografimodeller uten tilknytning til eldre formater.
- DTD'er som er komponenter i større dokumentformater.
- DTD'er som er komponenter i større katalogiserings- eller metadataprosjekter.

3.3.1 DTD'er for BibTeX

bibteXML[4] søker å lage et skjema for XML som avbilder BibTeXs dataformat direkte til XML, og å samle programmer for oversettelse mellom dette skjemaet og andre relevante formater.

Det opereres med to varianter, en for *loose documents* og en for *strict documents*. Den førstnevnte er en direkte representasjonen av BibTeX som XML, det vil si at hvert felt i en BibTeX-fil tilsvarer et element i XML-filen, mens den andre varianten er noe mer finkornet, idet den for eksempel har egne elementer for hver person i forfatterlisten og egne elementer for for- og etternavn.

Det finnes flere andre prosjekter med lignende målsetninger: BibTeXML[5], *BibTeXML*[6], *bibx.dtd*[7] og to DTD'er presentert i [28] er alle DTD'er med bortimot en-til-en-samsvar mellom elementer og felter i BibTeX-filer. Idet de er knyttet til BibTeX, har disse DTD'ene det til felles at de ikke er istand til å uttrykke mange språklige og typografiske detaljer som ble diskutert i kapittel 2. Det gjøres stort

[4] Se `http://bibtexml.sourceforge.net`. Skrevet av Vidar Bronken Gundersen og Zeger W. Hendrikse.

[5] Se `http://bibtexml.org`. Skrevet av Brenno Lurati og Luca Previtali.

[6] Se `http://www.ps.uni-sb.de/~kuhlmann/bibtexml/`. Skrevet av Marco Kuhlmann.

[7] Se `http://www.dmn.tzi.org/bib/bibx.dtd`. Skrevet av Olaf Bergmann.

sett heller ikke noen forsøk på å utvide repertoaret av referansetyper og felter eller å utbedre mangetydighetene ved for eksempel personnavn.

3.3.2 Nye DTD'er for bibliografier

`bibx.dtd`[8] er en DTD laget for neste utgave av `m-bib`, som er bibliografimodulen i dokumentprosesseringssystemet ConTEXt. Da denne oppgaven ble påbegynt, var `bibx.dtd` under utvikling, og når dette skrives sommeren 2003, ser det ut til at `bibx.dtd` vil utvikle seg til en omfattende DTD som tar utgangspunkt i, men ikke er bundet av, BibTEX og RIS. Hver referanse kan bestå av opptil tre **bibliografiske nivåer**, og disse brukes til å beskrive for eksempel et kapittel i en bok eller et bind i en bokserie. Innen hvert av nivåene gjentas den samme strukturen, slik dette utdraget fra versjon 0.2 av DTD'en viser:

```
<!ELEMENT work        (by, titles+) >

<!ELEMENT publication (by?, titles+, partdesc?, pubinfo?) >

<!ELEMENT set         (by?, titles+, partdesc?, pubinfo?) >
```

Alle felter som finnes i RIS og BibTEX, finnes i en eller annen form innenfor disse bibliografiske nivåene, i tillegg til ekstrafelter for notater, hyperlenker, nøkkelord, komponentvis oppdeling av personnavn og lignende.

Spesielt interessant er det at alle elementer med språkavhengig innhold bærer språkattributter. Dette eksempelet, som er redigert noe for lesbarhet, viser deler av modellen for forfatternavn fra versjon 0.2 av `bibx.dtd`:

```
<!ENTITY % lang
  "xml:lang  NMTOKEN                                  #IMPLIED
   translit  (yes|no)                                 'no'
   text-type (primary|translation)                    #IMPLIED">

<!ELEMENT person
  (note*,
   (((initials|(givenname+,middle?)),
     prelast?,familyname,lineage?)|name),
   titulae?, address?)+>

<!ATTLIST person
  %lang;
  role       (author|editor|serieseditor|publisher|
              translator|publisher)                  #REQUIRED
  id         CDATA                                   #IMPLIED>
```

[8] Se `http://tex.aanhet.net/bibx/`. Skrevet av Taco Hoekwater og Markus Hoenicka.

Denne modellen for navn retter langt på vei opp mange av problemene med de eldre formatene, idet det her er mulig å avgjøre hvordan et navn korrekt skal formateres i ulike språklige sammenhenger. Dessverre er dette utilstrekkelig for avanserte flerspråklige bibliografier, da `person`-elementet i disse må kunne ha navn på opprinnelig form, translitterert form og transkribert form samtidig. Det samme gjelder andre elementer i denne DTD'en.

3.3.3 DTD'er fra dokumentprosjekter

To større DTD'er for representasjon av hele dokumenter, DocBook og DTD'en til Text encoding initiative (TEI), kan begge beskrive siteringer og bibliografier. Til disse DTD'ene hører det XSLT-stilark for oversettelse av dokumenter til presentasjonsformater, og en begrenset prosessering av bibliografier gjøres vanligvis som del av denne oversettelsesprosessen.

DocBook

DocBook ble utviklet spesielt for programvaredokumentasjon, brukerveiledninger og referansehåndbøker, og var opprinnelig en DTD for SGML, senere tilpasset XML [28]. I denne oppgaven vil jeg kun omtale XML-inkarnasjonen av DocBook, da denne er iferd med å overta for SGML-utgaven, men det som sies om DocBook XML, gjelder oftest også for DocBook SGML.

For tiden inneholder standarddistribusjonen av DocBook XSLT-stilark for konvertering av DocBook XML til HTML, XHTML og FO, og fra FO er det mulig å produsere blant annet PDF. Støtte for MathML og SVG er mulig ved hjelp av tilleggsmoduler.

DocBook opererer med to former for referanselister. Den ene formen, ***«raw» entries***, inneholder referansedata oppdelt på database-form, det vil si uten noen form for presentasjonsinformasjon:

```
<biblioentry>
  <abbrev>fairbairns95</abbrev>
  <biblioset relation='article'>
    <title>Omega-why Bother with Unicode?</title>
    <author>
      <surname>Fairbairns</surname>
      <firstname>Robin</firstname>
    </author>
    <pagenums>325-328</pagenums>
  </biblioset>
  <biblioset relation='journal'>
    <title>TUGboat</title>
    <volumenum>16</volumenum>
    <issuenum>3</issuenum>
```

```
    <pubdate>september, 1995</pubdate>
    <issn>0896-3297</issn>
  </biblioset>
</biblioentry>
```

Den andre formen, ***«cooked» entries***, tillater endel presentasjonsinformasjon, slik som tegnsetning og fritekst, mellom feltene i referansene, og rekkefølgen av elementene har betydning for formateringen. Det er likevel ingen fullstendig formatering, idet for eksempel skrifttyper og plassering av tekst er overlatt til senere prosessering. Det forrige eksempelet kan se slik ut som en *«cooked» entry*:

```
<bibliomixed>
  <bibliomset relation='article'>
    <surname>Fairbairns</surname>, <firstname>Robin</firstname>.
    <title role='article'>Omega-why Bother with Unicode?</title>.
  </bibliomset>
  <bibliomset relation='journal'>
    <title>TUGboat</title>
    <volumenum>16</volumenum><issuenum>3</issuenum>.
    <pubdate>september, 1995</pubdate></bibliomset>.
</bibliomixed>
```

Vi ser at DocBook benytter flere bibliografiske nivåer, slik som `bibx.dtd` – de to `bibliomset`-elementene i eksemplene ovenfor tilsvarer i dette tilfellet `work` og `publication` i `bibx.dtd`.

Elementet `citation` brukes for å sette inn siteringer i et DocBook-dokument. Innholdet av et slikt `citation`-element skal være en streng som identifiserer en referanse, men denne identifikasjonsstrengen har ingen fast bestemt form. Normalt benyttes innholdet av `abbrev`-elementet i den referansen som siteres [78], og dette forutsettes også av standardstilarkene. De fleste DocBook-elementer, inkludert elementet `citation`, har standardattributtet `role`. Dette er ment for underklassifikasjon [78], og gjør det enkelt å skille de forekomster av `citation`-elementet som skal behandles av en ekstern bibliografiprosessor. Programmet *RefDB* (se avsnitt 3.4.4) benytter denne mekanismen til å markere de `citation`-elementene som *RefDB* skal formatere.

TEI

TEI er en rik og uttrykksfull DTD for humanistiske fag innrettet mot oppmerking av litterære og lingvistiske tekster [72], og inneholder en modell for bibliografier. Akkurat som DocBooks *«raw» entries* og *«cooked» entries* har TEI datasentriske og dokumentsentriske bibliografiformer, men heller ikke i TEI kan bibliografiene inneholde fullstendig presentasjonsinformasjon:

```
<listBibl>
  <head>Bibliography</head>
  <biblStruct id="fairbairns95">
    <analytic>
      <author>Fairbairns, Robin</author>
      <title>Omega&mdash;why Bother with Unicode?</title>
    </analytic>
    <monogr>
      <title>TUGboat</title>
      <imprint>
        <date>1995</date>
      </imprint>
      <biblScope>pp 325&ndash;328</biblScope>
    </monogr>
  </biblStruct>
</listBibl>
```

TEIs bibliografiformat har en svært detaljert dokumentasjon, og den kan opptre med ulik grad av oppmerking, fra ingen oppmerking til den datasentriske oppmerkingen i eksempelet ovenfor.

Gruppen bak TEI har tatt hensyn til eksisterende bibliografisk programvare, blant annet BibTeX og ProCite, og distinksjonene som opptrer i disse dataformatene, kan gjenskapes i TEI [72: 6.10, 6.10.4].

3.3.4 Katalogiserings- og metadata-DTD'er

Flere initiativer forsøker å videreutvikle, standardisere eller forenkle MARC. Et slikt initiativ er BiblioML[9], som er laget for å muliggjøre utveksling av informasjon i UNIMARC, den internasjonale utgaven av MARC, mellom applikasjoner. Et tilsvarende initiativ, Metadata Object Description Schema (MODS),[10] lager et skjema som skal representere en delmengde av MARC 21-formatet. Begge disse prosjektene er først og fremst myntet på bibliotekskataloger, og deres bibliografiske modeller er, i likhet med MARC, svært kompliserte i forhold til en bibliografiprosessors behov.

Dublin Core[11] er en metadatastandard laget for å beskrive informasjonsressurser som er tilgjengelige over et nettverk. Ressursbeskrivelsene settes sammense av en lite antall svært enkle elementer [36]. Dublin Core kan med disse elementene beskrive for eksempel bøker og artikler med titler, ansvarlige, publikasjonsdatoer, temaer og unike identifikatorer, men fraværet av detaljer i oppmerkingen gjør Dublin Core uegnet til bruk i en bibliografiprosessor.

[9] Se `http://www.culture.fr/BiblioML/en/index.html`. Ledet av Martin Sévigny med støtte fra Ministère de la culture et de la communication i Frankrike.

[10] Se `http://www.loc.gov/standards/mods/`. Laget av The Library of Congress' Network Development og MARC Standards Office.

[11] Se `http://dublincore.org/`.

3.4 Programvare

Programvaren som behandler referansedatabaser og siteringslister fra dokumenter, kan inndeles i tre kategorier:

- Programmer for referanseredigering. Disse programmene tilbyr brukergrensesnitt for redigering av referanser og søking i referansedatabaser. Disse er oftest basert på databaseprogramvare for lagring av referansene og benytter derfor interne formater tilpasset som databaseskjemaer. Import og eksport kan oftest gjøres i mange utvekslingsformater, og databaseskjemaet er laget på grunnlag av fellestrekk ved de støttede utvekslingsformatene.

- Bibliografiprosessorer. Disse leser siteringer fra en egnet kilde og referanser fra en referansedatabase og oversetter dem til et presentasjonsformat.

- Kombinasjonsprogrammer. Disse programmene kombinerer funksjonaliteten til referanseredigeringsprogrammene og bibliografiprosessorene ved å benytte den innebygde databasen som kilde når presentasjonsformen av siteringer og referanselister skal genereres.

Denne kategoriseringen er ikke absolutt; for eksempel kan et program for referanseredigering tilby et eksportformat som både kan kalles et utvekslingsformat og et presentasjonsformat, for eksempel DocBook-bibliografier med *«cooked» entries*.

Den første gruppen, referanseredigeringsprogrammer, er ikke av særlig interesse i denne oppgaven, og jeg vil derfor bare kort nevne to slike prosjekter som ser ut til å være populære: *JReferences*[12] er et program basert på MySQL-databaser med eksport- og importmuligheter for BibTeXML, BibTeX, DocBook og RIS. *BibDB*[13] er et tilsvarende program for Windows-plattformen; det leser og skriver de eldre databaseformatene BibTeX, *Tib* og *refer*.

3.4.1 Programmet BibTeX

BibTeX ble laget først og fremst for å brukes sammen med LaTeX, og det benytter seg av bibliografimakroer som er innebygget i LaTeX. Det er også mulig å bruke BibTeX med plain TeX ved hjelp av makrofilen `btxmac.tex` [64].

Et LaTeX-dokument for en BibTeX-bibliografi har følgende struktur:

```
...
\cite{aho88}
...
```

[12] Se `http://jreferences.sourceforge.net/`. Skrevet av E. L. Willighagen.

[13] Se `http://www.mackichan.com/index.html?bibdb/default.htm`. Skrevet av Eyal Doron.

```
\bibliography{mybibliography}
\bibliographystyle{plain}
...
```

Makroene i eksempelet angir henholdsvis en sitering med en referanseidentifikator, en referansedatabase som skal leses og bibliografistilen som skal brukes. En typisk BibTeX-sesjon med LaTeX forløper i fire trinn.

1. LaTeX leser dokumentet, og makroene for bibliografier skriver alle siteringene og endel annen BibTeX-informasjon, til en `aux`-fil.

2. BibTeX leser `aux`-filen, bibliografidatabasene og stilprogrammene. Av dette lages en loggfil og en referanseliste i en fil med ekstensjonen `bbl`.

3. LaTeX leser `bbl`-filen, og referanselisten fra denne inkorporeres i dokumentet. For hver referanse i referanselisten skrives referanseidentifikatorene til `aux`-filen av bibliografimakroene.

4. LaTeX leser referanseidentifikatorene fra `aux`-filen og binder disse til referansene i referanselisten.

Dette er en svært omstendelig prosess som burde ha vært implementert på en enklere måte. Man kunne man ha oppnådd dette ved å overlate genereringen av siteringer til BibTeX; da ville det siste trinnet ha vært overflødig, og det tredje trinnet forenklet.

En liste med alle nøklene i et TeX/LaTeX-dokument og en eller flere BibTeX-databaser danner altså sammen inndata for programmet. Selve formateringen gjøres av stilprogrammer skrevet i et navnløst stakk-språk med postfiks-notasjon [62]. Dette språket er ikke spesielt komplisert, men vil virke uvant på mange, nettopp på grunn av postfiks-notasjonen (se figur 3.2 for et eksempel). BibTeX tilbyr rundt førti innebygde funksjoner og variabler i dette språket, og disse muliggjør lesing av innholdet av referansedatabaser, strengmanipulasjon og oppdeling av navn og titler i logiske komponenter.

En liten samling standardstiler følger med BibTeX – de fleste av disse tilpasset naturvitenskapelige publikasjoner. I tillegg finnes det store mengder mer eller mindre velskrevne og vedlikeholdte stiler distribuert av organisasjoner og privatpersoner.

BibTeXs stilspråk er tungvint å bruke fordi det ikke finnes noen metode for å inkludere deler av andre stiler eller funksjonsbiblioteker. Da de fleste stiler må utføre mange av de samme operasjonene, replikerer de fleste stiler et stort antall funksjoner der kun detaljer skiller stilene fra hverandre. For eksepel er mange stiler oversettelser av standardstilene til andre språk med noen lokale tilpasninger, og disse må lages ved å kopiere stilfilen og modifisere kopien.

Grensen mellom referansedatabasene og stilprogrammene er tidvis også noe uklar. Månedsnavnene er definert i standardstilene ved hjelp av strengkonstanter, og disse

```
FUNCTION {format.editors}
{ editor empty$
    { "" }
    { editor format.names
      editor num.names$ #1 >
        { ", editors" * }
        { ", editor" * }
      if$
    }
  if$
}
```

Figur 3.2: Eksempel på stilspråket i BibTeX. Eksempelet, som er tatt fra standardstilen `plain.bst` i BibTeX 0.99b, viser en funksjon som formaterer listen over redaktører i en referanse. Funksjonen leses slik: Hvis feltet `editor` er tomt, returner en tom streng. Hvis ikke, utfør funksjonen `format.names` med feltet `editor` som argument, og konkatener resultatet enten med strengen `, editors` når feltet `editor` inneholder flere enn ett navn eller med strengen `, editor` når feltet inneholder kun ett navn.

brukes av de fleste referansedatabaser til koding av månedsnavn. Det er likevel ingen garanti for at disse konstantene finnes dersom man ikke benytter standardstilene, og i slike tilfeller er referansedatabasen ubrukelig.

Da BibTeX lar stilprogrammene behandle feltene i referansedatabasene, kan det fritt innføres nye felter og nye referansetyper, og hver stil kan behandle disse eller ignorere dem. De uoffisielle tilleggsfeltene som ble omtalt i tabell B.2, er mulige av denne grunn.

Som nevnt er det ingen bestemt tegnkoding for BibTeX-filer, og programmet BibTeX er da heller ikke istand til å håndtere 8-bits-tegnsett korrekt. Forfatteren av BibTeX, Oren Patashnik, hevder at versjon 1.0 skal løse dette problemet [63], men denne versjonen lar vente på seg, og i mellomtiden ble BibTeX-varianten BibTeX8[14] laget for å støtte 8-bits-tegnsett og tilpassede sorteringsregler.

Avansert språkstøtte mangler både i BibTeX og i BibTeX8, og brukere av disse programmene omgår gjerne dette i enkelttilfeller ved å bruke «lure» TeX-makroer i feltene, slik som dette eksempelet fra [27] viser:

```
@Preamble{"\newcommand\oneletter[1]{#1}"}
author = {{\oneletter{Yu}}ri Govorukhin}
```

Hensikten med denne konstruksjonen er å få BibTeX til å behandle «Yu.» som initialformen av navnet «Yuri» istedenfor «Y.».

[14] Tilgjengelig fra CTAN under `biblio/bibtex/8-bit/`. Skrevet av Niel Kempson og Alejandro Aguilar-Sierra.

3.4.2 Hjelpeprogrammer og makropakker for BibTeX

Uklarhetene rundt BibTeX-formatet har ført til et behov for hjelpeprogrammer som *bibclean*,[15] som blant annet forsøker å avgjøre om feltene inneholder korrekte data, slik som fornuftige sammensetninger av navn og gyldige ISBN og ISSN.

Blant makropakker som modifiserer integrasjonen mellom BibTeX og LaTeX, er pakken `natbib`[16] antagelig den viktigste. BibTeX støtter kun nøkkel-siteringer, men `natbib` reimplementerer siteringsmekanismen slik at den fungerer med både nøkkel- og forfatter-dato-stiler [14]. Pakken benytter et eget stilprogram som kan motta parametre fra makroer i et LaTeX-dokument. For eksempel kan en forfatter-dato-stil velges i et dokument på denne måten:

```
...
\usepackage{natbib}
\bibpunct[: ]{(}{)}{;}{a}{}{:}
...
\cite{aho88}
...
\bibliography{mybibliography}
\bibliographystyle{natbib}
...
```

Pakken `custom-bib`[17] forsøker på sin side å samle vanlige stilvarianter ved å lage brukertilpassede stilprogrammer, blant annet med oversettelser til flere språk, ved hjelp av en samling med «super-stilprogrammer». [13].

For korrekt orddeling i bibliografier med referanser på flere språk kan pakken `babelbib`[18] brukes. Den bruker et tilleggsfelt, `language`, i BibTeX-databasene for angivelse av referansens språk, og ved hjelp av modifiserte utgaver av standardstilene, vil korrekte orddelingsmønstre velges i alle referanser [35].

Videre finnes det pakker for å ha flere bibliografier i ett dokument, for å ha siteringer i fotnoter eller samlet som sluttnoter, for å slå sammen flere delangivelser innen en sitering og for å lette arbeidet med å plassere siteringer på riktige steder i teksten.

3.4.3 Andre bibliografiprosessorer for TeX

I tillegg til BibTeX og *Tib* finnes det flere prosjekter som forsøker å lage mer eller mindre selvstendige bibliografiprosessorer for TeX.

[15] Se `http://www.math.utah.edu/pub/bibclean`. Skrevet av Nelson H. F. Beebe.

[16] Tilgjengelig fra CTAN under `macros/latex/contrib/supported/natbib/`. Skrevet av Patrick W. Daly.

[17] Tilgjengelig fra CTAN under `biblio/bibtex/contrib/custom-bib/`. Skrevet av Patrick W. Daly.

[18] Tilgjengelig fra CTAN under `biblio/bibtex/contrib/babelbib/`. Skrevet av Harald Harders.

CAMEL[19] er en prototype for bibliografistøtten i LaTeX3. Den viktigste forskjellen mellom CAMEL og bibliografistøtten i LaTeX 2_ε er en styrket siteringskommando som fungerer med andre stiler enn nøkkelstiler. CAMEL er først og fremst tenkt brukt sammen med BibTeX, men en interessant utvidelse er at CAMEL også åpner for bruk av noen av de kommeriselle produktene (se avsnitt 3.4.5) [5].

`m-bib`,[20] som er bibliografimodulen for TeX-varianten ConTeXt, bruker BibTeX kun til sortering av referanser, mens både referansedata og stilinformasjon skrives som kommandoer i dokumentet [37]. Det er meningen at `m-bib` i en senere versjon skal gå over til referansedatabaser i XML som følger `bibx.dtd`.

En nokså lik pakke, men mye mer avansert, er LaTeX-pakken `amsrefs`[21] laget for $\mathcal{AMS}$-publikasjoner.[22] $\mathcal{AMS}$-pakken benytter et feltformat som ligner BibTeX-formatet, men med noen mindre og meget fornuftige endringer. Feltene skrives i LaTeX-dokumenter, og hele formateringen utføres av LaTeX-makroer. `amsrefs` støtter nøkkel-systemer og forfatter-dato-systemer, kan omforme titler til tittelform, kan substituere tidsskriftnavn og utgivere med forkortede former, har innebygget støtte som tilsvarer mange av de andre tilleggspakkene for BibTeX og bruker egne felter for angivelse av referansers språk og for orddelingsmønstre [19].

mlBibTeX[23] er en reimplementasjon av BibTeX som utvider BibTeX-filformatet slik at det er mulig å veksle mellom forskjellige språk fra referanse til referanse og å ha alternative verdier for samme felt på ulike språk [39]. Forfatteren av *mlBibTeX* planlegger å gå over til Unicode for intern tegnkoding og å legge til støtte for Texinfo- og DocBook-dokumenter [43].

clBibTeX[24] er enda en reimplementasjon, hvis hovedmål er å erstatte BibTeXs tungvinte stilspråk med Common Lisp. En interessant side ved dette prosjektet er forsøket på automatisk oversettelse av BibTeX-stilprogrammer til Common Lisp.

3.4.4 Andre ikke-kommersielle prosjekter

RefDB[25] er et kombinasjonsprogram som bruker relasjonsdatabasen *MySQL* med et internformat basert på RIS. Referansene kan importeres fra og eksporteres til en lang rekke formater, blant annet DocBook, RIS og BibTeX.

Fra referansedatabasen og siteringer i dokumenter kan *RefDB* lage formaterte siteringer og referanselister for DocBook, TEI og LaTeX [38]. Stilene er spesifisert i

[19] Tilgjengelig fra CTAN under `macros/latex/contrib/camel/`. Skrevet av Frank G. Bennett, Jr.

[20] Tilgjengelig fra `http://www.elvenkind.com/~taco/bibmod/`. Skrevet av Taco Hoekwater.

[21] Tilgjengelig fra `http://www.ams.org/tex/amsrefs.html`. Skrevet av Michael Downes.

[22] $\mathcal{AMS}$ står for American Mathematical Society.

[23] Skrevet av Jean-Michel Hufflen.

[24] Tilgjengelig fra `http://www.nongnu.org/cl-bibtex/`. Skrevet av Matthias Koeppe.

[25] Tilgjengelig fra `http://refdb.sourceforge.net`. Skrevet av Markus Hoenicka.

XML og definerer formateringen for de ulike referansetypene (i dette tilfellet er GEN en felleskategori for alle referansetyper) og hvordan siteringene i teksten skal se ut:

```
<?xml version="1.0"?>
<!DOCTYPE CITESTYLE PUBLIC "-//Markus Hoenicka//DTD CiteStyle XML//EN"
                       "http://refdb.sourceforge.net/dtd/citestylex.dtd">
<CITESTYLE>
  <STYLENAME>bibtex-full</STYLENAME>
  <!-- $Id: bibtex-full.xml,v 1.2 2002/12/13 20:35:44 mhoenicka Exp $ -->
  <REFSTYLE>
    <PUBTYPE TYPE="GEN">
      <JOURNALNAME CASE="ASIS" DEFAULTTEXT="FULL" ALTERNATETEXT="AABBREV"
        PUNCTUATION="SPACE">
      </JOURNALNAME>
    </PUBTYPE>
  </REFSTYLE>
  <CITSTYLE INTEXTSEQUENCE="ASIS" BIBLIOSEQUENCE="BASIS">
    <PRECEEDING>(</PRECEEDING>
    <FOLLOWING>)</FOLLOWING>
    <CITSEPARATOR>; </CITSEPARATOR>
    <RANGESEPARATOR>-</RANGESEPARATOR>
    <BIBLIOTITLE>Reference List</BIBLIOTITLE>
    <INTEXTDEF>
      <REFNUMBER></REFNUMBER>
    </INTEXTDEF>
  </CITSTYLE>
</CITESTYLE>
```

pybliographer[26] ligner mye på *RefDB*, men benytter ingen relasjonsdatabase for referansene og leser heller utvekslingsformater direkte. *Bibulus*[27] er på sin side en ren bibliografiprosessor med flerspråklighet som hovedmål. Prosjektet benytter et eget XML-format og skal kunne produsere mange ulike presentasjonsformater, men ettersom prosjektet i skrivende stund er svært uferdig, er det uklart hvordan dette vil se ut.

3.4.5 De kommersielle produktene

ProCite,[28] *EndNote*,[29] *Reference Manager*,[30] *Papyrus*[31] og *Library Master*[32] omtales av [68] som «de fem store» blant de kommersielle programmene. «De fem store» er

[26] Se `http://canvas.gnome.org:65348/pybliographer`. Skrevet av Frédéric Gobry.
[27] Se `http://savannah.nongnu.org/projects/bibulus`. Skrevet av Thomas Widmann.
[28] Se `http://www.procite.com`. Utgis av ISI ResearchSoft.
[29] Se `http://www.endnote.com`. Utgis av ISI ResearchSoft.
[30] Se `http://www.refman.com`. Utgis av ISI ResearchSoft.
[31] Se `http://www.researchsoftwaredesign.com/`. Utgis av Research Software Design.
[32] Se `http://www.balboa-software.com`. Utgis av Balboa Software.

kombinasjonsprogrammer som bruker proprietære dataformater – gjerne tilknyttet databaseprogramvare. Ved hjelp av databaseprogramvaren tilbyr de avanserte verktøy for vedlikehold og søking, og noen tillater også deling av databasene over nettverk. Bibliografiprosessordelene er laget for bruk med tekstbehandlingsprogrammer for Macintosh eller Microsoft Windows.

Internformatene er stort sett udokumenterte, men på samme måte som i BibTeX-formatet og i RIS-filer deles referansene opp i referansetyper, og et utvalg av databasefeltene er relevante for hver referansetype. Utvalget av typer og felter er rikt; *ProCite* omfatter, ifølge presentasjonen på produktets nettside, 39 referansetyper og 45 felter, hvorav noen er svært spesielle, slik som sender og mottager av en e-post.

En særlig interessant egenskap ved disse produktene er den brede støtten før søking i og import av data fra Internett-databaser. Dette omfatter for eksempel import fra store biblioteker som tilbyr Z39.50-grensesnitt, generelle siteringsdatabaser som *ISI Web of Science*[33] eller mer spesialiserte databaser som *PubMed*[34].

Stilutvalget er også svært rikt – *EndNote* leveres med rundt 700 ferdige stiler – og siteringer gjøres enkelt ved å plassere spesielle tegnsekvenser i dokumentet i tekstbehandleren.

Flerspråklighet har derimot ikke vært et satsningsfelt for produsentene av disse programmene. De er laget først og fremst for det amerikanske markedet, og grunnleggende flerspråklige egenskaper, som ulik tegnkoding i referansedatabaser, mangler.

3.5 Oppsummering

Det er ingen mangel på programvare for bibliografiformatering eller manipulasjon av bibliografier. Mange av produktene er svært mye brukt og fungerer godt for sine formål, det vil si innen en bestemt disiplin, for bestemte typer bibliografistiler, for bestemte dokumentprosesserings- eller typesettingssystemer eller for bestemte språk. For flerspråklig bibliografiprosessering ser det ut til at `bibx.dtd` kommer til å bli viktig, men det mangler fremdeles programvare for dette, og detaljene knyttet til dataformater for flerspråklige bibliografier er fremdeles ikke kartlagt.

[33] Se `http://www.isinet.com/isi/products/citation/wos/`. *ISI Web of Science* omfatter siteringer fra flere tusen tidsskrifter fra svært mange disipliner.

[34] Se for eksempel `http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=PubMed`. *PubMed* er en database laget av National Library of Medicine med siteringer innen medisin og biokjemi.

Kapittel 4

Unicode

Et gjennomgående problem med flerspråklig programvare er representasjon og behandling av tegnsett. Interaksjonen mellom språk og tegnsett kan være svært komplisert, særlig fordi det finnes tallrike såvel offisielle som uoffisielle standarder og konvensjoner i bruk. For en flerspråklig bibliografiprosessor vil vi ikke bare ha behov for å kunne representere data på flere språk og i ulike skriftspråk, men også å sortere strenger etter ulike språklige og nasjonale konvensjoner, å holde orden på versaler i titler og lignende.

ISO/IEC 10646 [48] er en internasjonal standard som definerer **Universal Multiple-Octet Character Set** (UCS), og var det første rimelig vellykkede forsøket på å lage et tegnsett for alle tegn i verdens skriftspråk. UCS omfatter i prinsippet alle andre større tegnsettstandarder og garanterer *round-tripping*.[1] Parallelt med utviklingen av UCS ble The Unicode Consortium dannet av en gruppe produsenter av kommersiell flerspråklig programvare. Formålet med dette var også å lage et felles flerspråklig tegnsett, kalt **Unicode**, men i 1991 forstod de to gruppene at verden kun behøver ett slikt tegnsett. Derfor er de to standardene fra og med Unicode 1.1 og ISO 10646-1:1993 synkronisert slik at de to standardenes tegnsett er identiske [75: 1.4]. I det følgende vil begrepene UCS og Unicode følgelig brukes noe om hverandre, selv om det er forskjeller mellom dem. UCS er først og fremst en tabell over tegn og definisjoner av relevant terminologi, mens Unicode også definerer semantisk informasjon og algoritmer for presentasjon av tekst og for sortering eller sammenligning av strenger.

[1] ***Round-tripping*** innebærer i denne sammenhengen at en konvertering av data fra et tegnsett A til et tegnsett B og tilbake igjen til tegnsett A er tapsfri og en-til-en.

Kodepunkt	Glyfeksempel	Beskrivelse
U+0042	B	LATIN CAPITAL LETTER B
U+0062	b	LATIN SMALL LETTER B
U+0392	Β	GREEK CAPITAL LETTER BETA
U+03B2	β	GREEK SMALL LETTER BETA
U+0411	Б	CYRILLIC CAPITAL LETTER BE
U+0431	б	CYRILLIC SMALL LETTER BE

Figur 4.1: Eksempel på beskrivelser av kodepunkter i Unicode. Beskrivelsene er tatt fra Unicode 3.0.

4.1 Tegn og kodepunkter

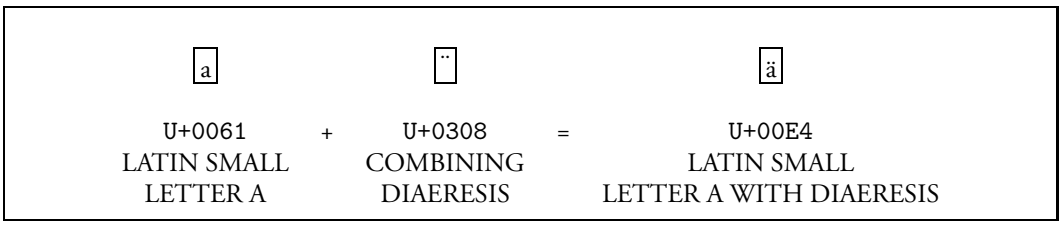
Et **tegn** (engelsk *character*) i UCS og Unicode er en abstrakt entitet som ikke er bundet til noe spesielt lagringsformat. Tegnene er også uten et klart definert utseende. Isteden defineres hvert tegn i standardene ved hjelp av en beskrivelse av tegnet og en tilordning av et unikt nummer. Numrene kalles **kodepunkter** (engelsk *code points*) og oppgis oftest i heksadesimal notasjon med prefikset U+ og utfyllende nuller for 16 eller 32 bits heltall. Det faktiske utseendet til et bestemt tegn er ikke del av standarden, men det presenteres en eksempelglyf for hvert tegn (se figur 4.1).

Formelt definerer UCS et kodepunktrom på 31 bits, men det er lite som tyder på at det noen gang vil bli definert tegn utover 21-bit-underrommet fra U+00000000 til U+0010FFFF, ettersom Unicode offisielt er begrenset til et 21-bit-rom. 16-bit-underrommet av UCS (intervallet U+0000 til U+FFFF) kalles **Basic Multilingual Plane** (BMP) eller **Plane 0** og omfatter de vanligste moderne skriftspråkene. Videre er tegnene i intervallet U+0000 til U+007F identiske med dem som er definert i US-ASCII-standarden (ISO 646 IRV), og tegnene i intervallet U+0000 til U+00FF er identiske med Latin 1-standarden (ISO/IEC 8859-1) [50]. US-ASCII og Latin 1 er antagelig de to mest brukte tegnsettene, og inklusjonen av disse to som delmengder av UCS og Unicode er gjort både av praktiske årsaker og for å sikre at *round-tripping* mellom disse tegnsettene skal fungere.[2]

4.2 Kombinasjonstegn og normalisering

Enkelte UCS-tegn er **kombinasjonstegn** (engelsk *combining characters*). Disse kan sammenlignes med dødtastene på et PC-tastatur: Når man trykker på dødtasten, flyttes ikke markøren før neste tast trykkes, og de to tegnene kombineres. Kombinasjonstegn i UCS er på tilsvarende måte ikke komplette tegn, men tegn, oftest

[2]Her er det først og fremst kontrolltegnene i US-ASCII og ISO/IEC 8859-x, slik som BEL (*bell*), BS (*backspace*) og lignende, som ville ha skapt problemer for *round-tripping*.



Figur 4.2: Eksempel på kombinasjonstegn og prekomponerte tegn.

diakritika,[3] som kombineres med *forrige* tegn. Det er mulig med flere kombinasjonstegn i sekvens. Enkelte språk, for eksempel koreansk, arabisk og thailandsk, kan kun støttes fullt ut ved hjelp av kombinasjonstegn.

De fleste diakritika finnes som kombinasjonstegn, men de vanligste aksentuerte tegnene har også egne kodepunkter og kalles **prekomponerte tegn** (engelsk *precomposed characters*). Slik oppstår ekvivalenser mellom tegnsekvenser og enkelttegn (se figur 4.2). Det samme skjer ved sekvenser av flere kombinasjonstegn, der alle permutasjoner av kombinasjonstegene kan være ekvivalente.[4] For det greske tegnet ΐ finnes det seks mulige kombinasjoner [31]!

Normalisering betegner den prosessen hvor en sekvens av tegn overføres til en sekvens som så langt mulig bare inneholder prekomponerte tegn eller en sekvens som bare inneholder grunntegn og kombinasjonstegn. Disse formene kalles henholdsvis normaliseringsform C eller KC og normaliseringsform D eller KD [16]. De ulike normaliseringsformene brukes i sammenhenger som krever en entydig representasjon av Unicode-strenger. I praksis er normaliseringsform C den viktigste, for eksempel brukes den i Character Model for the World Wide Web [20].

4.3 Koding og serialisering

Koding av et tegnsett betegner en avbildning mellom et tegnsett og sekvenser av 8-bit-, 16-bit- eller 32-bit-entiteter. For eldre tegnsett er det ofte et en-til-en-forhold mellom kodepunkter og entitetene i en slik sekvens. Kodepunktet `0x20` i US-ASCII kodes for eksempel som en byte med innholdet `0x20`. Dette fungerer ikke for UCS og Unicode, og det finnes i alt fire ulike kodinger som er vanlige å bruke:

[3] **Diakritika** eller **diakritiske tegn** er hjelpetegn som vanligvis plasseres over eller under et annet tegn og brukes for å markere varianter i uttale, bortfall av tegn, betydningsforskjeller etc. De vanligste i moderne vest-europeiske språk er ulike aksententer og prikker over bokstavene. Også i ikke-vestlige alfabeter brukes diakritika, for eksempel i japansk der tegnet *dakuten* (濁点 dakuten), som ser ut som to korte streker (゛), plasseres over *kana*-tegn (仮名 kana) og modifiserer uttalen [58].

[4] Ikke-ekvivalente permutasjoner oppstår for eksempel når flere aksenter skal plasseres over et grunntegn. Da blir den vertikale plasseringen av aksentene avhengig av kombinasjonstegnenes rekkefølge: a + ¨ + ` ≠ a + ` + ¨ [75: 5.17].

Navn	Entiteter per tegn	Kodbart Unicode-intervall	Kodbart UCS-intervall
UCS-4	1 à 32 bit		U+00000000–U+7FFFFFFF
UCS-2	1 à 16 bit		U+00000000–U+0000FFFF
UTF-8	1 til 6 à 8 bit	U+000000–U+10FFFF	U+00000000–U+7FFFFFFF
UTF-16	1 eller 2 à 16 bit	U+000000–U+10FFFF	U+00000000–U+0010FFFF
UTF-32	1 à 32 bit	U+000000–U+10FFFF	U+00000000–U+7FFFFFFF

Tabell 4.1: Et utvalg kodinger av UCS og Unicode.

UCS-4 Hvert tegn kodes som en 32-bit-entitet.

UCS-2 Hvert tegn kodes som en 16-bit-entitet. Tegnsettet begrenses da til et 16-bit-rom, og kodingen er derfor ikke i stand til å kode hele UCS eller Unicode.

UTF-16 Hvert tegn kodes som en eller to 16-bit-entiteter. De første 2^{16} tegnene representeres med én 16-bit-entitet, resten med to 16-bit-entiteter etter hverandre. Dette er mulig ettersom kodepunktintervallet U+D800–U+DFFF er avsatt som markører for par av 16-bit-entiteter.

UTF-8 Hvert tegn kodes som mellom en og seks 8-bit-entiteter.

Merk at UTF-8 og UTF-16 defineres både av UCS og Unicode, men for Unicode er UTF begrenset til et 21-bit-koderom. Unicode definerer også UTF-32, som tilsvarer UCS-4, men kun Unicode-koderommet tillates brukt (se tabell 4.1).

En komplikasjon oppstår når kodede tegn skal **serialiseres** til en sekvens av bytes for lagring eller kommunikasjon. Serialiseringen av kodinger med 16- eller 32-bit-entiteter vil nemlig bli avhengig av maskinarkitekturens *endianness*. Kodingsvariantene betegnes med suffiksene «LE» for *little-endian* (minst signifikante byte først) og «BE» for *big-endian* (mest signifikante byte først), for eksempel UTF-16LE og UTF-16BE.

Et ***byte-order mark*** (BOM) er en spesiell sekvens som skal brukes for å markere *endianness* i en serialisering. Serialiseringer uten BOM skal ifølge standardene være *big-endian*, men denne regelen overholdes ikke alltid.

De mange kodingsvariantene skyldes både den historiske utviklingen av UCS og Unicode og ulike krav til for eksempel plassforbruk. UTF-32 er særlig mye brukt som internformat ved strengmanipulasjon, ettersom hvert tegn tilsvarer en 32-bit-entitet og får plass i et prosessorregister, men den er mindre egnet for fillagring og for nettverkstrafikk. UTF-8 er vanlig i nyere Internett-protokoller som HTTP, LDAP og SNMP, og på mange Unix-lignende systemer, for eksempel Linux, Solaris og FreeBSD, mens UTF-16 brukes i Java og av Microsoft Windows. Heldigvis er oversettelse mellom formatene algoritmisk og effektiv, i motsetning til konversjon mellom tradisjonelle kodinger, der det ofte er nødvendig med tabelloppslag eller ineffektive operasjoner som multiplikasjoner.

x **i intervallet**	b	s
U+00000000–U+0000007f	0x00	0
U+00000080–U+000007ff	0xc0	6
U+00000800–U+0000ffff	0xe0	12
U+00010000–U+001fffff	0xf0	18
U+00200000–U+03ffffff	0xf8	24
U+04000000–U+7fffffff	0xfc	30

Tabell 4.2: Ledende bitverdier og bitskifting i UTF-8-koding. Tabellen viser den ledende bitverdien b og bitskiftingen s for et kodepunkt x.

4.3.1 UTF-8

UCS transformation format 8*/*Unicode transformation format 8 (UTF-8) er definert i ISO 10646-1 Annex R [48], i RFC 2279 [79] og i Unicode-standarden [75]. Det er en multibytekoding, det vil si at den består av en sekvens av en eller flere byte per tegn, og UTF-8-strenger kan derfor manipuleres med en C-*array* av typen `char`. Kodingen er utformet slik at hele UCS-koderommet på 31 bits kan kodes, og ett UCS-tegn kodet i UTF-8 kan bli opptil 6 bytes langt. Innenfor 21-bit-rommet definert av Unicode vil sekvensene dog aldri bli lengre enn 4 bytes.

Kodingen av et kodepunkt x består av en ledebyte med verdi l, eventuelt etterfulgt av en til fem fortsettelsesbytes p_i. Ledebytens verdi er gitt ved

$$l = b + (x >> s)$$

der $(x >> s)$ betyr «x skiftet s bits mot høyre», og b og s har verdiene angitt i tabell 4.2. Det legges til fortsettelsesbytes p_i gitt ved

$$p_i = \texttt{0x80} + (x >> s_i) \bmod 2^6 \quad 1 \leq i \leq 5$$

der $s_i = s_{i-1} - 6$ og $s_0 = s$, så lenge $s_i \geq 0$. For eksempel kodes kodepunktet $x = \texttt{U+0000263A}$ som

$$\begin{aligned}(l, p_1, p_2) &= (\texttt{0xe0} + (x >> 12), \texttt{0x80} + (x >> 6) \bmod 2^6, \texttt{0x80} + x \bmod 2^6) \\ &= (\texttt{0xe0} + \texttt{0x02}, \texttt{0x80} + \texttt{0x18}, \texttt{0x80} + \texttt{0x3a}) \\ &= (\texttt{0xe2}, \texttt{0x98}, \texttt{0xba})\end{aligned}$$

Som vi ser består kodingen av x kun av skift-, maskerings- og addisjonsoperasjoner og kan implementeres med enkle og effektive prosessorinstruksjoner.

I tabell 4.3 ser vi at tegn fra US-ASCII-tegnsettet (det vil si i intervallet U+0000 til U+007F) ganske enkelt kodes som tilsvarende bytes i intervallet 0x00 til 0x7f. Vi ser også at disse bytene ikke forekommer i kodingen av tegn over U+007F, da kodingen av disse alltid består av sekvenser på flere bytes med den mest signifikante biten satt. En streng i US-ASCII er altså en gyldig streng i UTF-8, og ethvert byte under 0x80

Kodepunktintervall	Koding
U+00000000–U+0000007F	(0xxxxxxx)
U+00000080–U+000007FF	(110xxxxx, 10xxxxxx)
U+00000800–U+0000FFFF	(1110xxxx, 10xxxxxx, 10xxxxxx)
U+00010000–U+001FFFFF	(11110xxx, 10xxxxxx, 10xxxxxx, 10xxxxxx)
U+00200000–U+03FFFFFF	(111110xx, 10xxxxxx, 10xxxxxx, 10xxxxxx, 10xxxxxx)
U+04000000–U+7FFFFFFF	(1111110x, 10xxxxxx, 10xxxxxx, 10xxxxxx, 10xxxxxx, 10xxxxxx)

Tabell 4.3: Bitrepresentasjon av UTF-8-sekvenser. Konkateneringen av hver x i bitkodingen tilsvarer bitrepresentasjonen til et kodepunkt.

Bitrepresentasjon	Mulige verdier	Bruk
(0xxxxxxx)	0x00–0x7f	1-bytesekvens/US-ASCII-tegn.
(10xxxxxx)	0x80–0xbf	Fortsettelse på sekvens.
(110xxxxx)	0xc0–0xdf	Begynnelsen på 2-bytesekvens.
(1110xxxx)	0xe0–0xef	Begynnelsen på 3-bytesekvens.
(11110xxx)	0xf0–0xf7	Begynnelsen på 4-bytesekvens.
(111110xx)	0xf8–0xfb	Begynnelsen på 5-bytesekvens.
(1111110x)	0xfc–0xfd	Begynnelsen på 6-bytesekvens.
(1111111x)	0xfe–0xff	Reservert for BOM.

Tabell 4.4: Betydningen av ulike bytes i en UTF-8-sekvens. Det er alltid mulig å avgjøre om en bestemt byte er begynnelsen på en sekvens, og hvor lang den i så fall er, eller om den er fortsettelsen av en sekvens.

er et kodepunkt i US-ASCII-tegnsettet. Nullterminering av strenger vil også fungere som for tradisjonelle kodinger, da byten 0x00 kun forekommer i kodingen av kodepunkt U+0000.

Videre ligger den første byten i en multibytesekvens alltid i intervallet 0xc0 til 0xfd, mens fortsettelsen på sekvensen er på bitformen 10xxxxxx (se tabell 4.4). Dette gjør UTF-8 til en **resynkroniserbar** eller **restartbar** koding, idet det er mulig å finne begynnelsen på en multibytesekvens. Den første byten i en multibytesekvens vil også indikere lengden på sekvensen, noe som ofte er nødvendig å vite i strengmanipulasjonsfunksjoner.

Ved sortering av UTF-8-sekvenser bevares den leksikografiske rekkefølgen. Det vil si at en binær sortering av UTF-8-kodede strenger gir den samme rekkefølgen som en binær sortering av de samme strengene i UCS-4-koding. En sortering av strenger etter UCS-kodepunkter kan altså gjøres byte for byte ved hjelp av primitive strengsammenligningsfunksjoner som `strcmp` i språket C.

4.4 *Locales* og den flerspråklige modellen

Det er to modeller i bruk som tar hensyn til slike flerspråklige variabler: *locale*-modellen og den flerspråklige modellen. I *locale*-modellen må brukeren selv veksle mellom ulike *locales*, som er samlinger av kultur- og regionavhengige attributter, blant annet valg av tegnsett. Vekslingen er oftest prosessvis, og det gjør bruk av flere språk på en gang komplisert. I den flerspråklige modellen benyttes et tegnsett som inneholder alle nødvendige tegn, og det veksles ikke mellom *locales*. Isteden spesifiseres kultur- og regionavhengig informasjon for hver enkelt operasjon.

Moderne C-biblioteker som implementerer *ISO C90 Amendment 1*, benytter *locale*-modellen, mens nyere Unicode-orienterte biblioteker, som *glib*,[5] benytter den flerspråklige modellen.

4.5 Sortering

Språkavhengig sortering av strenger er komplisert fordi tegn eller grupper av tegn ordnes ulikt i ulike språk, og det finnes ofte flere varianter i bruk parallelt innen ett språk- eller kulturområde. Sortering er altså en funksjon ikke bare av strengene som skal sorteres, men også av variabler som hvorvidt sorteringen skal gjøres avhengig eller uavhengig av versaler, om den skal gjøres med eller uten hensyn til diakritika eller om den skal være fonetisk eller etter radikaler og streker.

Den tradisjonelle løsningen på språkavhengig sortering er å oversette hver enkelt streng til en sekvens av **vekter** ved hjelp av en vektingstabell for det aktuelle språket. Vektingstabellen er bygget opp slik at alle strenger sorteres i den ønskede rekkefølge når vektsekvensene brukes som sorteringsnøkler for en passende sorteringsfunksjon. Vektene er gjerne heltall, og sorteringsfunksjonen er da en sammenligningsfunksjon som sammenligner sekvensene etter tallverdi.

4.5.1 Sortering i ISO C90

I ISO C90 fungerer funksjonen `strxfrm` som oversettelsesfunksjon fra strenger til vektsekvenser, og disse kan sammenlignes ved hjelp av den vanlige strengsammenligningsfunksjonen `strcmp`.[6]

I det første eksempelet i figur 4.3 vises vektene som ble tilordnet tre strenger med norske og tyske ord. Med norsk sortering blir rekkefølgen *unter*, *über*, *yngre*,

[5] *glib* er et bibliotek for C med funksjoner for datastrukturer, strenger, tråder og så videre. Se `http://www.gtk.org/`.

[6] Intensjonen er at `strxfrm` skal brukes forut for en operasjon som fører til mange sammenligningsoperasjoner slik at oversettelsen til vekter kun gjøres én gang. Er det kun et fåtall strenger som skal sammenlignes, utfører funksjonen `strcoll` det samme som `strxfrm` og `strcmp`.

	Norsk	Engelsk
absolute	13, 14, . . .	12, 13, . . .
always	13, 24,	12, 23, . . .
ærlig	39, . . .	12, 16, . . .

(a) Eksempel 1.

	Norsk	Tysk
unter	33, . . .	32, 25, . . .
über	37, 14, . . .	32, 13, . . .
yngre	37, 26, . . .	36, . . .

(b) Eksempel 2.

Figur 4.3: Eksempler på `strxfrm` og vekter. Eksemplene viser kun tilstrekkelig antall vekter til at det er mulig å se hvilken rekkefølge strengene vil sorteres i. Vektene er produsert med UTF-8-kodede strenger og `strxfrm` fra glibc 2.3.1. For norsk (bokmål) ble *locale* `no_NO.utf8` brukt, for tysk `de_DE.utf8` og for engelsk `en_GB.utf8`.

og vi kan se at *ü* og *y* begge har vekten 39 og derfor behandles likt. Med den tyske sorteringen blir derimot rekkefølgen *über*, *unter*, *yngre*, og *u* og *ü* er behandlet likt. Det andre eksempelet viser at ett tegn i den opprinnelige strengen kan oversettes til flere vekter. Den norske sorteringen av *absolute*, *always*, *ærlig* blir *absolute*, *ærlig*, *always* på engelsk, da tegnet *æ* er tilordnet de samme vektene som tegnsekvensen *ae*. I andre sammenhenger, for eksempel ved digrafer, kan det motsatte skje, nemlig at en tegngruppe oversettes til én vekt.

4.5.2 Unicode Collation Algorithm

Unicode Collation Algorithm (UCA) er en sorteringsalgoritme laget for sortering av Unicode-strenger beskrevet i Unicode Technical Standard 10 [17]. I UCA består sorteringsnøklene av vekter fra flere **vektingsnivåer** som brukes for å kunne skille mellom flere tegnegenskaper på en gang. For romersk-latinsk skrift brukes vektingsnivåene til alfabetisk sortering av grunntegn, sortering etter diakritika og sortering etter store eller små bokstaver. Sorteringsnøklene bygges opp i tre trinn:

1. Hver inndatastreng normaliseres til kanonisk dekomponert form (normaliseringsform D).

2. En sekvens med **vektingselementer** bygges opp ved å iterere gjennom den normaliserte strengen. Vektingselementene hentes ved oppslag i en tabell, og oppslagene gjøres med så lange delstrenger som mulig.

3. For hvert vektingsnivå itereres det gjennom sekvensen av vektingselementer, og vektene i hvert vektingsnivå plukkes ut og legges til sorteringsnøkkelen. Vekter som er lik null, skal ignoreres. Etter vektene fra ett vektingsnivå legges `0000` til som nivåskiller.

For strengen *cáb* vil normaliseringen gi fire tegn. Oppslag i vektingstabellen gir vektingselementene (`0A3D`, `0020`, `0002`), (`0A15`, `0020`, `0002`), (`0000`, `0032`, `0002`) og

Streng	Primære vekter	Sekundære vekter	Tertiære vekter
cab	0A3D 0A15 0A29	0020 0020 0020	0002 0002 0002
cáb	0A3D 0A15 0A29	0020 0020 0032 0020	0002 0002 0002 0002
Cab	0A3D 0A15 0A29	0020 0020 0020	0008 0002 0002
dab	0A49 0A15 0A29	0020 0020 0020	0002 0002 0002

Figur 4.4: Illustrasjon av vektingsnivåers funksjon. Tabellen viser sorteringsnøklene til fire strenger med ulikheter på forskjellige vektingsnivåer. Nivåskillerne mellom vektingsnivåene er utelatt. Eksempelet er basert på et eksempel i Unicode Technical Standard 10 [17], og vektene er en forenklet utgave av vektene i `allkeys.txt` versjon 3.1.1.

absolute	0A15, 0A29, ...	über	0BD7, 0A29, ...
always	0A15, 0B03, ...	unter	0BD7, 0B33, ...
ærlig	0A19, ...	yngre	0C07, ...

(a) Eksempel 1. (b) Eksempel 2.

Figur 4.5: Eksempel på sortering med standardtabellen for UCA. Kun de vektene som er relevante for sorteringen, er vist. Vektene er tatt fra `allkeys.txt` versjon 3.1.1.

(0A29, 0020, 0002). Den første delen av sorteringsnøkkelen bygges opp ved å sette sammen de primære vektene og legge til en nivåskiller: (0A3D, 0A15, 0A29, 0000). Blant de sekundære vektene skal vekten for aksenten ignoreres da den er null, mens alle de tertiære vektene skal med. Resten av sorteringsnøkkelen blir da: (0020, 0020, 0032, 0020, 0000, 0002, 0002, 0002, 0002).

Disse sorteringsnøklene kan binærsammenlignes, og ettersom vektene fra hvert vektingsnivå er plassert samlet, oppnår man at ulikheter på høyere vektingsnivåer ignoreres når det finnes ulikheter på et bestemt vektingsnivå. I figur 4.4 forekommer for eksempel den første ulikheten mellom strengen *cáb* og strengen *Cab* på det sekundære vektingsnivået, og det tertiære vektingsnivået ignoreres.

Sammen med UCA følger en standardtabell for vektingselementer, *The default Unicode collation element table*, distribuert som en fil med navnet `allkeys.txt`. Denne inneholder vektingselementer som tar hensyn til ekvivalente tegn (både kombinasjonsekvivalens og andre ekvivalenstyper som ikke er omtalt her), og som forøvrig anses som «kulturelt nøytral». I figur 4.5 ser vi at dette blant annet innebærer at *æ* er et eget tegn sortert etter alle ord som begynner med *a*, og at *ü* behandles som *u*. Store deler av tabellen, først og fremst på grunn av det store antallet ideografiske tegn, fordeler vektingselementene slik at rekkefølgen vil bli den samme som ved en binærsortering av kodepunkter.

Standardtabellen vil måtte tilpasses for å gi den sorteringen som forventes i en bestemt språklig sammenheng. Tilpasningene gjøres ved å modifisere eller legge vektingselementer til standardtabellen. Disse tilpasningene er ikke del av UCA og over-

lates til implementasjoner.

Blant sorteringsreglene for et språk finnes det sjelden regler for tegn fra fremmede skriftsystemer, og sorteringen av disse, såkalt ***out-of-scope sorting***, kan baseres på vektingselementene i standardtabellen, siden denne gir kulturelt nøytral sortering. Det samme gjelder irrelevante tegn som geometriske figurer, matematiske symboler og lignende.

Flere egenskaper ved spesielle typer sortering kan ikke oppnås med skreddersydde tabeller alene, men overlates til preprosessering. For eksempel kan man i bibliografisk sortering kreve at det ikke tas hensyn til artikler som «en» og «den» i begynnelsen av titler, eller at forkortelsen «St.» sorteres på samme måte som «Sankt». Derimot er standardtabellen og UCA tilrettelagt med **variabel vekting** av skilletegn og skilletegn for enkel tilpasning til forskjellige krav til behandling av disse. Det samme gjelder den spesielle franske prioriteringen av aksenter.[7]

4.5.3 Vurdering

I virkelig flerspråklig programvare er *locale*-modellen til ISO C90 bortimot ubrukelig, da bytte av *locale* er en potensielt tidkrevende funksjon og skjer for hele den kjørende prosessen. I tillegg er **locale** koblet med tegnsett, slik at tegnsett også veksles når **locale** byttes. Alle systemer har ikke de samme kombinasjoner av definisjoner for språk, dialekter og tegnsett, så bruk av **locale**-funksjonene introduserer også en problematisk systemavhengighet.

Den største svakheten med UCA er at det ikke finnes noen standardutgave av tilpasningene. Heldigvis finnes det *open source*-prosjekter som lager tilpasningstabeller for UCA, for eksempel IBMs ICU-prosjekt,[8] men bruken av disse er foreløpig lite utbredt.

4.6 Private områder

Områdene `U+E000`–`U+F8FF`, `U+F0000`–`U+FFFFF` og `U+100000`–`U+10FFFF`) i Unicode er reservert for programvareutviklere eller sluttbrukere som trenger spesielle tegnsett til sine applikasjoner [75: 13.5], eller de kan brukes for *round-tripping* av andre tegnkodinger [15: 1.1]. De private områdene brukes ofte av applikasjoner til representasjon av glyfvarianter når ett kodepunkt tilsvarer flere ulike glyfer.

[7] I fransk regnes ulikheter mellom aksenter fra slutten av ordet og ikke forfra slik som i de aller fleste andre språk. For eksempel vil både fransk og engelsk sortere *pécher* foran *pêcher*, men etter franske regler skal *pêche* gå foran *péché* da aksentulikheten i slutten av ordet er viktigere enn den i begynnelsen av ordet [75: 5.17].

[8] Se `http://oss.software.ibm.com/icu/`.

U+000C	FORM FEED	Begynner en ny side.
U+2028	LINE SEPARATOR	Forstetter teksten på begynnelsen av samme linje.
U+2029	PARAGRAPH SEPARATOR	Begynner et nytt avsnitt.
U+200C	ZERO WIDTH NON-JOINER	Hindrer en ligatur.
U+200D	ZERO WIDTH JOINER	Fremtvinger en ligatur.
U+034F	COMBINING GRAPHEME JOINER	Betyr at de to tilstøtende tegnene skal tolkes som én grafemisk enhet.
U+2060	WORD JOINER	Usynlig ordmellomrom som ikke tillater linjebrytning mellom ordene.
U+2000	EN QUAD	Horisontalt mellomrom.
U+2001	EM QUAD	Horisontalt mellomrom.
U+2002	EN SPACE	Horisontalt mellomrom.
U+2003	EM SPACE	Horisontalt mellomrom.
U+2004	THREE-PER-EM SPACE	Horisontalt mellomrom.
U+2005	FOUR-PER-EM SPACE	Horisontalt mellomrom.
U+2006	SIX-PER-EM SPACE	Horisontalt mellomrom.
U+2007	FIGURE SPACE	Horisontalt mellomrom.
U+2008	PUNCTUATION SPACE	Horisontalt mellomrom.
U+2009	THIN SPACE	Horisontalt mellomrom.
U+200A	HAIR SPACE	Horisontalt mellomrom.
U+205F	MEDIUM MATHEMATICAL SPACE	Horisontalt mellomrom.
U+00A0	NO-BREAK SPACE	Ordmellomrom som ikke tillater linjebryting mellom ordene.
U+202F	NARROW NO-BREAK SPACE	Smalt mellomrom som ikke tillater linjebrytning.
U+FEFF	ZERO WIDTH NO-BREAK SPACE	Usynlig mellomrom som ikke tillater linjebrytning.
U+2011	NON-BREAKING HYPHEN	Bindestrek som ikke tillater linjebryting.
U+00AD	SOFT HYPHEN	Bindestrek som kun synes ved orddeling.

Tabell 4.5: Unicode-kodepunkter for typografiske tegn. Beskrivelsene av tegnene er basert på Unicode 3.0 [75], Unicode Technical Report 20 [21] og Unicode Standard Annex 14 [25].

4.7 Unicode og typografi

Unicode er en informasjonsutvekslingsstandard og ikke en standard for typografisk koding. Grensene er her ikke alltid helt klare, og Unicode er ikke alltid konsistent på dette området [33]. Endel tegn som mange vil anse som **typografiske tegn**, er med i Unicode (se tabell 4.5 for en oversikt).

Flere av disse kodepunktene er med i Unicode for å sikre *round-tripping* med andre tegnkodingsstandarder; andre er med for å unngå flertydig bruk av andre kontrolltegn [21].

I XML og andre oppmerkingsspråk kan enkelte av de typografiske tegnene være problematiske fordi de kan være i konflikt med oppmerkingen eller kan håndteres

bedre ved oppmerking. Unicode Technical Report 20 [21: 3.2] fraråder for eksempel å benytte tegn som U+2029 PARAGRAPH SEPARATOR og U+2028 LINE SEPARATOR, da disse gjerne overlapper med oppmerking av typen `<xhtml:p>...</xhtml:p>` og `<xhtml:br/>`.

4.7.1 Glyfvarianter

I utgangspunktet avsettes det ikke egne kodepunkter for presentasjonsformer eller glyvarianter i Unicode. For eksempel er det kun ett kodepunkt for de to glyfene «a» og «*a*». Dette medfører også at man ikke tar hensyn til spesielle språklige tradisjoner; for eksempel plasseres det diakritiske tegnet diaeresis gjerne noe lavere over tegn i tysk enn i fransk [32], men begge variantene har ett kodepunkt i Unicode.

Hva som er presentasjonsformer, er heller ikke like alltid å avgjøre. Skillet mellom medial og final form av den greske bokstaven sigma er en kontekstuell egenskap, men de har likevel separate kodepunkter. Dette kan være nødvendig i spesielle sammenhenger for å skille mellom situasjoner som φιλοσ. og φίλος., der den første strengen er en forkortelse av et ord, kanskje φιλοσοφία, mens den andre er fullt ord etterfulgt av et setningspunktum [30: 1.2].

Valg av glyfer overlates til programvaren, som altså må ha informasjon om språk og sammenheng for å kunne velge riktige glyfer. Noen ganger kan glyfvalget påvirkes av kontrolltegn, slik som ZERO WIDTH NON-JOINER, som brukes til å hindre estetiske ligaturer[9] av typen «ff» og «fi». Dette er for eksempel nødvendig i det tyske ordet *Auflage* som skal presenteres «Auflage», og ikke «Auflage» [32].[10]

[9] **Estetiske ligaturer** er ligaturer som eksisterer kun av estetiske årsaker, for lesbarhet eller ved tradisjon, slik som *fi*. Dette i motsetning til **ligvistiske ligaturer**, som *æ* i norsk, som er uunnværlige for å kunne skrive et språk, og **kontekstuelle ligaturer** som opptrer i bestemte sammenhenger i alle språk som benytter skriftsystemet, for eksempel arabisk ـبـ + ـحـ → ـبحـ [29].

[10] Det finnes flere spesielle problemtilfeller knyttet til denne typen ligaturer. I germansk orddelingstradisjon skal konsonantsekvenser forenkles og ord deles etter sammensetninger. Sammen fører disse reglene til tilfeller av typen *oppussing*:*opp-pussing* [25]. Dette kan videre føre til vekslinger i ligaturbruk som i det tyske ordet *Schiffahrt*:*Schiff-fahrt* [29].

Kapittel 5

En ny bibliografiprosessor

En ny bibliografiprosessor må kunne håndtere flerspråklige bibliografier, gjøre det enkelt å lage nye stiler, være portabel og kunne brukes med flere typesettings- og dokumentprosesseringssystemer. Vi er først og fremst interessert i systemer som benytter oppmerkingsspråk, det vil si for eksempel TEX/LATEX, DocBook, TEI og XHTML, men det bør være mulig å tilpasse prosessoren til tekstbehandlingsprogrammer senere.

5.1 Videreutvikling eller nyimplementasjon?

Som vi har sett, finnes det allerede mye bibliografisk programvare. Borsett fra fokuset på flerspråklighet, skiller ikke en flerspråklig bibliografiprosessor seg mye fra det tradisjonelle mønsteret. Et naturlig spørsmål er da hvorvidt det er mulig å basere en flerspråklig bibliografiprosessor på allerede eksisterende programvare.

5.1.1 Videreutvikling av BIBTEX og TEX-pakker

Vi kan ta utgangspunkt i bibliografiprogramvaren for TEX og LATEX enten ved å modernisere og utvide BIBTEX eller ved å ta utgangspunkt i pakker som `m-bib`, `amsrefs` eller CAMEL.

Skal BIBTEX brukes som utgangspunkt, må programmet moderniseres, utvides til å støtte Unicode, og stilspråket må endres slik at det blir mer fleksibelt og tilbyr flere funksjoner – spesielt må det legges til en mekanisme for inklusjon av kildekode. Mange av ideene bak mlBibTeX og clBibTeX burde kunne brukes i et slikt arbeide. Det burde være forholdsvis enkelt å endre eller utvide den eksisterende BIBTEX, men den nære bindingen mellom BIBTEX-programmet og det problematiske databasefilformatet gjør dette til et lite attraktivt alternativ.

Den andre muligheten, en makropakke lik CAMEL, med stilspråk innebygget i

makropakken og databaser på et nytt format, løser langt på vei ulempene med en BibTeX-utvidelse. Dessuten er TeX-kode svært portabel, og integrasjon av bibliografiprosessoren med TeX og LaTeX vil være mulig. Siden det finnes svært mange makropakker med tildels avanserte og meget spesialiserte funksjoner, er det et godt grunnlag for å kunne bygge en slik makropakke basert på eksisterende programvare. Den avgjørende innsigelsen er at dette begrenser systemet til TeX.

5.1.2 Videreutvikling av andre bibliografiprosessorer

Ettersom videreutvikling av de kommersielle produktene er utelukket for oss, gjenstår kun kombinasjonsprogrammene *RefDB* og *pybliographer*. Begge disse opererer med interne dataformater som må endres betydelig for å oppnå flerspråklighet, bruker utilfredsstillende stilspråk og er i skrivende stund umodne og lite brukte prosjekter.

5.1.3 Nyimplementasjon

Hovedtrekkene ved en nyimplementasjon vil være de samme som for eksisterende bibliografiprosessorer: Programmet samler inn opplysninger fra referansedatabaser og dokumenter, formaterer referansene og siteringene, og skriver disse ut i et egnet presentasjonsformat. De prinsipielle spørsmålene ligger først og fremst i forholdet til stiler, og vi har sett at dette kan gjøres på iallfall to forskjellige måter:

1. Prosessoren leser stilspesifikasjoner. Spesifikasjonene er statiske beskrivelser av stilene, som tolkes av prosessoren, og denne har innebygget alle funksjoner som er nødvendig for korrekt formatering. Fordelen med dette er konsise og enkle beskrivelser av stilene. Dette prinsippet er benyttet i *RefDB*.

2. Prosessoren leser stilprogrammer. Programmene kan være skrevet i et vanlig programmeringsspråk, kanskje det samme som prosessoren selv, eller i et eget språk laget for formålet. Prosessoren tilbyr ferdige funksjoner for de grunnleggende stilprimitivene og for kommunikasjon med resten av prosessoren. Fordelen med dette er full frihet til å avgjøre formateringen av referansene og siteringene. Dette er prinsippet som brukes i BibTeX.

En kombinasjon av disse mulighetene vil kunne balansere fordelene ved spesifisering av stiler og friheten til å kunne programmere egne stilmoduler. Hvis de vanligste siteringsteknikkene er dekket av generelle stilprogrammer, kan detaljene, som faktisk utgjør størstedelen av forskjellene mellom stiler, plasseres i spesifikasjonsfiler.

Konkret kan dette realiseres ved lage en **stilmotor** for hver av hovedstilene som ble beskrevet i kapittel 2. En stilspesifikasjon vil således kunne velge for eksempel stilmotoren for forfatter-dato-stiler og angi parametre til denne, for eksempel det maksimale antallet forfattere som skal listes opp i en referanse.

5.2 Referansedatabaseformat

Ettersom ingen av de eksisterende referansedatabaseformatene som vi har sett på, er særlig godt egnet til flerspråklige bibliografier, er det naturlig å lage et nytt format. Det bør være et mål for dette formatet at det kan representere alle bibliografiske opplysninger som kan finnes i BibTEX-, RIS-, TEI-, DocBook- og `bibx.dtd`-filer, og at distinksjonene som kan representeres i disse, også kan representeres i det nye formatet.

5.3 Programmeringsspråk

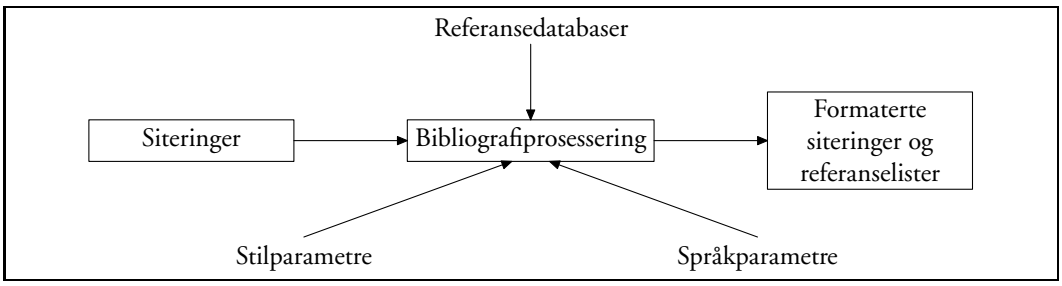
Hvis vi antar at inn- og ut-data for bibliografiprosessoren er i form av XML, enten ved at datafilene fra dokumentprosesseringssystemene er i XML, eller ved at man benytter preprosessorer, kan bibliografiprosessoren i sin helhet skrives i XSLT, akkurat slik som stilarkene for DocBook og TEI også behandler bibliografier. En ren XML- og XSLT-løsning har fordelen av å være portabel, idet grunnleggende XML-programvare finnes for de aller fleste moderne plattformer. XSLT er i prinsippet også tilstrekkelig ekspressivt til at dette skulle være mulig [53], men dette betyr ikke at det er en særlig praktisk løsning fra et implementasjonssynspunkt.

Det finnes XML-støtte for alle moderne programmeringsspråk, så dette er ikke avgjørende for valg av språk. Derimot er det avgjørende at språket tilbyr enkel og sikker strengmanipulasjon med god Unicode-støtte. Dette gjør nyere versjoner av Perl eller Python til fristende valg, og siden portabilitet er viktigere enn hastighet i dette tilfellet, er det ikke noe i veien for å foretrekke et *scripting*-språk fremfor for eksempel C, C++ eller Java. Det bør også være et språk som er håndterbart for sluttbrukere, da det er mulig at de må skrive stilmotorer selv, noe som gjør for eksempel LISP-lignende språk til mindre egnede kandidater.

5.4 Informasjonsflyt

Etter mønster fra bibliografiprosessorene som vi så på i kapittel 3, kan vi beskrive bibliografiprosessoren som et program som leser en eller flere **referansedatabaser** og en eller flere **siteringslister**, og som produserer **formaterte siteringer og referanselister**. Siteringslisten er en liste av identifikatorer ekstrahert fra dokumentet. De må være på en slik form at de kan sammenknyttes med de korrekte referansene fra referansedatabasen. Referansedatabasen er en mengde med identifikatorer der hver identifikator er assosiert med en samling av referanseopplysninger.

Prosessen er underlagt endel parametre som kan samles under betegnelsen **stil-**



Figur 5.1: Inn- og ut-data. Figuren viser datakildene og parametrene for en bibliografiprosessor.

parametre. I tillegg må det, for at denne prosessen skal kunne utføres korrekt i et flerspråklig miljø, introduseres **språkparametre** (se figur 5.1).

5.4.1 Stilparametre

Valget av siteringsstil kan enten gjøres i selve dokumentet eller angis som parameter til prosesseringen. Dersom man velger å lage makroer for LaTeX som tilsvarer dem som benyttes sammen med BibTeX idag, vil stilvalget gjøres av en makro tilsvarende `\bibliographystyle{<stil>}` i dokumentet, og stilparametrene er dermed del av dokumentet. Både TEI og DocBook overlater derimot alle presentasjonsvalg til XSLT-stilarkene, så for disse må stilvalget gjøres ved å sende med parametre til bibliografiprosessoren.

5.4.2 Språkparametre

De fleste dokumenter som bibliografiprosessoren vil brukes for, vil antagelig være ettspråklige, og i slike tilfeller er språkvalget en konstant. Benyttes derimot flere språk i dokumentet, viser det seg at det iallfall kan skilles mellom tre språkvariabler for dokumentet, siteringene og referansene.

I et dokument med bidrag på flere språk kan vi ikke anta at de samme språklige og typografiske reglene gjelder på ethvert sted i dokumentet. Det er likevel oftest mulig å peke ut ett språk som dominerer dokumentet, for eksempel ved at titler og overskrifter er på dette språket. I virkelig flerspråklige dokumenter, der titler og overskrifter er på flere språk, vil dokumentspråket kunne sies å være det språket som dikterer den helhetlige typografiske utformingen, for eksempel marger, skrifttyper og symbolbruk. Jeg vil kalle dette språket for **dokumentspråket**, og vi vil se at dokumentspråket oftest avgjør **referanselistespråket**, som er det språket som er assosiert med den helhetlige typografiske og språklige utformingen av en referanseliste.

Innen hver liste vil hver referanse være assosiert med et **referansespråk**. Referansespråket avgjør språket til referansens nøkkelord og de typografiske konvensjonene som følges når referansens deler settes sammen.

Referansespråket kan avgjøres på to måter. Ved en **dokumentavhengig tilnærmelse** vil referansespråket være det samme som dokumentspråket, mens det er det samme som referansens språk, slik dette er angitt i en referansedatabase, ved en **referanseavhengig tilnærmelse** [40].

I dette eksempelet (fra [83]) er referanselistespråket og referansespråket fransk, noe vi ser av overskriften og av at bynavnet Venezia er oppgitt på fransk, og det er derfor en dokumentavhengig tilnærmelse:

> Références
>
> [1] E. Layton, *The Sixteenth Century Greek Book in Italy. Printers and Publishers for the Greek World*, coll. Library of the Hellenic Institute of Byzanitine and post-Byzantine Studies 16, Venise 1994.

En referanseavhengig tilnærmelse illustreres av dette referanselisteutdraget (fra [94]):

> Список литературы
>
> 105. Зельдович Я Б, Райзер Ю П *Физика ударных волн и высокотемпературных гидродинамических явлений* (М.: Наука, 1966)
>
> 106. Tenorio-Tagle G, in *Violent Star Formation from 30 Doradus to QSOs* (Ed. G Tenorio-Tagle) (Cambridge: Cambridge Univ. Press, 1994) p. 50

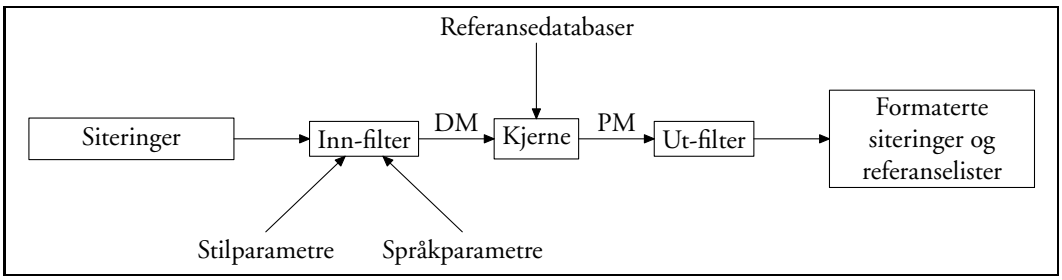
Språket vil også kunne veksle for hver opplysning lest ut fra referansedatabasene. Dette forekommer ofte med en dokumentavhengig tilnærmelse, når bestemte felters språk ikke er det samme som dokumentspråket. Vi trenger derfor et **områdespråk** for hvert område generert av en bestemt del av referansedatabasen.

Situasjonen er tilsvarende for siteringer. Siteringene forekommer i dokumentteksten, og siden språket i teksten kan endres på et hvilket som helst sted, vil siteringene måtte formateres i henhold til **siteringsspråket**. Siteringsspråket er enten det språket som benyttes i den umiddelbart omkringliggende teksten, eller det som benyttes i en videre omkringliggende del av teksten. Det siste tilfellet opptrer når en sitering forekommer i en del av teksten som anses å være på et fremmedspråk. I LaTeX kommer skillet mellom disse to typene vekslinger til uttrykk ved de to makroene `\selectlanguage` og `\foreignlanguage` i `babel`-pakken.

Hvis siteringene inneholder opplysninger tatt fra selve referansene, slik som i forfatter-tittel- eller forfatter-dato-systemer, vil språket kunne veksle hvor som helst inne siteringen, og vi må derfor operere med et områdespråk i siteringer, på samme måte som i referanser.

5.4.3 Filtre og mellomkode

Hvis bibliografiprosessoren skal brukes sammen med ulike dokumentprosesserings- og typesettingssystemer, må implementasjonsdetaljene ved systemene skjules ved en passende abstraksjon. Dette kan gjøres med **inn-filtre** og **ut-filtre** (se figur 5.2) som



Figur 5.2: Filtre og mellomkode. Inn- og ut-filtrene sørger for å samle inn og oversette data mellom de systemavhengige kildene og mellomkoden. DM står for dokumentmellomkode, og PM for presentasjonsmellomkode.

oversetter mellom den systemavhengige representasjonen og bibliografiprosessorens interne representasjon. Den systemuavhengige prosesseringen utføres i **kjernen**, og kommunikasjonen mellom denne og filtre gjøres i **mellomkode**; inn-filtrene produserer dokumentmellomkode og ut-filtrene presentasjonsmellomkode.

Inn-filtrenes hovedoppgave er å gjennomgå dokumentene og andre relevante filer og å bygge opp siteringslister fra disse. Som vi har sett, kan stil- og språk-parametrene være angitt i dokumentet, og inn-filtrene må derfor også ta hensyn til disse. Til forskjell fra pakker som `amsrefs` og `m-bib` ønsker vi i vår modell at referansedatabasene skal være adskilte fra dokumentene, og de skal derfor ikke behandles av filtrene.

Ut-filtrene produserer siteringer og referanselister på presentasjonsform. Dette kan som regel gjøres ved en enkel direkte oversettelse fra presentasjonsmellomkoden til filer som dokumentprosesserings- eller typesettingssystemets kan lese.

Legg merke til at det er mulig å tenke seg situasjoner der inn-filteret og ut-filteret ikke er tilknyttet samme dokumentprosesseringssystem. En artikkelforfatter kan ønske å gjøre dette når et dokument med referanser skal oversendes en utgiver, og forfatteren ønsker at kun det utvalget av referansedatabasen som faktisk er sitert i dokumentet, vedlegges. Dette er da mulig å gjøre ved å bruke et inn-filter for dokumentet og et ut-filter for bibliografiprosessorens referansedatabaseformat.

Kapittel 6

Databaser

I kapittel 3 presenterte jeg blant annet en oversikt over eksisterende formater for referansedatabaser og kommenterte endel svakhetstrekk ved disse, særlig i forbindelse med flerspråklighet. Blant formatene var flere basert på XML, og vi skal først se på hvorfor jeg finner det naturlig å basere et nytt referansedatabaseformat på nettopp XML. Jeg vil deretter gjennomgå hovedtrekkene i et forslag til en innholdsmodell og deler av en DTD som beskriver denne.

6.1 Hvorfor XML?

Det er en lang rekke grunner til at XML er et egnet valg for et nytt referansedatabaseformat. En praktisk grunn er at det finnes et stort utvalg ferdige programkomponenter for XML til de aller fleste programmeringsspråk, og disse frigjør oss fra blant annet å skrive en parser for databaseformatet. De lar oss også enkelt oversette databaser til eventuelle andre formater med standardverktøy som for eksempel XSLT-prosessorer.

Et svært viktig poeng for et flerspråklig format er at XML har mekanismer for å angi tegnkoding. XML ble laget med Unicode i tankene, og enhver XML-parser må støtte iallfall UTF-8 og UTF-16 [8: 2.2].

Det viser seg også at den frie uttrykksformen og den hierarkiske strukturen til XML er godt egnet for referansedatabaser. XML-dokumenter beskrives ofte som trær, samtidig som det fremheves at «tradisjonelle» strukturer som tabeller, for eksempel fra en relasjonsdatabase, også kan beskrives med XML. Begge disse aspektene er av interesse for referansedatabaser, ettersom de på den ene side er tabulariske – slik BibTeX-filene klart avslører – men på den annen side innholder mye informasjonen som best beskrives hierarkisk og med vekslende krav til typing.

For brukere vil det være en klar fordel at bibliografiprosessoren benytter et tekstbasert format, slik at databasene kan skrives for hånd med vanlige tekstbehandlings-

programmer, og slik at de til en viss grad er selvdokumenterende – forutsatt at navngivelse av elementer og attributter er fornuftig og presis. Det kan også argumenteres for at en innholdsmodell for XML enkelt kan utvides uten at bakoverkompatibilitet brytes ved å innføre nye elementer, attributter og lovlige verdier i modellen.

6.2 Grunnlag for modellen

Som utgangspunkt for modellen har jeg brukt BibTeX-formatet og vanlige utvidelser av dette. Videre har jeg tatt med de aller fleste ideer som opptrer i de ulike nyere XML DTD'ene for bibliografier som ble gjennomgått i kapittel 3, for eksempel `bibx.dtd`. Den viktigste kilden er likevel de ulike stilguidene og standardene som ble presentert i kapittel 2. Ettersom det vesentlige tillegget i forhold til de eksisterende formatene ligger i inkorporasjonen av språkavhengigheter, har jeg lagt liten vekt på de generelle trekkene ved modellen. Isteden beskrives for eksempel former av personnavn og ulike representasjoner av oversettelser og translitterasjoner av titler utfyllende.

6.3 Begreper og notasjon

Ettersom begrepene som benyttes i forbindelse med XML er noe flytende og ofte er overbenyttede uttrykk som «noder», «informasjon» og «data», er det nødvendig å presisere endel terminologi. I tillegg vil det være temmelig tungvint å diskutere et nytt referansedatabaseformat uten en kompakt notasjon som uttrykker den hierarkiske strukturen. For slike beskrivelser inneholder XML-eksempler altfor mye informasjon, og DTD-fragmenter er ikke tilstrekkelig presise.

Begrepene er i første omgang basert på XML Information Set [12]. Alle velformede XML-dokumenter som oppfyller visse navneromsbegrensninger har en **informasjonsmengde** (engelsk *information set*). Dette er en form for abstrakt modell for dokumentet på samme måte som de vanlige tre- eller tabell-fremstillingene av XML. Informasjonsmengden består av **informasjonsenheter** (engelsk *information items*), og hver informasjonsenhet har en mengde med navngitte **egenskaper** (engelsk *properties*). XML Information Set definerer elleve informasjonsenheter, men kun tre av disse er nødvendige i denne diskusjonen. En **elementinformasjonsenhet** (engelsk *element information item*) finnes for hvert element i det tilsvarende XML-dokumentet. **Barn** (engelsk *children*) av en elementinformasjonsenhet er en egenskap ved elementinformasjonsenheten og er en ordnet liste av informasjonsenheter på et dypere hierarkisk nivå i det tilsvarende XML-dokumentet. **Attributter** (engelsk *attributes*) til en elementinformasjonsenhet er videre en uordnet liste med **attributtinformasjonsenheter** (engelsk *attribute information items*), enten oppgitt i det tilsvarende XML-

dokumentet eller med standardverdier i DTD'en. Hver attributtinformasjonsenhet har en **normalisert verdi** (engelsk *normalized value*) som egenskap. I tillegg trenger vi en definisjon for tegndata: En **tegninformasjonsenhet** finnes for ethvert tegn i det tilsvarende dokumentet, og selve tegnkoden er en egenskap ved tegninformasjonsenheter.

Til det følgende XML-fragmentet

```
<notes xml:lang="en">
  <note>Also available as audio recording</note>
</notes>
```

er `notes` og `note` elementinformasjonsenheter, `xml:lang` en attributtinformasjonsenhet og hvert av tegnene i strengen *Also␣available␣as␣audio␣recording* en tegninformasjonsenhet. Egenskapene til `notes` er barnet `note` og attributtet `xml:lang`, egenskapene til `xml:lang` er den normalisete verdien `en`, alle tegninformasjonsenhenene utgjør egenskapen barn til `note`, og for hver tegninformasjonsenhet er den tilsvarende tegnkoden en egenskap.

For å unngå den noe omstendelige definisjonen av tegninformasjonsenheter, vil jeg benytte begrepet **streng** om konkateneringen av alle tegninformasjonsenheter som følger umiddelbart etter hverandre i en liste med informasjonsenheter. Videre er en **strenginformasjonsenhet** en elementinformasjonsenhet som kun har en streng som barn. *Whitespace* som kan settes inn mellom elementer for å gjøre dokumenter mer leselige, tar vi ikke med i denne modellen, og det overlates til implementasjonen å håndtere dette fornuftig.

6.3.1 Strengtyper

Ettersom vi søker å oppnå en modell som er godt egnet for beskrivelse av flerspråklige referanser, vil det vise seg nyttig å assosiere grad av språkavhengighet med strenginformasjonsenheter. De vil derfor inndeles i tre **strengtyper**:

- Strenger som er språkavhengige. Dette er antagelig den største gruppen og omfatter all form for fritekst, slik som titler på verker eller navn på organisasjoner eller forlag.

- Strenger som er språkuavhengige. Dette vil først og fremst være innhold myntet på maskinell behandling slik som URI'er, men også koder som ISSN og ISBN.

- Strenger som er «abstrahert bort» fra språkavhengighet. Et eksempel på dette er månedsangivelser i den gregoriansk-julianske kalenderen. Disse kan representeres ved tall i intervallet 1 til 12 uten tap av informasjon.

Notasjon	Beskrivelse
d	språkavhengig streng
i	språkuavhengig streng
a	abstrakt streng

Tabell 6.1: Strengtyper. Distinksjonen mellom de tre strengtypene ligger i den ulike graden av språkavhengighet.

I den videre diskusjonen vil jeg benytte henholdsvis d, i og a (for henholdsvis *dependent*, *independent* og *abstract*) som notasjon for disse typene (se tabell 6.1), og en strenginformasjonsenhet X av strengtype t noteres X_t. For enkelhets skyld vil det antas at alle attributtverdier er språkuavhengige strenger – motivasjonen bak dette valget er at det er umulig å beskrive attributter med attributter, og at det defor ikke er mulig å angi hvilket språk attributtverdien er assosiert med.

Det er viktig å legge merke til at det er det fullt mulig for en strenginformasjonsenhet å ha vekslende strengtype fra referanse til referanse. En angivelse av en utgave kan være så enkel som «1. utgave», og denne kan kategoriseres som strengtype a, mens andre utgaveangivelser omfatter fritekst av en eller annen form, for eksempel «utvidet utgave», og må karakteriseres som strengtype d. I modellen må da forekomster av disse tilfellene skilles i to forskjellige strenginformasjonsenheter, eller strengtype d må velges for begge tilfeller.

6.3.2 Samlingstyper

Elementinformasjonsenheter som kun har andre elementinformasjonsenheter som barn, inndeles i **samlingstyper**. Først og fremst er vi interessert i om ordningen til listen over barn er av betydning, og jeg skiller derfor mellom samlingstypen **sekvens** og samlingstypen **bag**, der ordningen henholdsvis er relevant og irrelevant. Inspirert av vanlig terminologi for avbildningslignende typer i programmeringsspråk kaller jeg elementinformasjonsenhetene i listen over barn for **nøkler** og samtlige egenskaper ved hver slik informasjonsenhet for tilhørende **verdier**, det vil si at sekvenser og bager inneholder par av nøkler og verdier for alle barn.

I dette eksempelet

```
<imprint>
  <publisher>Universitetsforlaget</publisher>
  <pubdate>Mai 1999</pubdate>
  <place>Oslo</place>
<imprint>
```

kan elementinformasjonsenheten `imprint` tenkes realisert som en bag, da den innbyrdes rekkefølgen til elementinformasjonsenhetens barn, `publisher`, `pubdate` og `place`, er irrelevant. Bagen vil bestå av par, slik som elementinformasjonsenheten `publisher` og dennes egenskaper. Dette vil jeg notere som

Notasjon	Betegnelse	Beskrivelse
M	en-til-en-bag	Én nøkkel er assosiert med én verdi. Rekkefølgen til nøkkel-verdi-parene er irrelevant.
N	en-til-mange-bag	Én nøkkel er assosiert med en eller flere verdier. Rekkefølgen til nøkkel-verdi-parene er irrelevant.
R	en-til-en-sekvens	Én nøkkel er assosiert med en eller flere verdier. Rekkefølgen til nøkkel-verdi-parene er relevant.
S	en-til-mange-sekvens	Én nøkkel er assosiert med en eller flere verdier. Rekkefølgen til nøkkel-verdi-parene er relevant.
C	valg	Én nøkkel er assosiert med én verdi. Kun ett par kan forekomme.

Tabell 6.2: Samlingstyper. De fem ulike samlingstypene er samlinger bestående av nøkkel- og verdipar. Forskjellene mellom typene utgjøres av hvorvidt hver nøkkel kan forekomme i flere par og om rekkefølgen av nøkkel-verdi-parene spiller noen rolle, i tillegg til en spesiell samlignstype som kun tillater ett nøkkel-verdi-par.

$$
\begin{aligned}
&imprint_M = \{\\
&\quad publisher_d = \{\text{Universitetsforlaget}\},\\
&\quad date_d = \{\text{Mai1999}\},\\
&\quad place_d = \{\text{Oslo}\}\\
&\}
\end{aligned}
$$

der M angir samlingstypen bag. Legg merke til at denne notasjonen gir oss muligheten til å utsette valget mellom elementer og attributter, da egenskapene barn og attributter noteres på samme måte.

Det er også nyttig å skille **en-til-en-sekvenser** og **en-til-en-bager**, i hvilke en bestemt nøkkel kun forekommer i ett par, fra **en-til-mange-sekvenser** og **en-til-mange-bager**, der en bestemt nøkkel forekommer i flere par. Notasjonen for disse typene vil være henholdsvis M, N, R og S.

I tillegg til dette vil vi trenge en spesiell samlingstype, et **valg**, notert C, som kun inneholder ett nøkkel-verdi-par blant flere angitte muligheter. Jeg vil notere dette som

$$edition_C = \{freetextedition_d | arabicedition_a\},$$

Tabell 6.2 viser en oversikt over samlingstypene og notasjonen for disse.

6.4 En enkel modell

Hvis vi i første omgang ser bort fra referanser der det refereres til navngitte deler innen et verk (for eksempel en artikkel i en bok) og også utsetter endel detaljer knyttet til språkavhengighet og oversettelser, vil vi kunne beskrive en modell som ligner mye på for eksempel BibTeX-formatet. Hovedtrekkene kan fremstilles ved å inndele alle elementinformasjonsenhetene i seks grupper:

- Informasjon om ansvarlige.
- Tittelinformasjon.
- Publikasjonsinformasjon.
- Informasjon om avgrensning.
- Omfangsinformasjon.
- Deskriptiv informasjon.

I tillegg til at disse gruppene innholder logisk avgrenset informasjon, gjenspeiler de den grupperingen av informasjon som ofte forekommer i presentasjonen av en referanse.

For hver referanse må det finnes en form for unik indentifikator som kan brukes ved siteringer. Etter vanlig praksis er denne identifikatoren kalt id, og den bør realiseres som et attributt. Dette blir da skjelettet for en referanse:

$$
\begin{aligned}
&entry_M = \{ \\
&\quad id_i, \\
&\quad authors, titles, imprint, scope, extent, description \\
&\}
\end{aligned}
$$

Her tilsvarer $authors$, $titles$, $imprint$, $scope$, $extent$ og $description$ de seks ovenfornevnte gruppene. La oss nå se nærmere på innholdet av hver av innholdsenhetene.

6.4.1 Informasjon om ansvarlige

De ansvarlige for en publikasjon omfatter først og fremst alle forfattere av publikasjonen. Dette er ofte kun én person, men i enkelte vitenskapelige miljøer er det ikke uvanlig at det er et dusin eller flere forfattere (fra [89]);

> [16] D. Finzi, M. Hermankova, T. Pierson, L. M. Carruth, C. Buck, R. E. Chaisson, T. C. Quinn, K. Chadwick, J. Margolick, R. Brookmeyer, J. Gallant, M. Markowitz, D. D. Ho, D. D. Richman, and R. F. Siliciano, *Identification of a reservoir for HIV-1 in patients on highly active antiretroviral therapy*, Science, 278 (1997), pp. 1295–1300.

Forfattere behøver heller ikke være personer; også en organisasjon kan være forfatter av et verk (fra [92]):

> American Psychiatric Association (1994). *Diagnostic and statistical manual of mental disorders* (4th. ed) (DSM-IV). Washingon, DC: Author.

author	Forfatter
coauthor	Medforfatter
collaborator	Samarbeidspartner
editor	Redaktør
translator	Oversetter
collector	Innsamler

Tabell 6.3: Ansvarlige for en publikasjon. Tabellen viser de vanligste funksjonene som ansvarlige for en publikasjon kan ha. Mulige benevnelser for de ulike funksjonene er vist i venstre kolonne.

For alle disse tilfellene er det et ufravikelig krav i bibliografisk praksis at rekkefølgen av forfattere slik de er oppgitt på publikasjonens tittelside, skal bevares ved all gjengivelse. Vi må derfor benytte samlingstypen sekvens for forfatterlisten.

Videre kan personer og organisasjoner ha andre roller enn forfatterrollen, men likevel være ansvarlige for verket i den forstand at de opptrer på verkets tittelside. Her tenker jeg på roller som redaktør, oversetter og lignende (se tabell 6.3). Det leder til to mulige formuleringer av sammensetningen av sekvenser med ansvarlige:

- En sekvens som består av personer og organisasjoner. Hver person eller organisasjon har en funksjon i forbindelse med publikasjonen, for eksempel forfatter, redaktør eller oversetter.

- En sekvens som består av forfattere, redaktører, oversettere og så videre. Hver forfatter, redaktør, oversetter og lignende er enten en person eller en organisasjon.

At det første alternativet nok er det heldigste valget, ser vi ved å observere at den nærmere beskrivelsen av personer og organisasjoner vil være svært ulik. Skal vi for eksempel kunne sortere referanser etter ansvarlige, vil navn på personer kreve en helt annen beskrivelse enn navn på organiasjoner. Noe tilsvarende skille finnes ikke mellom beskrivelsene av forfattere, redaktører og oversettere.

Et par spesialtilfeller bør også håndteres. Det kan hende at ikke alle de ansvarlige for verket er kjent, så vi må kunne markere at en liste over ansvarlige ikke er fullstendig. Dette vil stilene vanligvis håndtere ved å legge til «med flere» eller lignende etter forfatternavnene. Videre kan det hende at verket ikke har noen kjent forfatter. Enkelte stiler vil da plassere «anonym» på forfatterens plass, mens andre vil begynne referansen med verkets tittel [1: 16.41]. Da disse spesialtilfellene er komplementære, kan det hele uttrykkes ved én informasjonsenhet, for eksempel $completeness$ med verdiene $complete$, $incomplete$ og $anonymous$.

Referansedatabaseformatet `bibx.dtd` inkluderer i likhet med enkelte uoffisielle BibTeX-formater e-postadresser, internettadresser, og, for personer, organisatorisk tilhørighet. Selv om dette sjelden brukes i presentasjonen av referanser, kan det være

meget nyttig for brukere av referansedatabasene, og det bør derfor tas med. Den sammenfattede modellen for forfatterinformasjon vises i figur 6.1 under informasjonsenheten $authors$.

6.4.2 Tittelinformasjon

Det finnes sjelden mer enn én tittel for et verk. Den vanligste variasjonen er en undertittel – noe som ikke alltid er like enkelt å identifisere.

> [1] Robert I. Binnick. *Time and the verb: A guide to tense and aspect*. Oxford University Press, New York, 1991.

Stiler vil iallfall kreve spesiell tegnsetning i forbindelse med undertitler [1: 15.109, 16.61], så disse må skilles ut i en egen informasjonsenhet (se informasjonsenheten $titles$ i figur 6.1 for modell).

`bibx.dtd` opererer i tillegg med tittelvarianter som *alternate*, *short*, *abbrev[iated]* og *user*. Jeg har ikke sett eksempler på slike titler i noen sammenheng, og jeg har ikke klart å finne noen dokumentasjon som skulle tilsi at de er nødvendige.

6.4.3 Publikasjonsinformasjon

Publikasjonsinformasjon omfatter i streng forstand en utgiver, publikasjonsstedet og publikasjonstidspunktet, men ofte regnes også angivelse av utgivelse og numre som ISBN til publikasjonsinformasjonen. I figur 6.1 vises en modell for dette under informasjonsenheten $imprint$.

Utgiver, altså den person eller den organisasjon i hvis autoritet og navn et verk offentliggjøres, kan oppgis på flere måter – ofte vil navnet på en eller annen måte forkortes. Publikasjonsstedet er oftest en by, men hvis det anses nødvendig, tar man med land, stat eller til og med utgivers fulle adresse.

Landsnavn kan abstraheres til landskoder fra ISO 3166 [49], som for eksempel «NO» for Norge eller «GB» for Storbritannia.[1] For delstatsnavn finnes ingen internasjonal standard, men ulike nasjonale standarder kan brukes isteden. Hvorvidt lands- og statsnavn oppgis sammen med stedsnavnet, er bundet til valg både av stil og språk. For å gjøre referansedatabasen stiluavhengig, må vi derfor kreve at alle publikasjonssteder oppgis med lands- og statskoder.

Publikasjonstidspunktet faller som regel i en av gruppene

Dag, måned og årstall Vanligst ved moderne elektroniske publikasjoner.

Måned og årstall Brukes for numre av tidsskrifter og for konferansepublikasjoner.

[1] I likhet med ISO 639 [47], som angir koder for språk, finnes landskodene i ISO 3166 i to varianter. «alpha-2» er den opprinnelige typen og er forkortelser på to bokstaver. En senere utvidelse, «alpha-3», bruker trebokstavsforkortelser og listen over lovlige forkortelser ble utvidet. Heldigvis er variantene tillatt brukt om hverandre.

$$
\begin{aligned}
&authors_S = \{ \\
&\quad completeness_i, \\
&\quad person_M = \{function_i, personname, url_i, email_i, affiliation_d\}, \\
&\quad entity_M = \{function_i, entityname_d, url_i, email_i\} \\
&\} \\
&\\
&titles_M = \{maintitle_d, subtitle_d\} \\
&\\
&imprint_M = \{ \\
&\quad pubname_d, \\
&\quad pubplace_M = \{fulladdress_d, city_d, country_a, state_a\}, \\
&\quad pubdate_C = \{fulldate_a | monthyear_a | year_a | pubtime_a\}, \\
&\quad edition_C = \{freetextedition_d | arabicedition_a\}, \\
&\quad version_d, \\
&\quad locators_N = \{ \\
&\qquad lccn_i, mrnumber_i, doi_i, isbn_i, issn_i, isrc_i, isrn_i, ismn_i, isan_i, sici_i, bici_i, \\
&\qquad coden_i, pmid_i, uri_i bibcode_i, oai_i, pii_i \\
&\quad \}, \\
&\quad numbers_S = \{standardnumber_d, othernumber_d\} \\
&\} \\
&\\
&scope_S = \{ \\
&\quad unit_M = \{ \\
&\qquad type_i, \\
&\qquad value_C = \{range_M = \{start_d, end_d\} | single_d\} \\
&\quad \} \\
&\} \\
&\\
&extent_M = \{dimension_d, price_d, medium_d\} \\
&\\
&description_N = \{ \\
&\quad keywords_S = \{keyword_d\}, \\
&\quad annotations_S = \{annotation_d\}, \\
&\quad categories_S = \{category_d\}, \\
&\quad notes_S = \{note_d\}, \\
&\quad contents_d, abstract_d \\
&\}
\end{aligned}
$$

Figur 6.1: Modell for en referanse uten komposisjon.

Årstall Brukes for bøker.

Fritekst Brukes for eldre bøker. Denne typen oppstår vanligvis fordi det nøyaktige utgivelsestidspunktet ikke er kjent.

Alle variantene krever en hel del spesiell språk- og stilavhengig behandling, og de bør derfor tildeles egne informasjonsenheter. De to første tidspunktangivelsene kan med fordel angis på standardformatet beskrevet i ISO 8601 [51], det vil si for eksempel *2003-07-16* for 16. juli 2003 eller *2003-07* for juli 2003.

For angivelser av utgaver er standardeksempelet på formen «12. utgave», som enkelt kan abstraheres da utgaveangivelsen alltid er numerisk. Dette kan brukes for språkavhengige presentasjoner som «12. Auflage» og «12^{e} édition». Dette fungerer ikke for tilfeller som «12. reviderte utgave» eller «Abridged version». Idet utgaveangivelser med arabertall på ordinalform er såpass vanlige, og ettersom de lar oss gjøre spesiell språkavhengig formatering, er det på sin plass å ha en egen innholdsenhet for disse. Resten av tilfellene må anses å være fritekst.

Enkelte dokumenttyper, særlig standarder, brukerveiledninger og tekniske dokumenter på elektronisk form, oppdateres stadig og tildeles derfor versjonsnumre. Versjonsnumre synes å være kvalitativt ulike utgaveangivelser, da de indikerer en versjon av et dokument, mer enn en bestemt utgivelse av et dokument. For versjonsangivelser finnes det ikke noen iøynefallende abstraksjonsmuligheter, da disse finnes på en lang rekke former, så disse kategoriseres enklest som språkavhengig fritekst.

I de ulike referansedatabaseformatene har jeg funnet sytten ulike former for identifikasjonskoder (se tabell 6.4). Disse er ment å være unike for hvert verk, men i praksis forekommer det at man ønsker å oppgi flere slike numre av samme type for en og samme referanse, idet det for eksempel tildeles forskjellige ISBN for *hardcover*- og *paperback*-utgivelser av bøker. Felles for alle disse identifikasjonsnumrene er at de ikke er språkavhengige – de fleste er kombinasjoner av tegn og bokstaver som skal behandles maskinelt.

Relatert til identifikasjonsnumrene er andre ikke-standardiserte numre som settes på publikasjoner av bestemte typer. Dette kan være standardnumre, som for publikasjoner utgitt av standardiseringsorganisasjoner, for eksempel «ISO-690 (E)»:

> [1] International Organization for Standardization. *ISO 690: Documentation — Bibliographic references — Content, form and structure. ISO-690 (E)*. International Organization for Standardization, Geneva, 1987.

Eller numre på rapporter (fra [80]):

> Räihä, K.-J., M. Saarinen, M. Sarjakoski, S. Sippu, E. Soisalon-Soininen, and M. Tienari [1983]. "Revised report on the compiler writing system HLP78," Report A-1983-1, Dept. of Computer Science, University of Helsinki.

Informasjonsenhet	Navn
lccn	Library of Congress Control Number (LCCN)
mrnumber	Mathematical Review number
doi	Digital Object Identifier (DOI)
isbn	International Standard Book Number (ISBN)
issn	International Standard Serial Number (ISSN)
isrc	International Standard Recording Code (ISRC)
isrn	International Standard Technical Report Number (ISRN)
ismn	International Standard Music Numbering (ISMN)
isan	International Standard Audiovisual Number (ISAN)
sici	Serial Item and Contribution Identifier Standard (SICI)
bici	Book Item and Component Identifier (BICI)
coden	CODEN
pmid	PubMed Unique Identifiers (PMID)
uri	Uniform Resource Identifiers (URI)
bibcode	Bibliographic Reference Coding (Bibcode)
oai	Open Archives Initiative (OAI)
pii	Publisher Item identifier (PII)

Tabell 6.4: Identifikasjonskoder. Tabellen viser en oversikt over ulike mer eller mindre standardiserte systemer for tildeling av unike identifikasjonskoder til publikasjoner.

Jeg har valgt samlingstypen sekvens for disse, ettersom det godt kan være flere av disse numrene, og deres innbyrdes rekkefølge kan ha betydning. Skillet mellom standardnumre og andre numre er ikke strengt nødvendig, men synes fornuftig å ta med, da dette lar oss lage tittelforkortelser til forfatter-dato-systemet for standarddokumenter.

6.4.4 Avgrensning av deler

Referanser kan omfatte deler av verker, og angivelser av disse delene, **delavgrensninger**, forekommer i et mangfold av varianter. Standardeksemplene er denne gangen et sideintervall, for eksempel «s. 100–102», eller ett bestemt kapittel i et verk. Vi må altså kunne uttrykke både intervaller av slike logiske inndelingsenheter og enkeltstående enheter. For hvert slikt enhetsintervall og for hver enkeltstående enhet må vi angi hva slags enhet det er snakk om, slik at stilene kan velge riktige nøkkelord og forkortelser (for eksempel «pp. 10–11» for et sidetallsintervall på engelsk, men «p. 10» for en enkeltstående side).

Å forutsi alle mulige logiske enheter som kan benyttes i inndelinger, er umulig, så vi må nøye oss med å identifisere de vanligste og la språkavhengig fritekst samle opp de gjenværende tilfellene. Hierarkiske delavgrensninger er så kompliserte og uforutsigbare at jeg synes de bør anses som fritekst. Informasjonsenheten *scope* i figur 6.1 viser denne modellen. Det er ikke uvanlig at man for eksempel refererer «avsnitt 4 til 5 av kapittel 3 samt avsnitt 1 av kapittel 4», og ønsker at dette skal vises som «3.4–5, 4.1» i én sammenheng, og som «3.4–3.5, 4.1» eller til og med «3.4–

4.1» i andre sammenhenger. Å konstruere en modell for dette blir fort komplisert og dermed villedende, og den vil uansett neppe være istand til å tilfredsstille alle behov.

6.4.5 Omfangsinformasjon

Fullstendige bibliografiske opplysninger omfatter også omfangsbeskrivelser. Dette er vanlig praksis i katalogisering for biblioteker, og i noen bibliografistiler presenteres denne informasjonen. Normalt inkluderes en angivelse av verkets medium, pris og dimensjoner – det siste angitt ved hjelp av enten fysiske størrelser eller en eller annen logisk oppdeling som sidetall. Denne informasjonen kan deles opp på en mer finkornet måte, men idet denne informasjonen sjelden tas med og er uhyre mangfoldig, bør innholdsmodellen presentert under informasjonsenheten $extent$ i figur 6.1 være tilstrekkelig.

6.4.6 Deskriptiv informasjon

Den deskriptive informasjonen om en referanse er først og fremst viktig i kommenterte bibliografier. Slike bibliografier inneholder en beskrivelse av hver referanse, og listen er gjerne inndelt i grupper etter tema eller et eller annet klassifikasjonssystem. Her sier det seg selv at det må finnes en mulighet for å uttrykke kommentarene til hver referanse i referansedatabasen, men det kan også være fornuftig å ta med informasjon som ikke nødvendigvis brukes i slike kommenterte bibliografier, men er essensiell dersom man ønsker å la referansedatabasen være grunnlaget for en større, søkbar bibliografi. Her tenker jeg på nøkkelord som beskriver innholdet, kategorier etter en egnet klassifikasjon, sammendrag av verkets innhold og verkets innholdsfortegnelse. Et generelt notatfelt er ofte brukt som «søppelbøtte» og for personlige notater, og i modellen som vises under informasjonsenheten $description$ i figur 6.1, er notatfeltet også tatt med blant den deskriptive informasjonen.

6.5 Komposisjon

De ulike innfallsvinklene for modellering av komposisjon illustreres best ved hjelp av et eksempel. La oss se på en av de vanligste referansetypene i vitenskapelig publikasjon, nemlig en artikkel i et vitenskapelig tidsskrift:

> [1] Nelson Beebe. Bibliography prettyprinting and syntax checking. *TUGboat*, 14(4):395–419, desember 1993.

Typisk for denne referansetypen er at det refereres til en artikkel i et bestemt nummer av en publikasjon som utgis kontinuerlig med jevne mellomrom. Artikkelen har forfatter og tittel, mens tidsskriftet kun oppgis med tittel. Artikkelens plassering i rekken av numre av tidsskriftet angis ved hjelp av årgang («14»), nummer («4»),

sidetallsintervall («395–419»), årstall («1993») og, i dette tilfellet, også månedsnavn («desember»).

Vi kan velge å se på dette som en komposisjon av en artikkel og et tidsskriftnummer. Sidetallsintervallet angir da hvilken del av nummeret vi er interessert i, mens resten av opplysningene er egenskaper ved selve nummeret.

Dette kan representeres ved hjelp av en «flat» struktur tilsvarende den som benyttes i tradisjonelle referansedatabaser, her eksemplifisert ved hjelp av et sterkt forenklet XML-vokabular:

```
<entry id="beebe93">
  <authors>Nelson Beebe</authors>
  <titles>Bibliography prettyprinting and syntax checking</titles>
  <journaltitles>TUGboat</journaltitles>
  <imprint><pubdate>1993-12</pubdate></imprint>
  <scope>
    <issue>14</issue><volume>4</volume>
    <pages>395--419</pages>
  </scope>
</entry>
```

title tilsvarer her det som ofte betegnes som analytisk tittel i bibliotekskataloger, mens *journaltitle* blir tittelen på det aktuelle periodikum.

Eventuelt kan vi benytte oss av muligheten for å uttrykke hierarkier av XML-elementer og oppnå noe som ligner dette:

```
<entry id="beebe93">
  <authors>Nelson Beebe</authors>
  <titles>Bibliography prettyprinting and syntax checking</title>
  <scope><pages>395--419</pages></scope>
  <in>
    <titles>TUGboat</titles>
    <imprint><pubdate>1993-12</pubdate></imprint>
    <scope><issue>14</issue><volume>4</volume></scope>
  </in>
</entry>
```

Denne varianten kan varieres ved å gjøre den rekursiv, slik at det er mulig med enda flere komposisjonsnivåer, og man kan også la *in* inneholde en peker til en annen referanse med opplysningene om periodikumet.

Med flere nivåer enn to kan det være fristende å gå enda mer systematisk til verks og dele opp artikkelreferansen i tre komponenter: En for selve artikkelen innen tidsskriftnummeret, en for tidsskriftnummeret innen årgangen og en for tidsskriftet som kontinuerlig rekke av utgivelser.

Da avgrenses tidsskriftnummeret fra utgaverekken ved nummer og årgang, der deres *alter ego* årstall og månedsnavn er egenskaper ved nummeret. Sidetallsintervallet

avgrenser på sin side artikkelen innen nummeret. To innvendinger kan reises mot dette: Hvorfor lager vi ikke også et eget nivå for årganger? Og siden det forekommer tidsskrifter, særlig tyske, som er utgitt i ulike serier, bør det vel være et femte nivå? Man kan og innvende at sidetallsintervallet ikke avgrenser artikkelen fra nummeret, men fra årgangen, fordi de fleste tidsskrifter har kontinuerlig sidetallsnummerering innen hver årgang.

Det er klart at dette ikke fører oss videre – vi er ikke interessert i en fullstendig klassifikasjon av tidsskrifter, men trenger kun sørge for at referanseopplysningene er tilstrekkelig presise, og at disse kan presenteres riktig. Med dette *in mente* kan vi isteden begrense oss til tre bibliografiske nivåer med en nokså liberal, men praktisk definisjon inspirert av TEI [72: 6.10.2.1]:

Analytisk nivå Dette beskriver en enhet som er ikke er publisert uavhengig. Dette vil oftest si at enheten er del av en monografi eller et nummer av et periodikum.

Monografisk nivå Beskriver en enhet som er publisert uavhengig. Vanligvis vil dette være en monografi eller et nummer av et periodikum.

Serienivå Beskriver den serien som en selvstendig publisert enhet er del av. Dette vil være en serie av monografier eller periodika.

Her er det analytiske nivået det **høyeste bibliografiske nivået** og serienivået det **laveste bibliografiske nivået**. Dette er svært likt nivåene som benyttes av `bibx.dtd`, men i denne benyttes navnene *work*, *publication* og *set*.

Legg merke til at med en slik oppdeling vil for eksempel tidsskrifter og antologier behandles på monografisk nivå. For en tidsskriftartikkel vil altså selve artikkelen tilhøre det analytiske nivå og nummeret av tidsskriftet det monografiske nivå. Vi vil dog ikke kreve at det angis et serienivå for rekken av tidsskriftutgaver, ettersom disse opplysningene ikke er nødvendige for vårt formål. Derimot vil en bestemt artikkel i et leksikon plasseres på analytisk nivå, et bind av leksikonet på monografisk nivå, og hele leksikonserien på serienivå.

Med en slik tredeling kan vi skille de tre nivåene fra hverandre med denne konstruksjonen:

```
<entry id="beebe93">
  <analytic>...</analytic>
  <monographic>...</monographic>
  <series>...</series>
</entry>
```

Å realisere nivåene som elementer med unike navn er fordelaktig, da det lar oss regulere hvilke kombinasjoner av nivåer som er tillatte ved hjelp av DTD'en. Det er ikke meningsfylt å oppgi et analytisk nivå og et serienivå uten et monografisk nivå. Det er antagelig heller ikke meningsfylt å ha referanser som kun har et analytisk nivå.

Men også her oppstår det komplikasjoner, for en artikkel kan publiseres i flere tidsskrifter. Bibliotekspraksis er da vanligvis å opprette to separate oppføringer for de to publikasjonene, mens det i referanselister vel er like vanlig å samle dem i én referanse. Skal vi tillate slike tilfeller, er det altså lite å tjene på å bruke DTD'en til å begrense kombinasjonsmulighetene.

Fra dette kan vi igjen gå videre til å la hver referanse i referansedatabasen være knyttet til ett bibliografisk nivå og uttrykke komposisjon ved å kryssreferere til andre referanser:

```
<entry id="beebe93" level="analytic">
  ...
  <in ref="tugboat"/>
</entry>

<entry id="tugboat" level="monographic">
  ...
</entry>
```

Dette gjør det riktignok vanskelig å gjenskape BibTeXs adferd med kommandoen `\nocite{*}`, fordi flere referanser enn de som faktisk er i bruk, vil bli med i referanselisten, men dette er et nokså ubetydelig problem.

Dette er antagelig den mest elegante løsningen på problemet, men dessverre oppstår det på nytt problemer med opplysningene som avgrenser enheter innen komposisjonen. Ettersom tidsskriftnummeret er plassert på det monografiske nivået, blir årgang og nummer egenskaper ved dette nivået, og vi vil trenge egne oppføringer for alle numre av tidsskriftet, noe som ikke gjør kryssreferanser særlig interessante:

```
<entry id="beebe93" level="analytic">
  <authors>Nelson Beebe</authors>
  <titles>Bibliography prettyprinting and syntax checking</titles>
  <in ref="tugboat4:14">
    <scope><pages>395--419</pages></scope>
  </in>
</entry>

<entry id="tugboat4:14" level="monographic">
  <titles>TUGboat</titles>
  <scope><volume>14</volume><issue>4</issue></scope>
  <imprint><pubdate>1993-12</pubdate></imprint>
</entry>
```

For å oppnå den ønskede effekten må vi på en eller annen måte benytte alle tre nivåer slik at tidsskriftnummeret blir monografisk og rekken av utgaver plasseres på serienivået.

Konklusjonen blir derfor at vi dessverre ikke kan løse dette på noen fullgod måte uten at representasjonen blir for komplisert. Kompromisset med tre bibliografiske

nivåer realiseres antagelig best ved hjelp av egne elementer for hvert av nivåene, der disse fritt kan gjentas i de kombinasjoner som meningsfylte.

Vi ser også at delavgrensninger er nært knyttet til komposisjon. Ved en komposisjon brukes delavgrensninger til å angi forholdet mellom de bibliografiske nivåene i komposisjonen. Delavgrensninger kan også forekomme på det høyeste bibliografiske nivået i en referanse og avgrenser da den refererte delen av det høyeste bibliografiske nivået. Vi skal skille mellom disse to bruksområdene for delavgrensninger ved hjelp av begrepene **komposisjonsavgrensninger** og **referanseavgrensninger**. Begge delavgrensningstypene kan representeres ved hjelp av informasjonsenheten $scope$.

6.6 Realisering av modellen

Den enkle modellen som inntil nå er beskrevet, kan representere de fleste bibliografiske opplysninger som kan forekomme i BibTeX- eller RIS-databaser. Senere vil vi utdype og utvide modellen, men først skal vi se generelt på hvordan den foreløpige modellen kan realiseres som en DTD.

Alle strengtypene kan realiseres som tegndata, men innholdstype d må ledsages av en eller annen form for språkinformasjon. Vi vil sener se på hvordan dette kan gjøres. For å tydeliggjøre de ulike innholdstypene kan det i DTD'en brukes parameterentiteter som for eksempel

```
<!ENTITY % data.abstr  '(#PCDATA)'>
<!ENTITY % data.static '(#PCDATA)'>
<!ENTITY % data.lang   '(#PCDATA)'>
```

En en-til-en-bag som $extent_M = \{price_d, dimensions_d, medium_d\}$ kan ikke uttrykkes fullt ut i en DTD, og realisasjonen

```
<!ELEMENT extent       (price?, dimensions?, medium?)>
```

er det nærmeste vi kommer. En-til-mange-bager som for eksempel $locators_N = \{isbn_i, issn_i, \ldots\}$ kan uttrykkes som

```
<!ELEMENT locators     (isbn|issn|...)*>
```

Og for en sekvens som $titles_S = \{maintitle_d, subtitle_d, \ldots\}$ må DTD'en inneholde

```
<!ELEMENT titles       (maintitle|subtitle|...)+>
```

Valg som $value_C = \{range, single\}$ realiseres som

```
<!ELEMENT value        (range|single)>
```

6.6.1 Underklassifikasjon

For endel en-til-mange-bager og sekvenser kan det være hensiktsmessig med en realisasjon som attributter. Dette er tilfellet for identifikasjonsnumrene i informasjonsenheten $locators$. De hører naturlig sammen og forekommer neppe ofte nok til at de fortjener egne elementer. Derfor kan de tenkes realisert slik:

```
<locators>
  <locator class="isbn">1-56592-398-7</locator>
</locators>
```

Her er det innført et element `locator` for selve nummeret og et attributt `class` med samme verdi som nøkkelen. Denne realisasjonen, som jeg vil kalle **underklassifikasjon**, er kun mulig dersom alle nøklene i bagen har samme strengtype.

Realisasjonen er ikke like egnet for en-til-en-bager, da det kan være vanskelig å kontrollere om det forekommer gjentagelser av attributtet `class` med den samme verdien i en bag – det er blant annet umulig å uttrykke en slik begrensning i en DTD. Det finnes likevel tilfeller der en slik realisasjon av en-til-en-bager er fornuftig, for eksempel for informasjonsenheten $titles_M$.

6.6.2 Utvidbarhet

En alvorlig svakhet ved tidligere referansedatabaseformater er den manglende muligheten for utvidelse av vokabularet, det være seg enten i form av videreklassifikasjon av eksisterende felter eller helt nye tillegg. Dette problemet kan neppe helt unngås, men det bør være mulig å tilby iallfall en grunnleggende utvidbarhet som ikke interfererer med de andre delene av formatet.

For å tillate videreklassifikasjon av eksisterende elementer kan vi innføre attributtet `role` på alle elementer. Dette attributtet er det vanlig å bruke til semantisk utvidelse eller som *escape hatch* [59: 8.4.1]. Hvis senere utgaver av innholdsmodellen og DTD'en omfatter den aktuelle utvidelsen eller begrensningen, kan elementer med den aktuelle `role`-verdien omgjøres til den nye modellen uten informasjonstap eller manuelle inngrep.

Å innføre helt nye elementer er ikke mulig, og det ville neppe vært noen god løsning å tillate dette, da det gjør validering umulig og gir opphav til varianter av referansedatabaseformatet. For alle samlingstyper er det likevel mulig å tillate innføring av helt nye informasjonsenheter, så lenge samlingstypen er realisert ved underklassifikasjon.

For de tilfellene der det er ønskelig å tillate introduksjonen av nye underklassifikasjoner som ikke var kjent da DTD'en ble laget, har jeg valgt å benytte det samme mønsteret som DocBook gjør i slike tilfeller. I tillegg til attributtet `class` finnes det et attributt `otherclass` som skal brukes når verdien til attributtet `class` er `other`, for eksempel slik:

```
<locators>
  <locator class="other" otherclass="mynumber">12345</locator>
</locators>
```

Ettersom samlingen av lovlige verdier av attributtet `class` og verdien `other` til sammen dekker alle tenkelige underklassifikasjoner, kan `class`-attributtet gjøres obligatorisk:

```
<!ELEMENT locator      (%data.static;)>
<!ATTLIST locator      class      (isbn|issn|...|other) #REQUIRED
                       otherclass CDATA                 #IMPLIED>
```

En variasjon av dette oppstår når det finnes en kandidat blant alternativene som forekommer oftere enn de andre:

```
<!ELEMENT title        (%data.lang;)
<!ATTLIST title        class      (maintitle|subtitle|other) 'maintitle'
                       otherclass CDATA                      #IMPLIED>
```

6.7 Navngivelse

Navngivelsen av elementer og attributter bør være gjennomtenkt for å unngå feilaktig bruk. Selv om et elements eller attributts bruksområde er godt dokumentert, vil navnet styre oppfatningen av innholdet [59: 4.1.5]. I DTD'er som skal leses og skrives av mennesker, bør navnene på elementer og attributter være presise, intuitive og følge aktuell terminologi. Slike presise navn fører ofte til lange navn, noe som kan være et problem dersom databasene skal skrives for hånd eller ved ressursbegrensinger [59: 8.5].

For denne DTD'en er det ikke hensikten at alle brukere skal skrive databasene manuelt – det bør lages hjelpemidler som grafiske grensesnitt for manipulasjon av databasene. Navnenelengden er altså ikke noe stort problem. Navnene er likevel eksponert for sluttbrukeren, så presisjonskravet må oppfylles. I tråd med vanlig praksis er det benyttet engelske navn, og betegnelsene er, som tidligere nevnt, tatt fra allerede eksisterende referansedatabaseformater.

6.8 Språk

For alle informasjonsenheter som har strengtype d, må språk markeres. Å realisere språk i form av attributter er nok det mest naturlige valget, ettersom dette attributtet kan plasseres på alle elementer som realiserer strengtype d. Videre er det en naturlig antagelse at en referanse i de fleste tilfeller først og fremst vil være ettspråklig.

Unntaksvis vil det forekomme virkelig flerspråklige referanser eller referanser der noen av informasjonsenhetene er på et annet språk. Derfor hadde det vært interessant å finne en metode som gjør det unødvendig å angi språkattributtet på alle elementer som realiserer strengtype d. Dette kan oppnås ved å la elementer arve språkattributtet fra sine foreldre. Denne arven kan ikke uttrykkes eksplisitt i en DTD og må utføres av programvaren, og er et av de vanlige bruksområdene for attributter med implisitte verdier i en DTD [59: 8.3.3, 9.5].

XML-standarden definerer attributtet `xml:lang` for alle elementer, og det er avsatt nettopp til beskrivelse av elementets språktilknytning og forventes arvet av alle underelementer [8: 2.12]. Verdiene til dette attributtet er språkidentifikatorer definert i RFC 1766 [3] av typen `en` eller `en-GB`, der første del angir språk som i ISO 639 [47] og andre del, som kan utelates, angir dialekt eller regionkode som i ISO 3166 [49].

For å være sikker på at hver referanse har en korrekt språkangivelse, synes jeg det er fornuftig å gjøre attributtet `xml:lang` obligatorisk på det ytterste nivået i en referanse. Det er neppe noen god idé å la referanser arve attributtet `xml:lang` fra det ytterste nivået i referansedatabasen, idet dette lett kan føre til feil dersom referanser fra ulike referansedatabaser slås sammen manuelt.

Legg merke til at dersom dette arvesystemet skal gjennomføres, må alle informasjonsenheter med samlingstype som inneholder en eller flere informasjonsenheter av strengtype d, også bære `xml:lang`-attributtet. Dette gjelder for eksempel informasjonsenheten $extent$. Parameterentiteten `lang.inherited` står her for definisjonen av `xml:lang`-attributtet og gjør det mulig for $price$, $dimensions$ og $medium$ å arve språk-attributtet fra $extent$.

```
<!ELEMENT extent       (price?, dimensions?, medium?)>
<!ELEMENT extent       %lang.inherited;
```

6.8.1 Oversettelser og translitterasjoner

Alt innhold av innholdstype d kan tenkes oversatt eller translitterert:

> П. Я. Черных, Историко-этимологический словарь современного русского языка, Русский язык, Москва, 1993. (P. Ja. Tsjernykh, Istoriko-etimologitsjeskij slovar' sovremennogo russkogo jazyka, Moskva, 1993.)

Innen visse grenser kan translitterasjon gjøres av programvare, men ofte vil dette, i likhet med all oversettelse, måtte innføres i referansedatabasene. Man kan igjen tenke seg flere muligheter for å representere de ulike oversettelses- og translitterasjonsalternativene. Den enkleste løsningen er å gjenta elementet for det aktuelle innholdet som skal oversettes, men med attributter som angir hva slags alternativ gjentagelsen inneholder:

```
<title xml:lang="ru">Историко-этимологический словарь
  современного русского языка</title>
<title xml:lang="no" variant="transliteration">
  Istoriko-etimologitsjeskij slovar' sovremennogo russkogo
  jazyka</title>
<title xml:lang="no" variant="translation">
  Historisk-etymologisk ordbok for det moderne russiske
  språk</title>
```

Dessverre er dette i praksis vanskelig å kombinere med de tidligere diskuterte informasjonsenhetene av en samlingstype, og det er umulig å uttrykke i en DTD at noen av disse elementene representerer originalteksten, og andre representerer alternativer.

For å løse dette, må vi igjen ty til en mer komplisert modell:

```
<title>
 <primary xml:lang="ru">Историко-этимологический словарь
   современного русского языка</primary>
 <alternative xml:lang="no" variant="transliteration">
   Istoriko-etimologitsjeskij slovar' sovremennogo russkogo
   jazyka</alternative>
 <alternative xml:lang="no" variant="translation">
   Historisk-etymologisk ordbok for det moderne russiske
   språk</alternative>
</title>
```

Det er likevel ingen grunn til at innhold uten alternativer skal måtte uttrykkes så tungvint, så oppmerking av typen

```
<title xml:lang="ru">Историко-этимологический словарь
  современного русского языка</title>
```

bør også tillates.

For den deskriptive informasjonen, som består av flere informasjonsenheter av strengtype d, er ikke tanken om en primærtekst og et eller flere alternativer like egnet, ettersom det ikke finnes noen klar kandidat for hva som er primærteksten. For enkelhets skyld ser jeg bort fra dette, og benytter derfor samme metode for å muliggjøre oversettelser av disse.

6.9 Preformaterte data

Det vil i noen sammenhenger være behov for å plassere kommandoer myntet på et bestemt ut-filter i databasene. Dette kan for eksempel skje dersom man ønsker at symbolet TEX skal produseres med et TEX- eller LATEX-filter, mens teksten «TeX»

skal benyttes i alle andre sammenhenger. Vi kan kalle dette for **preformaterte data** i den forstand at bibliografiprosessoren ikke skal foreta noen formatering av slike kommandoer.

I XML er prosesseringsinstruksjoner satt av for bestemte opplysninger som skal overføres uendret til prosesseringsapplikasjoner. Vi kunne tenke oss å benytte prosesseringsinstruksjoner til å uttrykke preformaterte data, men dette er neppe noen god løsning, da vi også ønsker å ha en standardtekst som er gyldig for alle ut-filtre. Egne elementer for en slik konstruksjon synes da å være eneste løsning:

```
<titleinfo>
  <title xml:lang="en">The <special><default>TeX</default>
    <filter name="tex">\TeX{}</filter></special>book</title>
</titleinfo>
```

En slik bruk av elementer blandet med tekst kalles **blandet innholdsmodell** (engelsk *mixed content model*) og fører til en rekke komplikasjoner under prosessering. I andre sammenhenger i DTD'en er *whitespace* uvesentlig utenfor elementer, men i XML-fragmentet ovenfor er elementet `special` del av innholdet til et annet element sammen med tekst. *Whitespace* blir med andre ord av betydning her. Programmet som leser referansedatabaser, må altså konstruere et tre eller en annen passende datastruktur som tillater at både tekstnoder og elementnoder kan forekomme i samme node.

6.9.1 Navn og akronymer i titler

En variant av problemet med preformaterte data oppstår i forbindelse med titler. Det er vanlig i blant annet engelsk å benytte versaler i begynnelsen av de fleste ord i titler. Dessverre er det slik at noen stiler ikke godtar denne formen, andre gjennomfører den i alle titler, og en tredje gruppe bruker den formen som finnes på verkets tittelside. I alle tilfeller må bibliografiprosessoren ha muligheten til å omforme titler til en form kun bestående av minuskler eller til en passende tittelform.

For omforming til minuskler må databasen inneholde tilstrekkelig informasjon om hvilke ord som har versaler som må bevares. Dette omfatter først og fremst egennavn, men også akronymer og endel spesielle uttrykk og forkortelser. Et eksempel på et element som beskytter et egennavn, er vist nedenfor:

```
<titleinfo>
  <title xml:lang="en">Mastering Algorithms
    with <protect>Perl</protect></title>
</titleinfo>
```

Vi ser at, som for preformaterte data, fører disse hensynene til en blandet innholdsmodell.

6.10 Personnavn

Personnavn i referansedatabaser er et komplekst problem. Navnene spiller en svært viktig rolle i sorteringen av referanser, og reglene for sortering av navn på ulike språk er meget sammensatte. For å lage formaterte siteringer og referanselister har vi behov for følgende former av navn:

- En nevneform N (på full form eller med initialer). Denne benyttes i referanselister.
- Invertert form I (på full form eller med initialer). Denne benyttes i referanselister som er sortert alfabetisk etter forfatters etternavn.
- En form S med etternavn alene. Denne formen brukes i siteringer i forfatter-dato-stiler.
- Navnet oppdelt i en sekvens K av strenger som er relevante for sortering. Sorteringsformen er nødvendig for all sortering basert på personnavn.

Enkle vestlige navn består av et eller flere fornavn $f_1, f_2, \ldots, f_n$ og et eller flere etternavn $s_1, s_2, \ldots, s_m$. Hvis vi definerer xy som konkateneringen av strengen x, et mellomrom og strengen y, kan vi la $f = f_1 f_2 \cdots f_n$ stå for alle fornavn og $s = s_1 s_2 \cdots s_n$ for alle etternavn. Nevneformen N blir da ganske enkelt

$$N = fs$$

For den inverterte formen I av et navn definerer vi funksjonen $\mathrm{comma}(x, y)$ til å være konkateneringen av strengen x, et komma, et mellomrom og strengen y. Vi får da at

$$I = \mathrm{comma}(s, f)$$

Videre blir

$$S = s$$

og, ettersom alfabetisering foretas på grunnlag av etternavn, blir sorteringsnøkkelen

$$K = (s_1, s_2, \ldots, s_m, f_1, f_2, \ldots, f_n)$$

La oss nå se på mer kompliserte tilfeller, der disse enkle uttrykkene ikke er tilstrekkelige.

6.10.1 Partikler

Navn med partikler er det mest kompliserte aspektet ved vestlige navn. Partikler forekommer først og fremst i fransk, spansk, portugisisk, italiensk, tysk og nederlandsk. Partiklene skrives vanligvis med små bokstaver og fjernes oftest når etternavnet forekommer alene [1: 7.8, 7.10]:

Nevneform	Etternavn
Giovanni da Verrazano	Verrazano
Alexander von Humboldt	Humboldt
Ludwig van Beethoven	Beethoven

Men dette gjelder altså ikke i alle tilfeller:

Nevneform	Etternavn
Bernard ter Haar	ter Haar
Wouter Van Twiller	Van Twiller

Når slike navn omtales i andre språk, følges ikke disse reglene alltid, så for kjente navn kan andre former være aktuelle [1: 7.10]:

Nevneform	Kjent som
Luca della Robbia	della Robbia
Vasco da Gama	da Gama
Vincent van Gogh	van Gogh

På samme måte er praksis for personer i engelsktalende land med slike navn varierende, men generelt regnes partikkelen som del av etternavnet [1: 7.8]:

Nevneform	Etternavn
Eugen D'Albert	D'Albert
Lee De Forest	De Forest
Walter de la Mare	de la Mare
Martin Van Buren	Van Buren
Wernher von Braun	von Braun

De franske artiklene *le*, *la* og *les*, samt sammentrekningene *du* og *des*, er spesielle, da de alltid skrives med stor bokstav. Preposisjonen *de* og dens eliderte form *d'* skrives derimot med liten bokstav. Med mindre etternavnet er et enstavelsesord fjernes *de* fra etternavnet når dette står alene, mens *du*, *des* og *d'* beholdes [1: 7.9, 26: 3.6.5, 69: 4.26]:

Nevneform	Etternavn
Alexis de Tocqueville	Tocqueville
Charles de Gaulle	de Gaulle
Paul de Man	de Man
Pierre d'Arcy	d'Arcy
François, duc de La Rochefoucauld	La Rochefoucauld
Jean de La Bruyère	La Bruyère
Philippe Du Puy de Clinchamps	Du Puy de Clinchamps
Bonaventure Des Péreiers	Des Péreiers

6.10.2 Ærestitler og generasjonsangiver

Selv om slike ikke anbefales brukt i bibliografier, opptrer av og til titler i referansedatabaser, og disse skal plasseres foran fornavnet, men er ikke relevante for alfabetiseringen.

Generasjonsangivelsene *junior* eller *senior* er postpositive partikler som skiller mellom far og sønn med samme navn [69: 4.2.5]. Det finnes flere mulige forkortelser, blant de vanligere er *jr.* og *sr.*. Romertall kan også brukes til generasjonsangivelse, og også disse er postpositive.

På invertert form skal en generasjonsangivelse plasseres etter navnet og et komma, men kommaet skal ikke være med i nevneformen [1: 17.87]:

Nevneform	**Etternavn**
Theodore Roosevelt Jr.	Roosevelt, Theodore, Jr.
Adlai E. Stevenson III	Stevenson, Adlai E., III

6.10.3 Initialer

Vestlige fornavn oppgis av og til på initialform, og denne formen består oftest av den første bokstaven i navnet. Unntak finnes for navn fra språk med digrafer og i translitterert navn, slik som i (fra [94]):

71. Efremov Yu N, astro-ph/0206408
72. Fendt Ch, Beck R, Neininger N *Astron. Astrophys.* **335** 123 (1998)

I dobbeltnavn kan begge navnekomponentene måtte settes på initialform: Initialformen av *Jean-Jaques Rousseau* er *J.-J. Rousseau* [24].

6.10.4 Spanske navn

Tradisjonelle spanske navn består av både farens og morens familienavn, i den rekkefølgen, tidvis sammenbundet av konjunksjonen *y* («og»). Alfabetisering gjøres alltid etter farens etternavn, men forkortelser til kun ett av navnene forekommer [1: 7.11, 17.110]:

Nevneform	**Etternavn**	**Invertert form**
Federico García Lorca	García Lorca	García Lorca, Federico
José Ortega y Gasset	Ortega y Gasset *eller* Ortega	Ortega y Gasset, José

Noen kvinner erstatter morens familienavn med *de* etterfulgt av mannens familienavn [1: 17.110]:

Nevneform	**Invertert form**
María Esquivel de Sánchez	Esquivel de Sánchez, María

Navnet alfabetiseres likevel som «Sánchez».

Artikler og preposisjoner brukes også i andre sammenhenger, og det er mulig med doble fornavn [1: 7.11, 17.111]:

Nevneform	Etternavn
Tomás de Torquemada	Torquemada
Bartolomé de Las Casas	Las Casas
José Murguía	Murguía
Augustín Pedro Justo	Justo

Avvik fra dette forekommer av og til ved navn som er kjent i Europa på en annen form [1: 7.11, 17.111]:

Nevneform	Kjent som
Federico García Lorca	Lorca
Manuel de Falla	de Falla

6.10.5 Russiske navn

Russiske patronymer er del av fornavnet, og disse brukes av og til alene i omtale eller tiltale av en person. I en referanseliste skal derimot etternavnet alltid være med [1: 7.14], og sortering gjøres etter dette:

Nevneform	Etternavn	Invertert form
Антон Павлович Чехов	Чехов	Чехов, Антон Павлович

6.10.6 Islandske og gamle skandinaviske navn

Etter skandinavisk tradisjon skal patronymer veksle fra generasjon til generasjon: *Magnus Pálsson* kaller for eksempel sin sønn *Pál Magnusson* og sin datter *Guðrún Magnúsdottir*). I dagens islandske navn alternerer ikke navnene, men patronymer brukes fremdeles og navn alfabetiseres etter fornavn, slik at *Vígdís Finnbogadóttir* sorteres under *V* [69: 4.2.8]

6.10.7 Ungarske navn

Ungarske familienavn står foran fornavnet, men rekkefølgen byttes vanligvis om i ikke-ungarsk sammenheng, og denne formen danner grunnlag for den inverterte formen [1: 7.15, 17.113], [26: 3.6.7]:

Nevneform	Nevneform (ikke-ungarsk)	Invertert form
Molnár Ferenc	Ferenc Molnár	Molnár, Ferenc
Bartók Bela	Bela Bartók	Bartók, Bela

6.10.8 Arabiske navn

Arabiske navn består vanligvis enten av fornavn og familienavn eller av fornavn, farens fornavn og familienavn. Disse alfabetiseres alle under familienavnet. Former av den bestemte artikkel (vanligvis *al*) er del av etternavnet, men skal ignoreres under alfabetisering. Artikkelen plasseres enten etter fornavnet eller foran etternavnet ved invertering [1: 9.94, 17.114]:

Nevneform	**Invertert form**
Zakir Husain	Husain, Zakir
Ahmad Hamid Hmisi	Hmisi, Ahmad Hamid
Tawfiq al-Hakim	Hakim, Tawfiq al- *eller* al-Hakim, Tawfiq
Muhammad Hamid al-Jamal	Jamal, Muhammad Hamid al- *eller* al-Jamal, Muhammad Hamid[2]

Andre prefikser som *Abu* («far til») og *ibn* («sønn av») er også del av etternavnet [1: 7.12]:

Nevneform	**Etternavn**
Syed Abu Zafar Navdi	Abu Zafar Navdi
Aziz ibn Saud	Ibn Saud

Legg merke til endringen til stor bokstav i prefikset *ibn*.

I vestlig sammenheng oppgis eldre navn vanligvis på den formen som de er blitt kjent i Vesten [1: 17.115].

6.10.9 Kinesiske, japanske, koreanske og vietnamesiske navn

I nevneform oppgis kinesiske, japanske og vietnamesiske etternavn foran fornavn [1: 7.13, 17.117–17.120]:

Språk	**Etternavn**	**Fornavn**	**Eksempel**
Kinesisk	Ett tostavelsesnavn[3]	Ett enstavelsesnavn	Zhao Wuji
Japansk	Ett flerstavelsesnavn	Ett flerstavelsesnavn	Kurosawa Noriaki
Koreansk	Ett en- eller tostavelsesnavn	Ett en- eller tostavelsesnavn[4]	Kim Jong-il
Vietnamesisk	To flerstavelsesnavn	Ett flerstavelsesnavn	Ngo Dinh Diem

Navn alfabetiseres etter familienavnet, inversjon foretas aldri, og navn skilles da heller aldri med komma [1: 17.117, 17.119–17.120]:

[2] Det er valgfritt hvilken av disse formene man ønsker å bruke.

[3] Stavelsene skrives i ett i pīnyīn, for eksempel *Zhao Wuji*, sammenbundet med bindestrek i Wade-Giles, for eksempel *Chao Wu-chi*. Eldre etternavn har bare en stavelse.

[4] Stavelsene i tostavelsesnavn skilles med bindestrek [69: 4.2.8].

Nevneform	Invertert form	Etternavn
Zhao Wuji	Zhao Wuji	Zhao
Kurosawa Noriaki	Kurosawa Noriaki	Kurosawa
Ngo Dinh Diem	Ngo Dinh Diem	Ngo
Kim Jong-il	Kim Jong-il	Kim

Kinesere og japanere med forbindelse til Vesten velger av og til vestlig navneform med familienavn sist. Slike former må inverteres på vanlig måte [1: 17.118–119]:

Nevneform	Invertert form
Tang Tsou	Tsou, Tang
H. H. Kung	Kung, H. H.
Noriaki Kurosawa	Kurosawa, Noriaki

6.10.10 Indiske navn

Moderne indiske navn opptrer vanligvis med familienavnet sist, og dette navnet brukes ved alfabetisering [1: 17.122]:

Nevneform	Invertert form
Jawaharlal Nehru	Nehru, Jawaharlal
Mohandas Karamchand Gandhi	Gandhi, Mohandas Karamchand
Bhabhani Bhattacharya	Bhattacharya, Bhabhani

6.10.11 Andre asiatiske navn

De fleste asiatiske navn er avledet fra europeiske språk, arabisk eller kinesisk [1: 17.126], men ulike mindre språk har egne navnetradisjoner. Et mønster som vi ikke har sett i noen av de omtalte språkene, er den tradisjonen som forekommer for blant annet burmesiske, javanesiske og indonesiske navn. Disse består kun av fornavn, og sortering gjøres etter dette [1: 17.123-124].

6.10.12 Regler for kompliserte personnavn

Med det enkle systemet som allerede er beskrevet, dekkes enkle vestlige navn uten partikler, kinesiske og japanske navn med vestlig skrivemåte og indiske navn. Ved å legge til parameteren r, som indikerer at nevneformen av navnet skal ha etternavnet før fornavnet, og at det ikke finnes noen spesiell inversjonsform, får vi også dekket tradisjonelle kinesiske og japanske navn, samt de vietnamesiske og koreanske navnene:

$$N = \begin{cases} fs & \text{hvis } r = 0 \\ sf & \text{hvis } r = 1 \end{cases} \qquad I = \begin{cases} \text{comma}(s, f) & \text{hvis } r = 0 \\ sf & \text{hvis } r = 1 \end{cases}$$

For å håndtere navn med partikler som er en del av etternavnet, men som ikke regnes med under alfabetisering, må vi skille ut disse partiklene. Vi kaller denne typen

partikler for **preposisjonspartikler** og betegner dem $p = p_1 p_2 \cdots p_d$. Vi trenger også en funksjon $\mathrm{punct}(x, y)$ som er konkateneringen av strengene x og y hvis x slutter med et skilletegn, og ellers er identisk med xy. Ved å erstatte s med $o = \mathrm{punct}(p, s)$ i uttrykkene for S, N og I, kan vi dekke arabiske navn og europeiske navn med partikler.

For tradisjonelle spanske navn må vi lage enda en gruppe, $m = m_1, m_2, \cdots m_o$ som består av de navnene som utgjør siste del av etternavnet, men aldri har betydning for sorteringen. I tillegg må vi ha med de postpositive generasjonsangivelsene, som kun er definert for navn med $r = 0$. Da blir

$$S = om$$

og

$$N = \begin{cases} fSg & \text{hvis } r = 0 \\ Sf & \text{hvis } r = 1 \end{cases} \qquad I = \begin{cases} \mathrm{comma}(\mathrm{comma}(S, f), g) & \text{hvis } r = 0 \\ N & \text{hvis } r = 1 \end{cases}$$

Mens sorteringsnøkkelen blir

$$K = (s_1, s_2, \ldots, s_m, f_1, f_2, \ldots, f_n, g)$$

Måten sekvensen K benyttes på, vil avhenge av om sorteringen er en ord-for-ord-sortering eller en bokstav-for-bokstav-sortering, ettersom inverteringskommaer i inverterte navn ikke regnes som del av ord, men alltid stanser sorteringen [1: 17.97]. I enkelte stiler vil den også påvirkes av at navn med initialer sorteres foran fulle navn [1: 17.99–100]:

> Oppenheimer, J. Robert
> Oppenheimer, James N.
> Oppenheimer, K. T.
> Oppenheimer, Keven S.

Sekvensen K må derfor strengt tatt også inneholde et merke som markerer alle initialer.

6.11 Forhold til Unicode

Ettersom koding angis i XML-dokumentets innledning, behøver vi ikke ta spesielle forhåndsregler for å håndtere referansedatabasenes tegnkoding. Det kan derimot være en fordel å kreve at normaliseringsform C benyttes, da denne sikrer oss problemfri transformasjon via XSLT til for eksempel HTML.

6.11.1 Typografiske tegn

Som vi så i avsnitt 4.7, omfatter Unicode en del tegn som vi kalte typografiske tegn. Noen av disse er interessante for referansedatabasene. Særlig gjelder dette U+00A0, U+00AD, U+200C, U+200D og U+2011 (se tabell 4.5). De fleste av disse har entydige oversettelser til for eksempel TEX, og de er av betydning for representasjonen av referanser i informasjonsenheter som omfatter fritekst.

Kodepunktene som frarådes brukt av Unicode Technical Report 20 [21], bør ikke brukes i referansedatabasene, da disse kan skape uheldige overlappinger med oppmerkingen. Kodepunkter i de private områdene kan heller ikke benyttes, da referansedatabasefilene er ment som et informasjonsutvekslingsformat.

6.11.2 Entiteter for Unicode-tegn

Ettersom det ikke alltid er mulig å skrive inn alle tegnene med et vanlig tastatur, eller man velger å bruke for eksempel ISO-8859-1 som koding av XML-dokumentet, virker det som en god idé å ha en alternativ fremgangsmåte for å uttrykke de typografiske tegnene. Entiteter er velegnet for dette, og med ISO-8859-standardene følger det tabeller over anbefalte entitetsnavn og tegn – opprinnelig for bruk i SGML. Mange brukere vil ha erfaring med disse fra HTML og XHTML, der deler av ISO-8859-entitetene og endel nye tillegg er del av standardene.

Sammen med DocBook XML DTD v4.4.1beta1 følger utvidede entitetstabeller med mange typografiske tegn fra UCS i tillegg til ISO-8859-entitetene, for eksempel

```
<!ENTITY nbsp   " "> <!-- NO-BREAK SPACE -->
<!ENTITY ndash  "&#x2013;"> <!-- EN DASH -->
<!ENTITY mdash  "&#x2014;"> <!-- EM DASH -->
<!ENTITY hellip "&#x2026;"> <!-- HORIZONTAL ELLIPSIS -->
```

Disse entitetstabellene er et godt utgangspunkt som dekker både de typografiske tegnene og en lang rekke andre tegn som ofte ikke er enkelt tilgjengelige på vestlige PC-tastaturer.

Kapittel 7

Mellomkode og stiler

Kjernens oppgave i bibliografiprosessoren er å oversette fra dokumentmellomkode til presentasjonsmellomkode ved hjelp av stilmotorer og stilspesifikasjoner. I prosessen hentes opplysninger fra to eksterne kilder: Referansedatabasene, som vi så på i kapittel 6, og språkfunksjonene, som vi skal se på i kapittel 9.

Før vi ser på stilmotorene og stilspesifikasjonene, skal vi se på hvordan mellomkoden bygges opp. Først ser vi på noen generelle trekk og deretter på noen detaljer som blir viktige i kompliserte tilfeller.

7.1 Mellomkode som datastrukturer eller XML

Mellomkoden kan representeres i interne datastrukturer. Den interne datastrukturen kan også tenkes erstattet med XML, slik at det er mulig å skrive mellomkoden til filer og dermed bryte opp prosesseringen i diskrete trinn. Mellomkode XML bør også kunne gjøre gjenbruk av komponenter av bibliografiprosessoren i andre prosjekter enklere.

En annen mulighet er å la mellomkoden virkeliggjøres ved sekvenser av funksjonskall. Hvert funksjonskall signaliserer en hendelse som «referanse begynner», «fet skrift begynner» eller «kursiv slutter». Dette er særlig aktuelt for presentasjonsmellomkoden, ettersom disse funksjonskallene kan drive oversettelsen videre til formaterte siteringer og referanselister. Dette er en mindre minneintensiv løsning enn å holde hele datastrukturen eller XML-dokumenter i minnet, og vi unngår også å instansiere en XML-parser i ut-filteret.

Skulle rekkefølgen av hendelser ikke passe med den rekkefølgen som kreves i presentasjonskoden, må ut-filteret likevel bygge opp interne datastrukturer, og metoden mister sin fordeler.

En interessant observasjon er at mellomkode i XML enkelt kan omgjøres til en hendelsessekvens ved hjelp av en hendelsesbasert XML-parser. Hendelsessekvensen

vil stort sett være ekvivalent med sekvensen i eksempelet ovenfor, da slike parsere genererer hendelser nettopp for begynnelsen og slutten av hvert element, og for hver tekststreng.

Forskjellene mellom disse metodene i faktisk programkode blir derfor små, iallfall hva angår presentasjonsmellomkoden. Dokumentmellomkoden må nok i alle tilfeller representeres i egne datastrukturer, ettersom oversettelsen ikke er like enkel. I det følgende vil jeg derfor anta at mellomkoden er en datastruktur som kan skrives ut som XML dersom dette er ønskelig, og jeg vil bruke en slik XML-form av datastrukturen i eksemplene på mellomkode.

7.2 Dokumentmellomkode

Som vi har sett i kapittel 5, må dokumentmellomkoden inneholde en siteringsliste. I LaTeX-dokumenter for BibTeX inneholder dokumentene makroer på formen `\cite<referanseidentifikator>`, der referanseidentifikatoren knytter sitering til en bestemt referanse i referansedatabasene. Siteringslisten for BibTeX består altså ganske enkelt av **referanseidentifikatorer**. Dette fungerer, ettersom formateringen av siteringene gjøres av LaTeX selv.

For en generell bibliografiprosessor må ut-filteret også kunne knytte en bestemt formatert sitering til en bestemt siteringskommando i dokumentet, og hver sitering i dokumentmellomkoden må derfor ha en identifikator, en **siteringsidentifikator**. Denne må videreføres til de formaterte siteringene i presentasjonsmellomkoden.

Med flere referanselister i ett dokument må hver sitering også knyttes til en bestemt referanseliste. Siteringene må derfor også utstyres med en **referanselisteidentifikator**. Dette åpner strengt tatt for endel nokså ubrukelige siteringsteknikker, som for eksempel veksling i samme tekst mellom siteringer i to referanselister med de samme nøklene, men hva som er intelligent bruk av denne muligheten, får være opp til brukeren å avgjøre.

I tillegg til disse tre identifikatorene kommer den språklige sammenhengen som siteringen forekommer i, siteringsspråket, samt eventuelle for- og ettertekster til siteringen. Hver sitering må altså inneholde

- En siteringsidentifikator.
- En referanseidentifikator.
- En referanselisteidentifikator.
- Et siteringsspråk.
- En fortekst. Kan være tom.

```
<?xml version='1.0' encoding='utf-8'?>
<!-- This file was generated by ibibproc. Do not edit. -->
<idc>
 <reflist id="rl:dictionaries" style="btxplain" lang="en"/>
 <reflist id="rl:main" style="btxplain" lang="en"/>

 <citation id="cit:1" reflist="rl:dictionaries"
   reference="r:dna82" group="g:1" lang="en"/>
 <citation id="cit:2" reflist="rl:main"
   reference="r:hornblower98" group="g:2" lang="en"/>
 <citation id="cit:3" reflist="rl:dictionaries"
   reference="r:dna82" group="g:2" lang="en"/>
</idc>
```

Figur 7.1: Eksempel på dokumentmellomkode. Eksempelet viser mellomkoden for to referanselister i stilen `btxplain`. Legg merke til at det er benyttet prefikser, slik som `rl:`, i de ulike indentifikatorene for å unngå at samme identifikator benyttes flere ganger til ulike formål. Med en slik konstruksjon er det mulig å validere XML-dokumentet med identifikatorattributtene definert som XML-typene `ID` og `IDREF`. Siteringene med identifikatorer `cit:2` og `cit:3` er gruppert sammen.

– En ettertekst. Kan være tom.

Et eksempel på dokumentmellomkode er vist i figur 7.1. Representasjonen av for- og ettertekster, samt gruppering av flere sekvensielle siteringer, kan volde spesielle problemer, som må diskuteres i detalj.

7.2.1 Siteringer med for- og ettertekster

Fortekstene er gjerne korte fritekster av typen «se», «sammenlign med» eller «jevnfør»:

> (sammenlign med Aho 1988 side 88)

Ettertekstene kan også være fritekst, for eksempel (fra [86])

> «Obervers can also easily discriminate differences between actors walking and jogging, actors walking with and without a limp, male and female actors etc. (see Johansson, 1975, for an illuminating discussion of this work).»

men er oftere avgrensninger av den siterte referansen. Disse tilsvarer referanseavgrensningene i referansedatabasene, men forekommer altså for hver sitering og ikke for hver referanse. Analogt med begrepene referanseavgrensninger og komposisjonsavgrensninger bruker jeg betegnelsen **siteringsavgrensninger** om avgrensninger i ettertekster.

La oss anta at ettertekstene kun inneholder siteringsavgrensninger. Et eksempel på en slik ettertekst er vist nedenfor i en forenklet dokumentmellomkode, der kun

referanseidentifikatoren og en ettertekst er med, og med en formatert sitering i en forfatter-dato-stil til høyre:

```
<c id="aho88">88</c>  →  (Aho 1988: 88)
```

Etterteksten er her en sidetallsangivelse gitt ved strengen «88», og det antas at den aktuelle forfatter-dato-stilen genererer resten av den formaterte siteringen, inkludert et kolon og et mellomrom. Et åpenbart problem med en slik fremgangsmåte er at den eksakte presentasjonen av sidetallsangivelsen er like stilavhengig som resten av den formaterte siteringen. En ettertekst der forkortelsen «s.» for «side» forekommer, er bare en annen presentasjon av samme innhold, men ville måtte skrives slik:

```
<c id="aho88">s. 88</c>  →  (Aho 1988: s. 88)
```

Skal dokumentet som inneholder denne siteringen, behandles i et typografisk avansert system som LaTeX, ville vi videre måtte innføre en typografisk instruksjon i etterteksten som hindrer adskillelse av forkortelsen fra sidetallet:

```
<c id="aho88">s. 88</c>  →  (Aho 1988: s. 88)
```

For å unngå slike presentasjonsdetaljer i dokumentmellomkoden, kan vi legge til et eget attributt for sidetall og overlate valg av eventuell forkortelse og av typografiske instruksjoner til siteringsstilen:

```
<c id="aho88" page="88"/>  →  (Aho 1988: s. 88)
```

Men dette fører til at vi må legge til attributter for alle andre tenkbare avgrensninger, og vi havner i samme situasjon som for andre avgrensninger, nemlig at det neppe er mulig å forutse alle mulighetene. Skal for eksempel denne siteringen av tredje akt, tredje scene, linje 298 i Shakespeares *Othello* håndteres slik?

```
<c id="othello" act="3" scene="3" line="298"/>  →  (Othello: 3.3.298)
```

Vi vil stadig kunne komme over tilfeller med siteringsavgrensninger som involverer uvanlige eller uforutsette inndelingsprinsipper:

```
?  →  (Cicerō 63: bok IV, kapittel 9)
```

Når vi i tillegg må ta hensyn til at ettertekster kan inneholde fritekst, er det klart at graden av frihet i for- og ettertekster må veies mot begrensningene som påføres av presentasjonsuavhengighet. Det naturlige kompromisset er å tillate standardtilfellene av siteringsavgrensninger og fritekst i begrensede kombinasjoner ved hjelp av den samme mekanismen som jeg presenterte for referansedatabasene (se avsnitt 6.5).

`\citet{aho88}`	→	Aho et al. (1988)
`\citep{aho88}`	→	(Aho et al., 1988)
`\citet*{aho88}`	→	Aho, Sethi, and Ullman (1988)
`\citep*{aho88}`	→	(Aho, Sethi, and Ullman, 1988)
`\citealt{aho88}`	→	Aho et al. 1988
`\citealp{aho88}`	→	Aho et al., 1988
`\citealt*{aho88}`	→	Aho, Sethi, and Ullman 1988
`\citealp*{aho88}`	→	Aho, Sethi, and Ullman, 1988
`\citeauthor{aho88}`	→	Aho et al.
`\citeauthor*{aho88}`	→	Aho, Sethi, and Ullman
`\citeyear{aho88}`	→	1988
`\citeyearpar{aho88}`	→	(1988)

Figur 7.2: LaTeX-pakken `natbib`. Figuren viser kommandoer og formaterte siteringer i en forfatter-dato-stil.

7.2.2 Grupperte siteringer

I de fleste siteringsstiler vil flere siteringer umiddelbart etter hverandre slås sammen til én sitering. Ettersom dokumentmellomkoden ikke inneholder informasjon om siteringenes relative plassering i teksten, og heller ikke bør gjøre dette, må det innføres en egen konstruksjon som uttrykker gruppering av siteringer. Ettersom hele gruppen formateres til én sitering i presentasjonsmellomkoden, vil ikke lenger siteringsidentifikatorene kunne tjene til sammenbinding av formaterte siteringer og siteringskommandoer i dokumenter. Dette løses ved å innføre enda en identifikator, en **siteringsgruppeidentifikator**, som, istedenfor siteringsidentifikatorene, videreføres til presentasjonsmellomkoden.

```
<c id="aho88" group="1"/>
<c id="knuth85" group="1"/>  →  (Aho 1988, Knuth 1985, Niels 1998)
<c id="niels98" group="1"/>
```

7.2.3 Modifiserte siteringer

LaTeX-pakken `natbib` tilbyr flere modifiserte siteringsformer for forfatter-dato- og nøkkelstiler [14]. Figur 7.2 viser eksempler på kommandoer fra `natbib` og de resulterende formaterte siteringene i en forfatter-dato-stil. Tilsvarende kommandoer tilbys av andre forfatter-dato-pakker som `harvard` og `amsrefs`.

De siteringsformene som har forfatternavnet utenfor parentesene, brukes i disse stilene som setningsledd i den løpende teksten:

> Freud (1900) discusses this. . .

Ved første øyekast ser det ut til å være enkelt å bygge inn støtte for denne utvidelsen, men som vanlig finnes det kompliserende tilfeller, fordi disse siteringene vil måtte tilpasses de syntaktiske forhold i setningen, for eksempel til eierform:

> Freud's (1900) discussion of this. . .

Slik bruk er riktignok sjelden, og mange siteringer vil neppe kunne brukes på denne måten:

> Adorno et al. (1950) discuss this. . .
> *Adorno et al.'s (1950) discussion of this. . .

Det er mulig å anse disse variantformene som utenfor bibliografiprosessorens domene. Legg for eksempel merke til at verbet i eksemplene ovenfor er i entalls- eller flertallsform avhengig av hva siteringen inneholder, noe som likevel betyr at forfatteren må justere teksten etter siteringen. Derfor er det fornuftig at forfatteren selv skriver forfatternavnet inn i den løpende teksten, og eventuelt angir at siteringen skal gjøres i en avgrenset form som bare omfatter årstallet.

På den annen side finnes det eksempler som demonstrerer at dette ikke er er en fullgod beskrivelse av situasjonen, for uttrykket «Aho et al.» i denne siteringen

> Aho et al. (1988) discuss this...

er stilavhengig. En annen stil ville kanskje foreskrive denne varianten:

> Aho, Sethi og Ullman (1988) discuss this...

Hvis vi et øyeblikk forlater teknikalitetene og ser på den litterære siden av de to siste eksemplene, synes det klart at forfatteren her ikke er interessert i hverken Aho, Sethi eller Ullman som forfattere, men deres algoritmebeskrivelser. [1: 16.15] skiller mellom situasjoner der den inkorporerte siteringen brukes til omtale av forfatterne av et verk og situasjoner der siteringen brukes til omtale av et konkret verk. I begge tilfeller vil det ventes at bibliografiprosessoren kan formatere siteringen, men i det første tilfellet vil tidvis manuell inngripen være nødvendig, da bibliografiprosessoren ikke vil kunne avgjøre riktig form.

En annen side av dette er at utvalget av modifiserte siteringer er bundet til stil, og at variantene ikke alltid lar seg overføre fra én stil til en annen. Dokumentforfattere som bruker en nøkkelstil, vil neppe bry seg med å angi de korrekte modifikatorene som kun har betydning i en forfatter-dato-stil.

`natbib` tar faktisk hensyn til disse modifikasjonene også for nøkkelstiler og produserer formaterte siteringer på denne måten:

`\citet{aho88}`	→	Aho et al. [21]
`\citep{aho88}`	→	[21]

Jeg er ikke sikker på om det er lurt å likestille variantformene i nøkkelstiler med variantformene i forfatter-dato-stiler på denne måten, da skrivestilen påvirkes mye av valg av siteringsstil [77]. Hvordan skal for eksempel

> Aho et al. (1988) discuss this...

overføres til en nøkkelstil? En mulighet er

> Aho et al. [21] discuss this...

men vanlig praksis ser ut til å være

> [21] discusses this...

Legg her også merke til entallsformen av verbet.

I praksis burde dette ikke være et problem, da valg av siteringsteknikk oftest er bundet til fagdisiplin, men det finnes disipliner der dette ikke er like absolutt, slik som informatikk, der både nøkkelstiler og forfatter-dato-stiler benyttes.

7.3 Stilprosessen

Vi er nå istand til å beskrive stilprosessen i detalj. La sekvensen $S = (s_1, s_2, \ldots, s_n)$ være siteringene slik de forekommer i dokumentmellomkoden. Hvis vi for enkelhets skyld ser bort fra at hver sitering er assosiert med et siteringsspråk, siteringsmodifikatorer, samt for- og ettertekster, vil hver sitering s_i være en ordnet kvadruppel $s_i = (c_i, g_i, l_i, r_i, m_i)$ der $1 \leq i \leq n$, c_i er siteringsidentifikatoren, g_i er siteringsgruppeidentifikatoren, l_i er referanselisteidentifikatoren, r_i er referanseidentifikatoren og m_i er en merkelapp som genererers under prosesseringen.

Prosesseringen foregår i disse trinnene:

1. Alle referanser med referanseidentifikator $r \in \bigcup_{i=1}^{n} r_i$ leses inn fra de angitte referansedatabasene.[1]

2. For hver referanseliste $l \in \bigcup_{i=1}^{n} l_i$ utføres følgende:

   (a) Konstruer sekvensen T som består alle siteringer $s \in S$ med listeidentifikator l og den innbyrdes rekkefølgen til siteringene bevart.

   (b) Konstruer sekvensen R ved å plukke ut enhver sitering $s \in T$ som har en referanseidentifikator som ikke allerede forekommer i utplukket, og deretter utføre en stabil sortering av utplukket ved hjelp av informasjonen i de innleste referansene.

   (c) Ved hjelp av de innleste referansene genereres en merkelapp m som settes i alle siteringer $s \in R$.

   (d) Siteringer $s \in T$ som ikke har merkelapper, vil ha referanseidentifikatorer som forekommer i en av siteringene $s' \in R$. Merkelappene til siteringene $s \in T$ settes til merkelappen fra den siteringen $s' \in R$ med samme referanseidentifikator.

[1] Her må man også ta hensyn til at ukjente referanser kan være refererte i siteringene. Disse må isåfall markeres på en hensiktsmessig måte og fjernes fra siteringslisten.

(e) Formateringsfunksjonen for referanser kalles for alle $s \in R$.

3. For alle siteringsgrupper $g \in \bigcup_{i=1}^{n} g_i$ kalles en formateringsfunksjon for siteringsgrupper. Denne funksjonen skal igjen kalle en formateringsfunksjon for siteringer for alle siteringer i siteringsgruppen.

7.4 Presentasjonsmellomkode

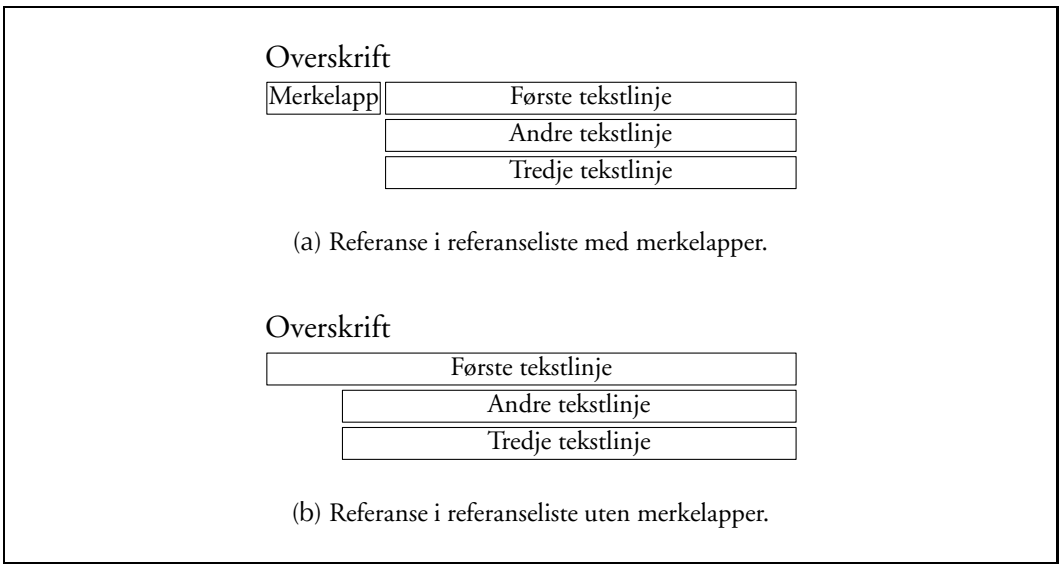
Presentasjonsmellomspråket fungerer som kommunikasjonsspråk mellom stilmotor og ut-filter. Det må inneholde både referanselister og formaterte siteringer, og disse vil, i tillegg til teksten i dem, bestå av typografiske instruksjoner som uthevelse, horisontalt og vertikalt mellomrom og lignende.

For presentasjonsmellomkoden blir forholdet mellom typografiske instruksjoner og Unicode igjen et tema. I tillegg til Unicode-kodepunktene for typografiske tegn som vi så på i avsnitt 4.7, kunne man tenke seg å benytte de private områdene i Unicode-koderommet til slike typografiske instruksjoner. Dette er en mulig løsning, da mellomspråket er et internt dokumentformat, og de private kodene ikke vil eksponeres for sluttbrukere. Hver instruksjon av typen «fet skrift begynner» vil kunne representeres med ett Unicode-tegn med de åpenbare ressursbruksfordeler det medfører sammenlignet med XML-elementer og -attributter. Erstatningen av tegnene i ut-filtrene med kommandoer for dokumentprosserings- eller typesettingssystemet kan også gjøres ved hjelp av enkle regulære uttrykk istedenfor en XML-parser. Skulle private koder forekomme i referansedatabasene eller i siteringene, oppstår det en konflikt, men dette er lite sannsynlig, da blokkene som er avsatt til private koder, er store, og vårt behov for kodepunkter for de typografiske kodene er lite.

Et mulig problem er at det kan være behov for instruksjoner som tar argumenter. Dette kunne for eksempel være instruksjoner som margjustering eller angivelse av horisontalt mellomrom, som er kontinuerlige størrelser. Det viser seg heldigvis at bibliografiprosessoren kan overlate disse oppgavene til dokumentprosesserings- eller typesettingssystemet, og så lenge vi er villige til å avsette kodepunkter til for eksempel koder for alle støttede språk, skulle det ikke være noe i veien for å benytte private Unicode-tegn til typografiske instruksjoner.

En referanseliste består av flere deler (se figur 7.3). Vi trenger derfor instruksjoner som indikerer begynnelsen på og avslutningen av disse delene, det vil si en referanseliste, en enkelt referanse, en eventuell merkelapp i referansen, selve teksten i referansen og de ulike blokkene og områdene som referansen består av. Både blokker og områder brukes til å markere områdespråk; forskjellen er at det er vanlig å sette inn litt ekstra horisontalt mellomrom mellom blokkene, mens områdene settes helt inntil hverandre.

For formaterte siteringer er bildet enklere, og vi behøver kun en instruksjon for



Figur 7.3: Delene av en referanseliste. Referanselistene med merkelapper brukes i nøkkelstilene, der nøkkelen, ofte omsluttet av klammer eller lignende, plasseres i merkelappen. Andre stiler benytter den merkelappfrie formen. Tekstlinjene består igjen av blokker og områder.

start og slutt på siteringen og for områdene i siteringen.

I merkelapper og i tekstene i referansene og siteringene trenger vi instruksjoner for fet skrift, kursivert skrift, kapiteler, hevet og senket skrift, samt preformaterte data. Figur 7.4 viser et eksempel på presentasjonsmellomkode. Her er identifikatorene fra dokumentmellomkoden overført til referanselistene, referansene og de formaterte siteringene, og språkattributtene er beregnet. For å få korrekte språkinnstillinger for de enkelte delene av hver referanse, er elementet `div` innført hver gang en del av en referanse er på et annet språk enn referansespråket.

7.5 Stilfunksjoner

Tidligere så vi at den systemuavhengige prosesseringen kan beskrives i tre separate trinn, og i denne beskrivelsen ble fem stilavhengige operasjoner referert:

- En sorteringsfunksjon. Funksjonen sorterer referanseidentifikatorer ved hjelp av opplysningene fra referansedatabasene.
- En formateringsfunksjon for merkelapper.
- En formateringsfunksjon for referanser.
- En formateringsfunksjon for siteringsgrupper.

```
<?xml version='1.0' encoding='utf-8'?>
<ipc>
 <reflist entries="1" id="dictionaries" lang="no">
  <reference key="dna82" lang="no">
   <text>
    <block>Guttu, T., K. Skadberg og I. Wettergren-Jensen.</block>
    <block>1982.</block>
    <block><i>Riksmålsordboken</i>.</block>
    <block>Kunnskapsforlaget</block>
   </text>
  </reference>
 </reflist>

 <reflist entries="1" id="main" lang="no">
  <reference key="hornblower98" lang="no">
   <text>
    <block><div lang="en">Hornblower, S.</div> og
      <div lang="en">A. Spawforth.</div></block>
    <block>1998.</block>
    <block><i><div lang="en">The Oxford Companion to Classical
      Civilization.</div></i></block>
    <block><div lang="en">Oxford</div>:
      <div lang="en">Oxford University Press</div></pubname>
   </text>
  </reference>
 </reflist>

 <group id="g:1" lang="no">(Guttu 1982)</group>
 <group id="g:2" lang="no">(Hornblower 1998, Guttu 1982)</group>
</ipc>
```

Figur 7.4: Eksempel på presentasjonsmellomkode. Koden tilsvarer dokumentmellomkoden fra figur 7.1. Den inneholder to referanselister, en med referanslisteidentifikator `dictionaries` og en med identifikator `main`. Listene er satt i en norsk tekst og med norske nøkkelord uavhengig av de referete verkers språk. Av denne grunn inneholder presentasjonskoden for det engelske verket vekslinger til engelsk for alt innhold tatt fra referansen. `block` omslutter blokker, og `div` omslutter områder. (Ekstra linjeskift er satt inn i XML-koden for lesbarhet.)

– En formateringsfunksjon for siteringer.

De fire sistnevnte funksjonene kan utformes etter flere filosofier. Den tradisjonelle løsningen, for eksempel benyttet i BibTeX, er å ha separate formateringsfunksjoner for hver referansetype. Disse funksjonene plukker ut relevant informasjon fra datastrukturene og formaterer dem. Motivasjonen for en slik fremgangsmåte er at formateringskravene er svært ulike fra referansetype til referansetype.

Et alternativ til dette er å benytte mønstre. Mønstrene kan lages etter samme tankegang som i stilspråket XSLT, der hvert mønster passer for en bestemt mengde av feltene. Slike regler er godt egnet til å stilbehandle referanser representert som trær, og det er enkelt å spesifisere standardmønstre for generelle tilfeller. Da slike mønstre er statiske, ville de kunne plasseres i stilspesifikasjonen, og stilmotoren vil stort sett kun omfatte sorteringsfunksjonen.

Kombinasjoner av referansetypefunksjoner og mønstre er mulig, men kun ved spesielle tilpasninger i bruken av mønstrene [43].

7.6 Stilmotorer og stilspesifikasjoner

Forholdet mellom stilmotorer og stilspesifikasjonene reflekterer balansen mellom programmerbarhet og enkelhet. På den ene siden ønsker vi stor frihet i utformingen av stilene – det er ikke mulig å forutse alle bibliografistiler som en bruker kan ønske. På den annen side er de aller fleste bibliografistiler temmelig like. Vi har allerede sett at de kan klassifiseres i fem hovedgrupper, og selv om variasjonen innen hver gruppe er stor, ligger ulikhetene først og fremst i detaljer. Det vil derfor kunne være heldig å ha et felles stilprogram som kan modifiseres ved hjelp av parametre.

Stilmotorene utfører oversettelsen mellom dokument- og presentasjonsmellomkode. Slik som vi klassifiserte stilene i kapittel 2, fulgte inndelingen først og fremst de ulike kravene til sortering og nøkkelgenerering, og det er også disse forholdene som gir de største forskjellene i programkode. Klassifikasjonen i fem bibliografistiler er derfor godt egnet som grunnlag for oppdelingen av stilmotorer.

Stilspesifikasjonene angir den statiske delen av stilvalget. Tanken er at de skal velge stilmotor, angi parametre til den valgte stilmotoren og inneholde metainformasjon om stilen, slik som stilens navn, lister over hvilke publikasjoner eller forlag som benytter den, og lignende.

I likhet med tidligere valg faller det naturlig å velge XML for stilspesifikasjonene, og i tradisjonen til BibTeX kan det være naturlig med en stilspesifikasjonsfil for hver stil. Figur 7.5 viser stilspesifikasjonen som er brukt i denne oppgaven.

```
<?xml version='1.0' encoding='utf-8'?>
<!DOCTYPE style PUBLIC
  "-//Marius Johndal//DTD ibibproc style V1.0//EN"
  "http://ibibproc.sourceforge.net/1.0/ibibproc-style.dtd">

<style>
  <name>sccons</name>
  <summary>Custom small caps numeric key style.</summary>
  <author>Marius L. Jøhndal</author>
  <version>1.0</version>
  <engine>number</engine>
  <engine-version>0.1</engine-version>
  <property name="do_urls">1</property>
  <property name="authors_all_sc">1</property>
  <property name="authors_first_invert">1</property>
</style>
```

Figur 7.5: Eksempel på stilspesifikasjon. Spesifikasjonen er laget for denne oppgaven og er basert på stilmotoren for nøkkelstiler med referanser ordnet etter forfatternavn. Når parametrene ikke angir noe annet, vil denne stilmotoren bruke numre i nøklene, sette disse i klammer i merkelappene og sette siteringene i klammer. Stilmotorparametrene oppgitt i denne spesifikasjonen instruerer stilmotoren om å sette inn URL'er, å sette forfatternavn i kapiteler og å oppgi det første forfaternavnet på invertert form.

Kapittel 8

Filtre

I dette kapitlet diskuteres interaksjonen mellom filtre og dokumentprosesseringseller typesettingsystemer. For hvert filter må vi se på hvordan siteringslisten skal overføres fra den omkringliggende programvaren til bibliografiprosessoren, og på hvilken måte bibliografiprosessoren skal produsere referanselistene og de formaterte siteringene. Denne kommunikasjonen foregår oftest ved hjelp av filer som leses og skrives på ulike trinn i produksjonen av det endelige dokumentet, og det er særlig viktig å undersøke hvordan typografiske instruksjoner uttrykkes, og hvilke tegnsett som benyttes i disse filene.

Jeg har valgt ut to filtre som belyser problemstillingene med filtrene: TEX-filteret og DocBook-filteret. TEX-filteret vil måtte utføre oversettelse fra en Unicode-tegnkoding til tegnkodinger og kommandoer som TEX kan forstå, og det må ta hensyn til at TEX finnes i en lang rekke varianter og konfigurasjoner. I DocBook-filteret er utfordringen først og fremst hvordan man skal forholde seg til DocBooks presentasjonsuavhengighet og hvordan interaksjonen mellom bibliografiprosessoren og vektøy som benyttes for prosessering av DocBook, skal være.

8.1 TEX-filteret

TEX-filteret skal ta seg av kommunikasjon med typesettingssystemet TEX. Det bør ta hensyn til at det finnes et stort utvalg tillegg i form av ferdige makropakker og samlinger av fonter som enkelte TEX-distribusjoner inkluderer og andre ikke, og mange av disse er nødvendige for å kunne bruke ulike deler av Unicode-repertoaret.

De to mest utbredte **formatene** for TEX er plain TEX og LATEX. Førstnevnte er det «opprinnelige» formatet laget av Donald Knuth og omfatter en grunnleggende gruppe makroer for typesetting. LATEX er i større grad et helt dokumentprosesseringssystem som lar brukeren benytte kommandoer for beskrivelse av struktur fremfor presentasjon. På mange måter er plain TEX en delmengde av LATEX, men forholdet

er ikke alltid like enkelt. Andre formater er oftest variasjoner av plain TEX og LATEX tilpasset bestemte språk eller spesielle dokumenttyper.

Et naturlig utgangspunkt er at TEX-filteret skal kunne brukes med dokumenter skrevet for plain TEX og LATEX uten at det bindes sterkt til noen av disse formatene. Med en bevisst bruk av kommandoer som finnes både i plain TEX og i LATEX, burde det være mulig å lage ett filter for begge formatene.

8.1.1 Filer for siterings- og referanselister

Det vil være naturlig å benytte de samme filene som BIBTEX gjør til inn- og utdata. For siteringer kan bibliografiprosessoren benytte `aux`-filen og generelle TEX-makroer som etterligner dem som finnes i LATEX for BIBTEX-bibliografier. Siteringsmakroene må utvides for å ta hensyn til de mer avanserte siteringsformene, men prinsippet forblir det samme. Ettersom det er mulig å ha flere referanselister i ett dokument, må bibliografiprosessoren enten produsere én fil med alle referanselistene plassert inne i makroer, flere filer for ren inklusjon i dokumentet eller benytte en mekanisme som velger ut en bestemt referanseliste fra én inkludert fil. Ettersom TEX benytter statisk minneallokering, er makroløsningen ikke særlig aktuell. I tillegg skal bibliografiprosessoren, i motsetning til BIBTEX, produsere de ferdig formaterte siteringene, og disse kan med fordel skrives til samme fil som de formaterte referanselistene. Å skrive alle referanselistene til samme fil, og benytte en `bbl`-fil, slik BIBTEX gjør, synes som den mest ryddige løsningen.

En heldig sideeffekt ved å produsere ferdig formaterte siteringer er at den tredje kjøringen av LATEX blir overflødig, idet såvel referanselister som siteringer finnes i endelig form ved andre LATEX-invokasjon.

8.1.2 TEX og tegnsett

TEX er et svært portabelt system og er blant annet ikke bundet til noe bestemt tegnsett – TEX vil uansett oversette data til ASCII for bruk internt [54: 8]. Forbindelsen mellom tegn i et dokument og tegnene i det ferdig behandlede dokumentet er heller ikke innebygget i TEX, men avgjøres av tabeller med avbildninger mellom tegn i dokumenter og tegn i innleste fonter. TEX er istand til å bruke fonter med opptil 256 tegn, men de tradisjonelle TEX-fontene, Computer Modern, er av historiske grunner 7-bit-fonter. I plain TEX er Computer Modern-fontene innleste, og tegntabeller satt opp for ASCII-kodede dokumenter. Enkelte bokstavtegn som ikke finnes i ASCII-repertoaret, kan representeres med makroer (se tabell 8.1), og det kan lages komposisjoner av bokstaver og de vanligste diakritika (se tabell 8.2). Denne løsningen er ikke optimal, da det fører til problemer med orddelingsmekanismen, men det er en effektiv metode som omgår problemene knyttet til et større tegnsett.

Skal andre tegn brukes i et dokument, må egnede fonter lastes inn og tegntabeller

æ	\ae	Æ	\AE	ø	\o	Ø	\O
œ	\oe	Œ	\OE	å	\aa	Å	\AA
ł	\l	Ł	\L	ß	\ss		

Tabell 8.1: Spesialtegnmakroer i plain TEX.

ó	\'o	ò	\`o	ô	\^o	õ	\~o	ō	\=o
ȯ	\.o	ö	\"o	ç	\c c	ǒ	\u o	ǒ	\v o
ő	\H o	ọ	\d o	o̲	\b o	o͡o	\t oo		

Tabell 8.2: Diakritikamakroer i plain TEX.

bygges opp. Dette er en ganske omfattende oppgave, som mest hensiktsmessig utføres ved hjelp av ferdige makropakker – eventuelt kan TEX-formatet modifiseres til å benytte andre fonter og andre tegnkodinger. I prinsippet er det altså mulig å lage varianter av TEX som fungerer med de fleste tegnkodinger og med tegnrepertoarer utover ASCII. I praksis er nok likevel plain TEX mest brukt med ASCII-kodede dokumenter, og bibliografiprosessoren bør derfor tilpasses dette. Det vil si at den må emittere ren ASCII-kode, at den bare kan benytte de makroene som plain TEX tilbyr, og at den ikke kan gjøre videre spesielle tilpasninger eller forutsette bruk av andre makropakker.

8.1.3 LATEX og tegnsett

De generelle betraktningene om TEX i forrige avsnitt er også gyldige for LATEX. Spesielt er makroene fra plain TEX for spesialtegn og diakritika også gyldige i LATEX. I tillegg til disse er det lagt til en lang rekke andre diakritika- og spesialtegnmakroer i ulike LATEX-versjoner (se tabell A.5, tabell A.6 og tabell A.8 for oversikter over disse). Det finnes også generelle mekanismer for valg av dokumentkoding og fontkodinger ved hjelp av pakkene `inputenc` og `fontenc`, som begge er del av alle LATEX-distribusjoner. Begge disse pakkene anpasser tegnmakrodefinisjonene til de aktuelle dokumentkodingene og de tilgjengelige fontene (disse tilpasningene gjelder da først og fremst de «lange» tegnmakroene av typen `\textonequarter`). Denne muligheten utnyttes også av andre sentrale tilleggspakker, som for eksempel pakken `textcomp`. Denne utfyller de T1-kodede standardfontene med endel symboler som mangler eller finnes i ulike varianter.

8.1.4 Ligaturer

I TEXs forstand er ligaturer tegnsekvenser i dokumentet som skal slås sammen til ett symbol. Dette gjelder for eksempel både sekvensen `--`, som får TEX til å produsere en intervallstrek, og sekvensen `fi`, som gir den estetiske ligaturen «fi». Alle ligaturene er definert i fontfilene, så ligaturene brukes uavhengig av om LATEX eller plain TEX be-

nyttes. Til forskjell fra sekvenser som gir de estetiske ligaturene, er det ikke meningen at en sekvens som `--` skal oppfattes som en ligatur når disse forekommer i referansedatabaser, og disse må derfor håndteres spesielt. Man kan hindre at sekvensene tolkes som ligaturer ved å sette inn `{}` mellom de tegnene som skal skilles. Disse tegnsekvensene, som av [18] kalles «*ligatures of convenience*», burde etter min mening egentlig ikke ha vært ligaturer, ettersom de skaper store problemer og vel kun er laget for å unngå manglene ved ASCII-kodingen og ulempene ved å skrive lange kommandoer. En liste over ligaturer som må behandles på denne måten finnes i tabell A.4.

8.1.5 `babel` og aktive tegn

Pakken `babel`, som skal ta seg av enkelte språkavhengige tilpasninger og distribueres sammen med LaTeX, kan definere aktive tegn, det vil si tegn som stort sett oppfører seg som kontrollsekvenser. En del av disse aktive tegnene bruker `babel` for å få til korrekt orddeling med spesielle tegn og for korrekt behandling av estetiske ligaturer, for eksempel i portugisisk, der estetiske ligaturer aldri skal brukes [34]. Andre aktive tegn har sideeffekter som ikke er ønskelige, og disse må også håndteres spesielt av filteret. Dette gjøres enklest ved å midlertidig redefinere dem som vanlige tegn når filteret emitterer tekst.

8.1.6 Pakken `ucs`

Pakken `ucs` gjør det mulig å bruke LaTeX-dokumenter kodet i UTF-8 og omfatter også støtte for kombinasjonstegn [76]. Pakken gjør dette ved hjelp av en rekke tabeller som oversetter kodepunkter til LaTeX-kommandoer. Kommandoene kan være spesialtegnmakroer, diakritikamakroer eller kommandoer som er definert i andre sentrale pakker for bestemte tegnrepertoarer. Figur 8.1 viser noen slike tegndefinisjoner, blant annet for tegnet ħ (`U+0127 LATIN SMALL LETTER H WITH STROKE`), som er et symbol i IPA.[1] Dette symbolet kan lages ved hjelp av kommandoen `\textcrh` definert i makropakken `tipa`. Pakken `tipa` må da leses inn, og `ucs`-pakken vil bruke makroen når UTF-8-sekvensen for tegnet ħ forekommer i dokumentet.

`ucs`-pakken kan også brukes sammen med andre kodinger, idet enkelttegn i Unicode-repertoaret er tilgjengelige med kommandoen `\unichar{`*`<kodepunkt>`*`}`. På denne måten er det mulig å produsere dokumenter i for eksempel ren ASCII-kode og likevel håndtere alle tegn i Unicode-repertoaret. En ulempe med en slik fremgangsmåte er at teksten ikke lenger er lesbar, og dokumentene kan bli temmelig store.

[1] **International Phonetic Alphabet** er et internasjonalt fonetisk transkripsjonssystem fastsatt av International Phonetic Association. Det skal kunne brukes til å beskrive alle foner i alle verdens språk. Tegnene er basert på romersk-latinsk skrift modifisert med en lang rekke diakritika [22].

```
\uc@dclc{292}{default}{\^H}%
\uc@dclc{293}{default}{\^h}%
\uc@dclc{295}{tipa}{\textcrh}%
\uc@dclc{296}{default}{\~I}%
\uc@dclc{297}{default}{\~\i}%
```

Figur 8.1: Eksempel på tegndefinisjoner i `ucs`-pakken. Utdraget er fra filen `uni-1.def`, som omfatter tegndefinisjoner i intervallet `U+0100` til `U+01FF`.

8.1.7 TEX og spesialtegn

Tegnene \, {, }, $, &, #, ^, _, % og ~ har spesielle funksjoner i plain TEX, og de må derfor håndteres spesielt. Noen av disse tegnene kan uttrykkes ved å prefigere dem med tegnet \ [54: tillegg B]. Blant disse er tegnene er \{ og \} kun tilgjengelige i matematikkmodus. For tegnene \, ~ og ^ har også kombinasjonen med tegnet \ spesiell funksjon, og de må derfor uttrykkes på andre måter. Tegnet \ kan skrives som `\backslash` i matematikkmodus, mens ~ og ^ må emuleres som `\~{}` og `\^{}` eller plukkes ut fra den aktuelle fonten med `\char`-makroen.

For LATEX er bildet tilsvarende, men det finnes flere og bedre representasjonsalternativer for spesialtegnene på grunn av det rikere utvalget av tegnmakroer.

8.1.8 Ω og Λ

Ω er en utvidelse av TEX som er istand til å lese dokumenter kodet i en Unicode-koding eller en annen koding som kan omformes til Unicode. Ω bruker Unicode som intern tegnrepresentasjon, men fungerer ellers på samme måte som TEX [23]. Det finnes også et LATEX-format basert på Ω, kalt Λ. Per dags dato er Ω og Λ ikke særlig mye brukt, og prosjektets status er noe uklar – med tanke på både videreutvikling og aksept blant brukere av TEX. Noen mener for eksempel at prosjektet trenger en annen utviklingsmodell for å lykkes [65]. Det virker derfor lite rimelig å prioritere Ω i TEX-filteret, men det skulle være trivielt å legge til støtte for dette senere, da Ω kan håndtere UTF-8-kodet tekst ved hjelp av sine inndatafiltre.

8.1.9 En enhetlig representasjon av UCS i TEX

Med så mange forskjellige formater, varianter, makropakker, fonter og kodinger er det umulig å støtte alle mulige kombinasjoner. Det virker derfor mest fornuftig å støtte de konvensjonelle kombinasjonene av formater, pakker og kodinger. For plain TEX er dette enkelt: Disse dokumentene er vanligvis ASCII-kodede, og det er et begrenset antall spesialtegn og diakritika. TEX-filteret må produsere ASCII-kodet tekst og slå opp andre tegn i tabeller som oversetter til makroer for spesialtegn og diakritika.

For LATEX er bildet mer komplisert, da endringer og tillegg er gjort gradvis ved

utgivelsen av nye versjoner. Det burde dog være rimelig å kreve en moderne LaTeX-versjon (LaTeX 2$_\varepsilon$ 1994/12/01 eller nyere), da det ikke skal være nødvendig å holde nye pakker ved like for eldre LaTeX-versjoner [57: 2.1]. Med en såpass ny versjon kan vi blant annet anta at T1-kodede fonter er tilgjengelige, og at alle de «lange» tegnmakroene finnes. Pakken `inputenc` har innebygget støtte først og fremst for ASCII-lignende kodinger som US-ASCII og ISO-8859-kodinger, og den samme fremgangsmåten som for plain TeX-dokumenter (altså produksjon av ASCII-kodet tekst og oppslag i tabeller for alle andre tegn) skal da også fungere her. Oversettelsestabellene bør skille mellom plain TeX og LaTeX slik at alle tegnene som er lagt til i de senere LaTeX-versjonene, også kan oversettes.

Å bygge inn støtte for andre pakker, som for eksempel pakken `textcomp`, virker unødvendig, ettersom pakken `ucs` tar seg av nettopp denne formen for oversettelser. Dersom LaTeX-miljøet er avansert nok til at man kan benytte andre tegnmakroer enn dem som LaTeX-kjernen definerer, er det med andre ord rimelig å kreve at også `ucs`-pakken er tilgjengelig. Dette gir oss en tredje kombinasjon som enkelt kan dekkes, nemlig en moderne LaTeX-versjon med pakken `ucs` tilgjengelig.

For å holde TeX-filteret så enkelt og generelt som mulig, og for å unngå at fremtidige tillegg bryter bakoverkompabilitet, er det en fordel at disse tre kodingsformene behandles ved hjelp av den samme generelle mekanismen. Dette kan enkelt oppnås ved at filteret for alle tegn utenfor ASCII-koderommet emitterer en makro med kodepunktet som argument, og at en makropakke som inkluderes av dokumenter, definerer denne makroen. Makroen utfører oversettelsen i en av tre modi:

a) Oversettelse til plain TeX-kommandoer ved hjelp av egne tabeller. Dette dekker ASCII og de spesialtegn og diakritikakomposisjoner som er mulige i plain TeX.

b) Oversettelse til LaTeX-kommandoer ved hjelp av egne tabeller. Dette dekker ASCII og de spesialtegn og diakritikakomposisjoner som er mulige i moderne LaTeX-versjoner.

c) Oversettelse til `\unichar{<kodepunkt>}` for LaTeX med pakken `ucs`. Dette dekker store deler av Unicode-repertoaret, og dekningen vil bli bedre etterhvert som `ucs`-pakken videreutvikles.

For alle modi må tegnene innenfor ASCII-koderommet filtreres for spesialtegn, aktive tegn og ligaturer.

Tabellene i de to første oversettelsestilfellene kan fungere etter samme prinsipp som tabellene i `ucs`-pakken. Tabellene kan lages på grunnlag av beskrivelsene av hvert tegn i UCD,[2] ved hjelp av tabellene som er distribuert i `ucs`-pakken og makrodefinisjonene for spesielle tegn i LaTeX-distribusjonen. Tabellene A.1 til A.3 viser

[2] **Unicode Character Database** (UCS) er en samling med offentlig tilgjengelige datafiler som inneholder beskrivelser av alle Unicode-tegn og deres egenskaper.

tegnkart for bibliografiprosessoren generert på denne måten under de diskuterte forutsetningene om versjoner og tilgjengelige fonter.

8.1.10 Referanselistemiljø

For formateringen av referanselister er det nyttig å se på LaTeX-makroene for bibliografier. Referanse plasseres i et eget miljø med navnet `thebibliography`. Miljøet er basert på det generelle listemiljøet, men med modifiserte kommandoer for hvert punkt i listen. For plain TeX emuleres dette listemiljøet ved hjelp av kode tatt fra LaTeX-makroene.

TeX-filteret vil ha behov for det samme miljøet, men i to varianter, en for referanselister med merkelapper og en for referanselister uten merkelapper.

En vanskelighet med `thebibliography`-miljøet i LaTeX er at overskriften er avhengig av dokumentklassen. For dokumentklassen `article` skal referanselisten være et avsnitt med den lokaliserte formen av «Referanser» som overskrift, mens for andre klasser er referanselisten et kapittel med overskriften «Bibliografi» [55, 7]. I plain TeX finnes ikke dette skillet, og koden som emulerer `thebibliography`-miljøet kan heller ikke inkludere noen overskrift. Den enkleste løsningen er naturligvis å utelate overskriften fra referanseliste-miljøet, og bare produsere en overskrift dersom makroene lastes inn med en egen opsjon satt.

8.2 DocBook-filter

Takket være den nære koblingen mellom Unicode og XML, vil tegnsett ikke være et problemområde for DocBook-filteret. Kun de fem spesialtegnene i XML må særbehandles og oversettes til entiteter.[3] DocBooks presentasjonsuavhengighet er derimot en utfordring, idet bibliografiprosessorens oppgave nettopp er å produsere en presentasjonsutgave av siteringer og referanser.

8.2.1 Inklusjon av referanselister

En generert referanseliste kan enkelt leses inn i et dokument ved hjelp av entiteter:

```
<!DOCTYPE book PUBLIC
  "-//OASIS//DTD DocBook XML V4.1.2//EN"
  "http://www.oasis-open.org/docbook/xml/4.1.2/docbookx.dtd" [
<!ENTITY bibliography "mybibliography.xml">
]>
```

[3] I praksis er det kun nødvendig å ta hensyn til `<` og `&`, ettersom disse er de eneste som vil kunne forvirre parsere.

```
...

&bibliography;

...
```

Problemet er at det her forventes en bibliografi på enten *«raw»* eller *«cooked»* form, og ikke en ferdig formatert referanseliste. DocBook-filteret befinner seg med andre ord på samme trinn i produksjonsprosessen som stilarkene. Dersom vi ønsker at referansene skal kunne produseres i ett trinn adskilt fra resten av dokumentprosesseringen, det vil adskilt fra XSLT-stilarkenes konvertering av dokumentet til et eller annet målformat, må vi introdusere en form for presentasjonsvokabular i DocBook.

8.2.2 Tilpasning av DocBook

DocBook er laget for å kunne tilpasses spesielle behov – særlig med tanke på enkle tillegg til vokabularet i form av nye attributter og nye elementer, eller begrensning av vokabularet. Dette oppnås ved en svært stor grad av parametrisering av DTD'en, for eksempel ser parameterentiteten `tech.char.class`, som angir elementer som blant annet kan brukes i avsnitt under elementet `para`, ser slik ut:

```
<!ENTITY % local.tech.char.class "">
<!ENTITY % tech.char.class
  "action|application
   |classname|methodname|interfacename|exceptionname
   |ooclass|oointerface|ooexception
   |command|computeroutput|database|email|envar|errorcode|errorname
   |errortype|errortext|filename
   |function|guibutton|guiicon|guilabel|guimenu|guimenuitem
   |guisubmenu|hardware|interface|keycap
   |keycode|keycombo|keysym|literal|constant|markup|medialabel
   |menuchoice|mousebutton|option|optional|parameter
   |prompt|property|replaceable|returnvalue|sgmltag|structfield
   |structname|symbol|systemitem|token|type|userinput|varname
   %ebnf.inline.hook;
   %local.tech.char.class;">
```

En viktig egenskap ved DTD'er er at entitetsdeklarasjoner kan gjentas, og ved slike gjentagelser gjelder den første deklarasjonen. Slik er det mulig å overstyre standarddeklarasjonen av parameterentiteten `local.tech.char.class` og derigjennom for eksempel legge et nytt element `argument` til `tech.char.class`:

```
<!ENTITY % local.tech.char.class "|argument">

<!ENTITY % docbook PUBLIC
  "-//OASIS//DTD DocBook XML V4.1.2//EN"
```

```
  "http://www.oasis-open.org/docbook/xml/4.1.2/docbookx.dtd">

%docbook;

<!ENTITY % local.argument.attrib "">
<!ENTITY % argument.role.attrib "%role.attrib;">
<!ELEMENT argument %ho (%smallcptr.char.mix;)*>
<!ATTLIST argument
  %common.attrib;
  %argument.role.attrib;
  %local.argument.attrib;>
```

En slik modifikasjon av DTD'en til en ny utvidet eller begrenset DTD kalles i DocBook-terminologi for et **tilpasningslag** (engelsk *customisation layer*) og er den anbefalte metoden for modifikasjon av DTD'en. Som man kan se av eksempelet ovenfor, må DTD'ens *public identifier* eksplisitt nevnes i tilpasningslaget, og derfor blir det nødvendig å vedlikeholde flere utgaver av samme tilpasningslag for ulike aktuelle versjoner av DTD'en. Dette er en liten ulempe, men det sikrer iallfall at modifikasjonene som gjøres, fungerer med bestemte DTD-versjoner. Skal bibliografiprosessoren benyttes for et dokument, må dette dessuten benytte tilpasningslaget som DTD istedenfor DTD'en til DocBook XML – noe mange brukere ikke vil ønske å gjøre.

Tilpasningslag er en enkel løsning og gir forholdsvis gode resultater. Programmet *RefDB* har en annen tilnærmingsmåte, idet det genererer egne XSLT-stilark for de ulike målformatene fra en stilspesifikasjon. De genererte stilarkene formatererer DocBooks egne semantiske elementer uten noe *RefDB*-spesifikt vokabular. Med denne løsningenen unngår man en lang rekke av ulempene knyttet til tilpasningslag, siden den ikke forutsetter noen modifikasjon av DTD'en. Likevel vil dette være en uheldig løsning for oss, da vi ønsker at bibliografiprosessoren selv skal foreta all formatering. Å generere XSLT-stilarkene er også temmelig komplisert og lite fleksibelt.

Det er også mulig å tenke seg bruk av for eksempel DocBooks `markup`-element eller XMLs prosesseringsinstruksjoner for formateringsinstruksjonene. Begge disse løsningene umuliggjør validering av dokumenter og skaper vanskeligheter for videre prosessering.

Den antagelig mest fornuftige løsningen er altså å utvide DocBook med et eget stilvokabular ved hjelp av et tilpasningslag. Dette kan igjen gjøres på en lang rekke måter. Stilelementer, da gjerne i et eget navnerom, kan introduseres for bruk innenfor de enkelte semantiske elementene:

```
<bibliomixed>
  <title>
    <ibibproc:i>Foobar</ibibproc:i>
  </title>
</bibliomixed>
```

For å gjøre sammenkoblingen mellom semantiske elementer og stil klarere, kan stilinformasjonen legges til som attributter til de semantiske elementene:

```
<bibliomixed>
  <title style="i">Foobar</title>
</bibliomixed>
```

Det er også mulig å se helt bort fra DocBooks bibliografimodell og definere stilelementer – igjen helst i et eget navnerom – som gyldige elementer alene i en referanseliste:

```
<bibliomixed>
  <ibibproc:i>Foobar</ibibproc:i>
</bibliomixed>
```

Et tilpasningslag med denne siste formen for vokabularutvidelse kan i sin helhet se for eksempel slik ut:

```
<!ENTITY % local.bibliocomponent.mix "|ibibproc:entry">

<!ENTITY % docbook PUBLIC "-//OASIS//DTD DocBook XML V4.1.2//EN"
  "http://www.oasis-open.org/docbook/xml/4.1.2/docbookx.dtd">
%docbook;

<!ENTITY % ibibproc.common.attrib
  "xmlns:ibibproc CDATA #FIXED
    'http://ibibproc.sourceforge.net/xmlns/ibibproc-isl'">

<!ELEMENT ibibproc:entry (#PCDATA|ibibproc:i|
                          ibibproc:b|ibibproc:url)*>
<!ATTLIST ibibproc:entry %ibibproc.common.attrib;>
```

De to første forslagene til vokabularutvidelse har den fordelen at de bevarer de semantiske elementene i de formaterte referanselistene. Dette betyr at semantiske elementer også må være del av mellomspråket som stilene produserer, og det er svært lite ønskelig. Dessuten vil disse semantiske elementene aldri vil bli benyttet, idet XSLT-stilarkenes regler for disse elementene i alle tilfeller vil måtte overstyres og erstattes med stilinformasjonen som bibliografiprosessoren produserer. Det siste forslaget, hvor DocBooks semantiske elementer overhodet ikke er med, er slik sett det mest fornuftige.

Et naturlig spørsmål i den forbindelse er hvorvidt det er akseptabelt å innføre et slikt presentasjonsorientert vokabular på bekostning av de semantiske elementene. Dette er uakseptabelt dersom man velger å se på dokumentet og de genererte siteringene og referansene som kildekode, altså som noe som kan være grunnlag for videre redigering, men burde være mindre problematisk dersom de isteden er et mellomstadium før det endelige produktet.

8.2.3 Andre aktuelle XML DTD'er

DocBook er ikke egnet for alle dokumenttyper, og den er da heller ikke den eneste aktuelle dokumentsentriske DTD'en tilgjengelig idag. En lang rekke DTD'er med ulike formål og ulike fokus er i bruk, men de aller fleste har til felles at de i større eller mindre grad er presentasjonsuavhengige. De samme problemstillingene som vi har sett på for DTD'en til DocBook, er derfor aktuelle også for disse.

Som eksempel på dette kan vi se på TEI. Et filter for TEI burde enkelt kunne lages med DocBook-filteret som modell. Ulikhetene ligger først og fremst i hvilken strategi som er best egnet til å uttrykke presentasjonsinformasjonen i de genererte referansene og siteringene. For TEI er tilpasningslag ikke vanlige. Isteden finnes det et attributt `rend` som kan benyttes på de aller fleste elementer og kan brukes til å assosiere stilinformasjon med elementet. Ved hjelp av dette attributtet kan vi altså enkelt introdusere det samme stilvokabularet i TEI-filer, som vi gjorde i DocBook.

Kapittel 9

Språkfunksjoner

Alle språkavhengige data i dokumentmellomkoden må sendes gjennom språkfunksjoner. I tidligere kapitler har jeg omtalt ulike språkavhengigheter som nødvendiggjør bruk av iallfall disse funksjonene under behandlingen av dokumentmellomkoden:

- Funksjoner som returnerer navn på land, delstater og steder.
- Funksjoner som returnerer nøkkelord.
- Funksjoner som returnerer ordinaltall.
- Funksjoner som returnerer kardinaltall.
- Funksjoner som returnerer datoer.
- Funksjoner som returnerer initialformen av navn.
- Funksjoner som returnerer en forkortet form av et forlagsnavn.
- Funksjoner som returnerer en forkortet form av et tidsskriftnavn.
- Funksjoner som returnerer verktitler på tittelform.
- Funksjoner som returnerer en strengsekvens på en sortert form.

Man vil ganske sikkert finne at ytterligere funksjoner er nødvendige når nye språk legges til bibliografiprosessoren. Det er nemlig ikke mulig, iallfall ikke innenfor denne oppgaven, å lage en uttømmmende oversikt over alle påkrevde språkfunksjoner for alle språk og for alle mulige former for bibliografiske data. Funksjonene, deres argumenter og datastrukturene blir da ikke like egnet for alle språk, og endel *ad hoc*-løsninger må påregnes.

Blant disse funksjonene har jeg valgt ut tre funksjoner som jeg vil diskutere i detalj. Dette er funksjonene for stedsnavn, ordinaltall og tittelformer. Disse er plukket ut som et representativt utvalg for hele gruppen av språkfunksjoner. Stedsnavnfunksjonene er på det nærmeste rene tabelloppslag, men med mange unntak og endel stilavhengigheter. Ordinaltallsfunksjonen er først og fremst avhengig av grammatisk sammenheng, mens tittelfunksjonen er en språkfunksjon som kun er relevant for ett språk og en mindre gruppe stiler.

Flere av de andre språkfunksjonene er også stilavhengige, særlig funksjonene for forkortelser av forlags- og tidsskriftnavn, og derfor må hver stilspesifikasjon eller stilmotor kunne ha tilgang til sin egen gruppe med datafiler. For hver stil skal det altså være mulig med unntakslister, endringslister og parametre som justerer språkfunksjoner. Et åpent spørsmål er nøyaktig hva funksjonene skal gjøre i standardtilfellene, og hva som skal klassifiseres som stilavhengige avvik.

9.1 Stedsnavnsfunksjoner

Stedsnavnsfunksjonene må kunne tilby stilene tre opplysninger basert på et stedsnavn, ISO-landskode og aktuelt språk:

- Oversettelse av stedsnavnet.
- Om nødvendig, kvalifikasjon av stedsnavnet med land, delstat, provins eller lignende.
- Forkortelse av stedsnavnet dersom dette er vanlig praksis for stedsnavnet.

Noen av disse opplysningene kan dog være stilavhengige, for eksempel vil MLA-stilen benytte landsforkortelser som *Manchester, Eng.* [26: 6.6.1], mens de fleste andre stiler vil benytte det fulle landsnavnet. APA-stilen nevner på sin side hvilke byer som kan listes opp uten kvalifikasjon, mens MLA-stilen ikke gjør det.

Stedsnavnsfunksjonen burde i alle tilfeller kunne realiseres som tabelloppslag med unntakslister som følger med stilene.

9.2 Ordinaltallsfunksjoner

Ordinaltall skrives i vestlige språk enten på en numerisk form, for eksempel som *1.*, eller på en tekstlig form, slik som *første*. Iallfall i indoeuropeiske språk er den tekstlige formen av ordinaltall adjektiver, og disse bøyes i samsvar med ordene som adjektivet kvalifiserer. For den tekstlige formen av ordenstall må ordinaltallsfunksjonen altså kjenne de grammatiske kategoriene til ordene som adjektivet kvalifiserer. Formene

primero, *primera*, *primeros*, *primeras* og *primer* er for eksempel mulige former for ordenstallet «første» i spansk [11].

Også den numeriske formen av ordinaltallene kan reflektere grammatiske kategorier, for eksempel russisk 2-е издание av второе издание «andre utgave» eller fransk *1er* for *premier* og *1ère* for *première*.

Variantformer forekommer også; engelsk kan benytte både *1st* og *1.* som numerisk form. Posisjonen vil også kunne variere etter hvilket ord som kvalifiseres; spanske ordinaltall plasseres noen ganger foran og noen ganger etter det kvalifiserte ord. Den tekstlige formene kan i tillegg være være kontekstavhengig, slik som i bikameral skrift, der det er nødvendig med en egen form som begynner med en versal.

Ofte er ordinaltallene laget av kardinaltallene etter nokså enkle regler. I kinesisk lages ordinaltall ved å prefigere kardinaltall med 第 (dì), for eksempel 第一 (dìyi) «første» av 一 (yi) «en». I slike tilfeller vil ordinaltallsfunksjonen kunne generere alle ordinaltall fra en tabell med kardinaltallene, men oftere vil ordinaltallsfunksjonen måtte benytte seg av en kombinasjon av generering og tabelloppslag. Dette er tilfellet i fransk, der ordinaltall kan lages av kardinaltall ved å fjerne utlyden *-e* når denne forekommer, og legge til endelsen *-ième* [52], for eksempel *sixième* «sjette» av *six* «seks». De få unntakene som finnes, blant annet *cinquième* «femte» av *cinq* «fem», må plasseres i tabeller. I mer syntetiske språk vil infleksjonen også måtte genereres delvis ved hjelp av tabeller og regler.

9.3 Tittelfunksjoner

Denne funksjonen er antagelig bare av betydning i engelsk og illustrerer at det kan være nødvendig å lage spesielle funksjoner for enkeltspråk. Funksjonen skal også bare brukes i enkelte stiler. For eksempel benyttes vanlige setningsregler i APA-stilen [2: 4.10], men tittelformer i MLA-stilen [26: 3.7.1].

Hva som er en korrekt tittelform, beror til en viss grad på skjønn [69], men likevel angir stilguidene vanligvis regler som kan brukes som grunnlag for en tittelfunksjon. [1: 15.104] sier at alle ord bortsett fra preposisjoner (uansett lengde), artiklene *the*, *a* og *an*, de sideordnende konjunksjonene *and*, *or*, *for*, *nor* og infinitivsmerket *to* skal begynne med en versal.

Pakken `amsrefs` inneholder en funksjon som konverterer til tittelform etter reglene ovenfor, og i et eksperiment med to hundre titler fra $\mathcal{AMS}$-tidsskrifter utførte denne funksjonen en riktig konvertering [18]. Det er altså være mulig å lage en god funksjon for slik konvertering.

Dette betyr at man også må ta stilling til hvilken form titler i referansedatabasene skal være på. Den ideelle løsningen er å benytte nøyaktig den form som angis på verkets tittelside. Gjør man dette, vil det være nødvendig å utvide tittelfunksjonen til

også å omgjøre titler på tittelform til setningsform, og reglene for hvilke ord som skal begynne med versaler i ordinære setninger, er adskillig mer kompliserte enn reglene for tittelformen. I referansedatabasene har jeg foreslått å benytte egne elementer for å markere egennavn og akronymer, slik at tittelfunksjonen skal kunne være istand til å avgjøre hvilke ord i en tittel som skal begynne med versaler i setningsform. Vanlige ord, slik som månedsnavn og pronomenet *I*, kan oversettes automatisk.

9.4 Datakilder

Datamaterialet for språkfunksjonene kan hentes fra ordbøker, grammatikker, stilguider og eksisterende stiler. Deler av det statiske materialet kan også hentes fra prosjekter som ICU og Openi18n.org[1] dersom lisenser tillater dette.

[1]Openi18n.org er et av prosjektene til Free Standards Group. Prosjektet søker å standardisere portable internasjonaliseringsmekanismer for Linux og andre *open source*-prosjekter. Se `http://www.openi18n.org`.

Kapittel 10

Implementasjon

Implementasjonen av bibliografiprosessoren ble gjort i Perl 5.6 i form av en gruppe moduler samlet under navnet *ibibproc*, der *ibibproc* står for *Internationalised bibliography processor*.

Moduloppdelingen følger modellen som er diskutert tidligere:

- Moduler i navnerommet `IBibProc::Filter` implementerer inn- og ut-filtre.
- Modulen `IBibProc::Process` implementerer kjernen.
- Modulene `IBibProc::IDC` og `IBibProc::IPC` benyttes til utveksling av IDC- og IPC-objekter mellom kjernen og filtrene.
- Moduler i navnerommet `IBibProc::Db` implementerer klasser for innlesing og manipulasjon av referansedatabaser.
- Moduler i navnerommet `IBibProc::Style` implementerer stilmotorer og innlesing av stilspesifikasjoner.

10.1 Referansedatabaser

Modulen `IBibProc::Db` utgjør grensenittet mot referansedatbasene og tilbyr tre funksjoner: En for innlesing av referansedatabaser, en for skriving av referansedatabaser og en for få tilgang til en bestemt innlest referanse.

Lese- og skrivefunksjonene instansierer begge objekter som leser eller skriver et bestemt format. Kun modulen `IBibProc::Db::XML`, som leser *ibibproc*-referansedatabaser, er implementert, men intensjonen er at det senere skal være mulig å lage moduler som `IBibProc::Db::BibTeX` eller `IBibProc::Db::RIS` for direkte innlesing og skriving av disse formatene.

Funksjonen som returnerer en bestemt referanse, returnerer objekter fra modulen `IBibProc::Db::Record`. Disse inneholder en trerepresentasjon av referansen, og de tilbyr aksessfunksjoner som forstår et XPath-lignende språk. Forutsatt at `$db` inneholder en innlest referansedatabase, kan for eksempel publikasjonsstedet i en referanse med referanseidentifikator `aho93` leses ut slik:

```
my $entry = $db->get_entry('aho93');
my $place = $entry->get_path('monographic/imprint/place');
```

10.1.1 Datastrukturer for referansedatabaser

Prosessering av XML-data kan gjøres på flere måter, blant annet med hendelsesbaserte og trebaserte modeller.

Den vanligste hendelsesbaserte modellen er **Simple API for XML** (SAX). Parsere med et API for SAX emitterer hendelser for alle dokumentets begynnelseselementer, avslutningselementer, tegnsekvenser og så videre, i den rekkefølgen disse forekomer i dokumentet. Oppbygningen av eventuelle datastrukturer overlates til applikasjonen. SAX gir en svært rask prosessering med effektiv minnebruk, men kan være svært tungvint å bruke da programmet hele tiden må vedlikeholde tilstandsinformasjon.

I trebaserte modeller – den mest kjente av disse er **Document Object Model** (DOM) – gis programmet tilgang til dokumentet ved tretraverseringsfunksjoner som «finn barn», «finn søsken» og «hent innhold», og hele dokumentet er tilgjengelig for tilfeldige aksesser. Dette fører til at dokumentet i sin helhet må befinne seg i minnet eller iallfall på en mellomlagret form.

For referansedatabaser er det ønskelig med en hybridløsning tilpasset behovet for tilfeldig dataaksess innen hver referanse under formateringen, men med muligheten for raskt å kunne filtrere bort uønskede referanser som ikke refereres av siteringslistene. Da sortering av referanser uansett vil kreve tilgang til disse, vil alle refererte referanser måtte befinne seg i minnet. En tremodell for de refererte referansene og en hendelsesbasert modell for utplukking av refererte referanser, synes da som en fornuftig balanse mellom ressursbruk og tilgjengelighet.

I tillegg til tre-traverseringsfunksjoner er det, som vi har sett, også mulig å benytte XPath-lignende uttrykk til å plukke ut noder fra trærne.

For den første implementasjonen valgte jeg modulen `XML::Parser` som parser. Dette er den klassiske hendelsesbaserte modulen for XML-parsing i Perl med et SAX-lignende API. Senere er jeg blitt klar over at modulen `XML::Twig` tilbyr en hybridløsning svært lik det som er beskrevet ovenfor, og referansedatabasemodulene bør omskrives til å bruke denne.

Trærne bygges opp under parsingen ved at elementer og deres innhold oversettes til Perls egne datastrukturer, slik som *arrays* og *hashes*, ved hjelp av innholdstypene. For eksempel oversettes informasjonsenheten `extent`, definert som

$extent_M = \{dimension_d, price_d, medium_d\}$

til en *hash*. Denne innlesingen er triviell for alle informasjonsenheter bortsett fra dem som har blandet innholdsmodell.

10.1.2 Validering av referanser

Validering av referanser gjøres under konstruksjonen av trærne ved at innholdstyper og lovlige barn kontrolleres ved hver insetting i et tre. En validering basert på DTD kunne vært gjort av en validerende XML-parser, men dette medfører ingen ytterligere fordeler. Etter en innlesing foretar databasemodulene en gjennomgang av alle trær og kontrollerer gyldigheten av alle tekstnoders innhold, for eksempel kontrolleres det om sjekksummer i ISBN og ISSN stemmer.

10.2 Stilmotorer og stilspesifikasjoner

Grensesnittet mot stilmotorene utgjøres av modulen `IBibProc::Style`, som tar seg av innlesing av stilspesifikasjoner, setter sammen parametrene til stilmotorene og instansierer stilmotorer.

Stilmotorene er implementert som et klassehierarki. Grunnklassen `IBibProc::Style::Base` implementerer formatering av alle referansetypene, og denne formateringen er oppdelt i mange trinn med separate metoder. Stilmotorene er underklasser av grunnklassen og overstyrer utvalgte metoder for å oppnå den ønskede formatering for den aktuelle bibliografistilen.

Ettersom stilmotorene arver funksjonalitet fra en grunnklasse, er det viktig å kontrollere at en endring i grunnklassen gir de forventede resultater for alle stilspesifikasjoner. Dette sikres ved at regresstesting av stimotorer og stilspesifikasjoner er del av testpakken.

10.3 Filtre

På samme måte som stilmotorene, er filtrene bygget opp som et klassehierarki med en grunnklasse, `IBibProc::Filter::Base`, og en modul `IBibProc::Filter` som instansierer filtrene. For filtrene er nytteverdien av et slik klassehierarki mindre, men siden kompleksiteten i filtrene varierer mye, er det hensiktsmessig å ha en grunnklasse med standardimplementasjoner av alle metodene.

10.4 Emulering av BibTeX

En typisk BibTeX-sesjon kan se slik ut:

```
$ latex mydocument
$ bibtex mydocument
$ latex mydocument
$ latex mydocument
```

Emulering av BibTeX-gjøres ved hjelp av skriptet `ibibproctex` som brukes på samme måte som programmet `bibtex`:

```
$ latex mydocument
$ ibibproctex mydocument
$ latex mydocument
$ latex mydocument
```

Skriptet instansierer inn- og ut-filtre for TeX og instruerer dem om å lese `aux`-filer og å skrive en `bbl`-fil. For at denne `bbl`-filen skal kunne inkluderes i LaTeX-dokumentet, må dette inneholde kommandoen

```
\usepackage[bibtex]{ibibproc}
```

Opsjonen `bibtex` sørger for at `ibibproc`-pakken redefinerer LaTeX-kommandoene for bibliografier, det vil si `\bibliography`, `\bibliographystyle`, `\cite` og `\nocite`. Videre må referansedatabasene konverteres fra BibTeX-formatet ved hjelp av et konverteringsverktøy, og det må finnes en stilspesifikasjon for den aktuelle bibliografistilen. Alle standardstilene er oversatt manuelt fra BibTeXs stilprogrammer, og resultatene de produserer er sammenlignet visuelt.

Adferden til programmet `ibibproctex` avviker fra BibTeX-programmets adferd kun på noen områder: Det viktigste avviket er at `ibibproctex` ikke benytter biblioteket `kpathsea` til filoppslag, slik BibTeX i de fleste TeX-distribusjoner gjør.

10.5 Konverteringsprogrammer

Konverteringsprogrammer for RIS og BibTeX skal kunne lages med `IBibProc::Db::BibTeX` og `IBibProc::Db::RIS`. Disse skal lese inn kildefilen, bygge opp `IBibProc::Db::Record`-trær for alle referansene og deretter skrive resultatet til målfilen. For både RIS og BibTeX må modulene gjøre en del gjetninger om hvordan data skal tolkes, så det bør forutsettes at brukeren skal finjustere resultatene. Idet disse modulene ikke er implementert, er kun en enkel og uavhengig utgave av programmet `bibtex2ibibproc` implementert.

10.6 Unicode, sortering og språkfunksjoner

Perl 5.6 har god Unicode-støtte på de fleste områder og benytter UTF-8 som intern tegnkoding. Det er derfor ikke nødvendig med noen spesiell håndtering av UTF-8. Oversettelsesfunksjoner mellom ulike tegnkodinger følger også med standarddistribusjonen, men jeg har foreløpig ikke hatt bruk for disse. Mer interessant er Perl-modulen `Unicode::Collate`, som implementerer UCA for Perl. Denne modulen brukes for sortering i stilmotorene, og tilpasningene av standardsorteringen hentes fra språkfunksjonene.

Språkfunksjonene er samlet i en modul `IBibProc::Language`, som tar seg av innlesing av alle datafiler, for eksempel `collation.xml` for UCA-tilpasninger og `dates.xml` for datosystembeskrivelser, og tilbyr de språkfunksjonene som ble beskrevet i kapittel 9.

10.7 Oppsummering

Når denne oppgaven avsluttes, er rammeverket for bibliografiprosessoren ferdigstilt, men mange av komponentene som vil kunne gjøre den brukbar, mangler fremdeles.

Kun to stilmotorer er implementert: En for nummerbaserte nøkkelstiler og en for nøkkelstiler med forkortede forfatternavn i nøklene. Det finnes stilspesifikasjoner for BibTeX-stilene `plain`, `alpha`, `unsrt`, `abbrev` og `siam`, i tillegg til spesifikasjonen `sccons`, som ble brukt i denne oppgaven.

Komplette filtre finnes for TeX og LaTeX, mens for DocBook er kun ut-filteret laget. Enkle utgaver av ut-filtre for ren tekst og XHTML er også laget.

Språkfunksjonene er kun implementert for engelsk og norsk. Også her er det noen mangler, for eksempel mangler funksjonene for tittelformer.

Kapittel 11

Vurdering

Det generelle problemet med formatering av flerspråklige bibliografier er ikke et trivielt problem å løse. Årsaken er først og fremst uforutsigbarheten i bibliografiske data og mangfoldet av stilkrav, og man vil hele tiden måtte etterstrebe en balanse mellom regler og frihet. På den ene siden følger bibliografiske data klare mønstre, men samtidig er unntakene hyppige og mangesidige.

Videre vil man også måtte være forberedt på at nye former for språkavhengigheter vil dukke opp når nye språk skal støttes. En del *ad hoc*-løsninger er derfor ikke til å unngå, og man kan heller ikke forvente at man skal lage en bibliografiprosessor som kan brukes til alle formål.

En komplett og endelig formell beskrivelse av en bibliografiprosessor og egnede dataformater, er derfor neppe mulig. Man må derimot søke å lage programmer og datamodeller som er tilstrekkelig fleksible til at de ikke bryter sammen når nye behov oppstår. Dette krever forutseenhet, et stort datamateriale og inngående kjennskap til detaljer og mønstre ved de aktuelle språk.

11.1 Resultater

Vi har sett at XML er egnet som dataformat for flere formål i en bibliografiprosessor. Dette er tilfellet først og fremst fordi XML enkelt lar oss utstyre tekst med ekstra informasjon og takket være integrasjonen med Unicode. DTD'en som ble utviklet i denne oppgaven, er istand til å representere flerspråklige bibliografier og fungerer i praktisk bruk – den er benyttet for genereringen av siteringene og referansen i denne teksten. Å lage et program som benytter XML-formatene og funksjonene som er forklart i denne teksten, er heller ikke særlig komplisert, men krever mye arbeid og detaljert testing.

11.2 Implementasjonen

ibibproc ble publisert på SourceForge[1] i mars 2003[2]. Den grunnleggende modulstrukturen var da på plass, men programmet ble ikke annonsert på relevante e-postlister, nyhetsgrupper eller nyhetssider, da ingen fungerende utgivelse var tilgjengelig. Mål for den første utgivelsen var et TEX/LATEX-filter, et konverteringsprogram for BIBTEX, emulering av standardsstilene og språk- og sorteringsfunksjoner for engelsk, tysk, fransk og norsk. Ettersom dette målet ennå ikke er nådd, er programmet fremdeles kun tilgjengelig fra CVS.

11.3 Fremtidig utvikling av *ibibproc*

I løpet av denne oppgavens tilblivelse er det blitt klart at `bibx.dtd` (se kapittel 3) vil bli en viktig DTD med mye av representasjonsevnen som er diskutert i denne oppgaven. Ettersom det er ønskelig å ha én standard DTD for utveksling av referansedatabaser, virker det fornuftig å slutte seg til dette prosjektet. Som en første tilnærming vil DTD'en til *ibibproc* modifiseres noe slik at den samme terminologien benyttes i begge DTD'er. De større ulikhetene mellom DTD'ene ligger først og fremst i hvilken grad det tillates oversettelser, translitterasjoner og transkripsjoner av innhold, og i detaljer knyttet til personnavnsemantikk, flerspråklige stedsnavn, omfanget av identifikatortyper og så videre. Jeg håper at det skal være mulig å få inkorporert forslag til forbedringer på disse områdene i `bibx.dtd`.

Sammenlignet med de beslektede prosjekter, ser det ut til at *ibibproc* har mest til felles med prosjektet *Bibulus* til Thomas Widmann: Begge deler samme målsetning, benytter samme implementasjonsspråk, har referansedatabaser i XML, og er GPL-lisensierte. Da en velfungerende bibliografiprosessor er et svært omfattende prosjekt, ville det være ønskelig å samle en gruppe personer med interesse for emnet og å samarbeide om utviklingen. Idet denne oppgaven avsluttes, ser det ut til at dette kan virkeliggjøres gjennom en sammenslåing av *Bibulus* og *ibibproc*, men detaljene rundt dette er ikke klarlagt.

[1] SourceForge er et nettsted som tilbyr hjemmesider og utviklingsverktøy for *open source*-prosjekter. Se `http://www.sourceforge.net`.

[2] Se `http://ibibproc.sourceforge.net`.

Bibliografi

Referanseliste

[1] *The Chicago Manual of Style*. 14. utgave, University of Chicago Press, Chicago, 1993.

[2] *Publication Manual of the American Psychological Association*. 5. utgave, American Psychological Association, Washington, D.C., 2001.

[3] Alvestrand, H. *Tags for the Identification of Languages*. RFC 1766. UNINETT, mars 1995. Tilgjengelig fra `http://www.ietf.org/rfc/rfc1766.txt`.

[4] Beebe, Nelson. Bibliography Prettyprinting and Syntax Checking. *TUGboat*, 14(4): 395–419, desember 1993.

[5] Bennett, Frank G., jr. Camel: Kicking over the bibliographic traces in BibTeX. *TUGboat*, 17(1): 22–28, mars 1996.

[6] Blomberg, Wenche. *Vade me cum: liten kokebok for kildehenvisnings- og litteraturlisteskrivere*. Universitetet i Oslo, Studie- og forskningsadministrativ avdeling, Seksjon for læringsmiljø og studiekvalitet, Oslo, 2000.

[7] Braams, Johannes, David Carlisle, Alan Jeffrey, Leslie Lamport, Frank Mittelbach, Chris Rowley og Rainer Schöpf. *ltbibl.dtx*. 24. april 1997. Tilgjengelig fra CTAN som `macros/latex/base/ltbibl.dtx`.

[8] Bray, Tim, Jean Paoli, C. M. Sperberg-McQueen og Eve Maler (red.). *Extensible Markup Language (XML) 1.0*. W3C Recommendation. World Wide Web Consortium, 6. oktober 2000. Tilgjengelig fra `http://www.w3.org/TR/2000/REC-xml-20001006`.

[9] British Standards. *British Standards 5605: Recommendations for Citing and Referencing Published Materials*. BS 5605:1990. British Standards, Milton Keyns, Storbritannia, 1990.

[10] Butcher, Judith. *Copy-editing: The Cambridge Handbook for Editors, Authors and Publishers*. Cambridge University Press, Cambridge, Storbritannia, 1992.

[11] Butt, John. *Spanish Grammar*. Oxford University Press, Oxford, 2000.

[12] Cowan, John og Richard Tobin (red.). *XML Information Set*. W3C Recommendation. World Wide Web Consortium, 24. oktober 2001. Tilgjengelig fra `http://www.w3.org/TR/2001/REC-xml-infoset-20011024`.

[13] Daly, Patrick W. *Customizing Bibliographic Style Files*. Versjon 3.2, 24. februar 1999. Tilgjengelig fra CTAN som `macros/latex/contrib/supported/custom-bib/makebst.dtx`.

[14] ——. *Natural Sciences Citations and References: Author-Year and Numerical Schemes*. Versjon 7.0, 28. mai 1999. Tilgjengelig fra CTAN som `macros/latex/contrib/supported/natbib/natbib.dtx`.

[15] Davis, Mark. *Unicode Technical Report #22: Character Mapping Markup Language*. UTR 22. Versjon 3, The Unicode Consortium, 16. august 2002. Tilgjengelig fra `http://www.unicode.org/reports/tr22/tr22-3.html`.

[16] Davis, Mark og Martin Dürst. *Unicode Standard Annex #15: Unicode Normalization Forms*. UAX 15. Versjon 4.0.0, The Unicode Consortium, 17. april 2003. Tilgjengelig fra `http://www.unicode.org/unicode/reports/tr15/tr15-23.html`.

[17] Davis, Mark og Ken Whistler. *Unicode Technical Standard 10: Unicode Collation Algorithm*. UTS 10. Versjon 9, The Unicode Consortium, 16. juli 2002. Tilgjengelig fra `http://www.unicode.org/reports/tr10/tr10-9.html`.

[18] Downes, Michael. The `amsrefs` LaTeX package and the `amsxport` BibTeX style. *TUGboat*, 21(3): 201–209, september 2000.

[19] ——. *The* `amsrefs` *package*. Versjon 1.27, 16. april 2002. Tilgjengelig fra `ftp://ftp.ams.org/pub/tex/doc/amsrefs/amsrefs.ps`.

[20] Dürst, Martin J., François Yergeau, Richard Ishida, Misha Wolf, Asmus Freytag og Tex Texin (red.). *Character Model for the World Wide Web 1.0*. W3C Working Draft. World Wide Web Consortium, 30. april 2001. Tilgjengelig fra `http://www.w3.org/TR/2002/WD-charmod-20020430`.

[21] Dürst, Martin og Asmus Freytag. *Unicode Technical Report #20: Unicode in XML and other Markup Languages*. UTR 20. Versjon 7, The Unicode Consortium, 13. juni 2003. Tilgjengelig fra `http://www.unicode.org/reports/tr20/tr20-7.html`.

[22] Endresen, Rolf Theil, Hanne Gram Simonsen og Andreas Sveen (red.). *Innføring i ligvistikk*. Universitetsforlaget, Oslo, 1997.

[23] Fairbairns, Robin. Omega—why Bother with Unicode? *TUGboat*, 16(3): 325–328, september 1995.

[24] Felici, James. *The Complete Manual of Typography*. Adobe Press, Berkeley, California, USA, 2003.

[25] Freytag, Asmus. *Unicode Standard Annex #14: Line Breaking Properties*. UAX 14. Versjon 4.0.0, The Unicode Consortium, 17. april 2003. Tilgjengelig fra `http://www.unicode.org/unicode/reports/tr14/tr14-14.html`.

[26] Gibaldi, Joseph. *MLA style manual and guide to scholarly publishing*. 2. utgave, Modern Language Association of America, New York, 1998.

[27] Gibbons, Jeremy. Hints and Tricks: Hey — it works! *TUGboat*, 19(4): 426–427, september 1998.

[28] Goossens, Michel og Sebastian Rahtz. *The LaTeX Web Companion: Integrating TeX, HTML and XML*. Addison-Wesley Publishing Company, Reading, Massachusetts, USA, 1999.

[29] Haralambous, Yannis. Tour du monde des ligatures. *Cahiers GUTenberg*, (22): 87–99, september 1995. Tilgjengelig fra `http://www.gutenberg.eu.org/pub/GUTenberg/publicationsPDF/22-yannis.pdf`.

[30] ——. From Unicode to Typography, a Case Study: the Greek Script. I *14th International Unicode Conference*, Boston, Massachusetts, USA, mai 1999, side b.10.1–b.10.36. Tilgjengelig fra `http://omega.enstb.org/yannis/pdf/boston99.pdf`.

[31] ——. Unicode, XML, TEI, Ω and Scholary Documents. I *16th International Unicode Conference*, Amsterdam, Nederland, mars 2000.

[32] ——. Unicode et typographie : un amour impossible ? *Document Numérique*, 6(3–4): 107–139, 2002. Tilgjengelig fra `http://omega.enstb.org/yannis/pdf/docnum.pdf`.

[33] Haralambous, Yannis og John Plaice. Ω + *Virtual METAFONT = Unicode + Typography (First Draft)*. 6. januar 1996. Tilgjengelig fra `http://www.loria.fr/services/tex/moteurs/cernomeg.ps.gz`.

[34] Haralambous, Yannis, John Plaice og Johannes Braams. Never again active characters! Ω-Babel. *TUGboat*, 16(4): 418–427, desember 1995.

[35] Harders, Harald. *Multilingual bibliographies: The* `babelbib` *package*. 1. mai 2003. Tilgjengelig fra CTAN som `biblio/bibtex/contrib/babelbib/babelbib.dtx`.

[36] Hillmann, Diane. *Using Dublin Core*. DCMI Recommendation. Dublin Core Metadata Initiative, 12. april 2001. Tilgjengelig fra `http://dublincore.org/documents/2001/04/12/usageguide/`.

[37] Hoekwater, Taco. ConTEXt Publication Module, The user documentation. I *EuroTEX 2001: Proceedings of the Twelfth European TEX conference*, Kerkrade, Nederland, september 2001, side 61–73.

[38] Hoenicka, Markus. *refdb handbook*. 2002. Tilgjengelig fra `http://refdb.sourceforge.net/manual/refdb-manual-pdf.tar.gz`.

[39] Hufflen, Jean-Michel. Vers une extension multilingue de BibTEX. *Cahiers GUTenberg*, (39–40): 23–38, mai 2001.

[40] ——. MLBIBTEX: a New Implementation of BIBTEX. I *EuroTEX 2001: Proceedings of the Twelfth European TEX conference*, Kerkrade, Nederland, september 2001, side 74–94.

[41] ——. Lessons from a Bibliography Program's Reimplementation. *Electronic Notes in Theoretical Computer Science*, 65(3), 2002. Tilgjengelig fra `http://www.elsevier.nl/locate/entcs/volume65.html`.

[42] ——. Multilingual Features for Bibliography Programs: from XML to MlBibTEX. I *EuroTEX 2002: Proceedings of the Thirteenth European TEX conference*, Bachotek, Polen, mai 2002, side 46–59.

[43] ——. Mixing Two Bibliography Style Languages. I *LDTA '03: Third Workshop on Language Descriptions, Tools and Applications*, Warszawa, Polen, april 2003.

[44] Høeg, Ove Arbo. *Vitenskapelig forfatterskap*. 2. utgave, Universitetsforlaget, Oslo, 1971.

[45] International Organization for Standardization. *ISO 9: Documentation — Transliteration of Slavic Cyrillic characters into Latin characters*. ISO 9:1986 (E). International Organization for Standardization, Genève, 1986.

[46] ——. *ISO 690: Documentation — Bibliographic references — Content, form and structure*. ISO 690:1987 (E). International Organization for Standardization, Genève, 1987.

[47] ——. *ISO 639: Code for the Representation of Names of Languages*. ISO 639:1988. International Organization for Standardization, Genève, 1988.

[48] ——. *ISO/IEC 10646-1: Information Technology — Universal Multiple-Octet Coded Character Set (UCS) — Part 1: Architecture and Basic Multilingual Plane*. ISO/IEC 10646-1:1993 (E). International Organization for Standardization, Genève, 1993.

[49] ——. *ISO 3166: Codes for the Representation of Names of Countries and Their Subdivisions — Part 1: Country Codes*. ISO 3166-1:1997 (E). International Organization for Standardization, Genève, 1997.

[50] ——. *ISO/IEC 8859-1: Information Technology — 8-Bit Single-Byte Coded Graphic Character Sets — Part 1: Latin Alphabet No. 1*. ISO/IEC 8859-1:1998. International Organization for Standardization, Genève, 1998.

[51] ——. *ISO 8601: Data elements and interchange formats — Information interchange — Representation of dates and times*. ISO 8601:2000 (E). International Organization for Standardization, Genève, 2000.

[52] Judge, Anne og F. G. Healey. *A Reference Grammar of Modern French*. Edward Arnold, London, 1983.

[53] Kepser, Stephan. *Technical Report, Sonderforschungsbereich 441: A Proof of the Turing-completeness of XSLT and XQuery*. Eberhard-Karls-Universität Tübingen, Tyskland, 13. mai 2002. Tilgjengelig fra `http://tcl.sfs.uni-tuebingen.de/~kepser/papers/xsltxq.pdf`.

[54] Knuth, Donald E. *The TEXbook*. Addison-Wesley Publishing Company, Reading, Massachusetts, USA, 1986.

[55] Lamport, Leslie, Frank Mittelbach og Johannes Braams. *Standard Document Classes for LATEX version 2e*. 21. april 2001. Tilgjengelig fra CTAN som `macros/latex/base/classes.dtx`.

[56] Langballe, Anne M. Hasund. *Faglig forfatterskap*. 5. reviderte utgave, Statens bibliotek- og informasjonshøgskole, Oslo, 1990.

[57] The LaTeX3 Project. *LaTeX2ε for class and package writers*. 12. mars 1999. Tilgjengelig fra CTAN som `macros/latex/base/clsguide.tex`.

[58] Lunde, Ken. *CJKV Information processing*. O'Reilly & Associates, Sebastopol, California, USA, 1999.

[59] Maler, Eve og Jeanne El Andaloussi. *Developing SGML DTDs: From Text to Model to Markup*. Prentice Hall, Upper Saddle River, New Jersey, USA, 1996.

[60] Mathiassen, Terje. *Russisk grammatikk*. Universitetsforlaget, Oslo, 1990.

[61] Patashnik, Oren. *BibTeXing*. 8. februar 1988. Tilgjengelig fra CTAN som `biblio/bibtex/distributions/btxdoc.tex`.

[62] ——. *Designing BibTeX styles*. 8. februar 1988. Tilgjengelig fra CTAN som `biblio/bibtex/distributions/btxhak.tex`.

[63] ——. BibTeX 1.0. *TUGboat*, 15(3): 269–273, september 1994.

[64] ——. BibTeX 101. *TUGboat*, 19(2): 204–207, juni 1998.

[65] Pournader, Roozbeh. Catching up to Unicode. *TUGboat*, 23(1): 80–85, 2002.

[66] Rahtz, Sebastian. Bibliographic Tools. *Literary and Linguistic Computing*, 2(4): 231–241, oktober 1987.

[67] Ramsey, S. Robert. *The Languages of China*. Princeton University Press, Princeton, New Jersey, USA, 1989.

[68] Rhead, David. The "operational requirement" (?) for support of bibliographies. *TUGboat*, 14(4): 425–432, desember 1993.

[69] Ritter, R. M. *The Oxford Guide to Style*. Oxford University Press, Oxford, 2002.

[70] Rounds, Carol. *Hungarian: An Essential Grammar*. Routledge, London, 2001.

[71] Spangen, Inger Cathrine og Anne M. Hasund Langballe. *Ta leseren alvorlig!, eller Hvordan gjøre en faglig tekst faglig*. 3. utgave, Høgskolen i Oslo, Avdeling for journalistikk, bibliotek- og informasjonsfag, Oslo, 2002.

[72] Sperberg-McQueen, C. M. og Lou Burnard (red.). *TEI P4: Guidelines for Electronic Text Encoding and Interchange*. Text Encoding Initiative Consortium, 2002.

[73] Stigleman, Sue. Bibliography Formatting Software: An Updated Buying Guide. *Database*, 17(6): 53–65, desember 1994.

[74] Torp, Arne. Skandinaviske «særbokstaver». *Språknytt*, 1: 1–4, 2002.

[75] The Unicode Consortium. *The Unicode Standard, Version 3.0*. Addison-Wesley Publishing Company, Reading, Massachusetts, USA, 2000.

[76] Unruh, Dominique P. G. `ucs.sty` — *Unicode Support*. 17. desember 2002. Tilgjengelig fra CTAN som `macros/latex/contrib/supported/unicode/ucs.ps.gz`.

[77] van Leunen, Mary-Claire. *A Handbook for Scholars*. Alfred A. Knopf, New York, 1978.

[78] Walsh, Norman og Leonard Muellner. *DocBook: the definitive guide*. O'Reilly & Associates, Sebastopol, California, USA, 1999.

[79] Yergeau, F. *UTF-8, a transformation format of ISO 10646*. RFC 2279. Alis Technologies, januar 1998. Tilgjengelig fra `http://www.ietf.org/rfc/rfc2279.txt`.

Eksempelreferanser

[80] Aho, Alfred V., Ravi Sethi og Jeffrey D. Ullman. *Compilers: Principles, Techniques and Tools*. Addison-Wesley Publishing Company, Reading, Massachusetts, USA, 1986.

[81] Alemayehu, Nega og Peter Willett. Stemming of Amharic Words for Information Retrieval. *Literary and Linguistic Computing*, 17(1): 1–17, april 2002.

[82] Altschul, Stephen F. Fundamentals of database searching. I *Trends guide to bioinformatics*, Elsevier Science, 1998, side 7–9.

[83] Cacouros, Michel. L'évolution de la mise en page dans l'imprimerie (grecque) traditionnelle ou sur la façon dont le livre électronique pourrait — et devrait — bénéficier des pratiques de mise en page anciennes. I *EuroTEX 2003: Proceedings of the Fourteenth European TEX conference*, Brest, Frankrike, juni 2003, side 170–182.

[84] Chen, Wei og Koichi Wada. On Computing the Upper Envelope of Segments in Parallel. *IEEE transactions on parallel and distributed systems*, 13(1): 5–13, januar 2002.

[85] Holly, Michael Ann. *Panofsky and the Foundations of Art History*. Cornell University Press, Ithaca, New York, USA, 1984.

[86] Humphreys, Glyn W. og M. Jane Riddoch. *To See But Not To See: A Case Study of Visual Agnosia*. Psychology Press, 27 Church Road, Hove, East Sussex, BN3 2FA, Storbritannia, 1998.

[87] Maharaj, Savi, Judi Romijn og Carron Shankland. IEEE 1394 Tree Identity Protocol: Introduction to the Case Study. *Formal Aspects of Computing*, 14(3): 200–214, 2003.

[88] Pearson, William R. og David J. Lipman. Improved tools for biological sequence comparison. *Proceedings of the National Academy of Science, USA*, 85(8): 2444–2448, april 1988.

[89] Perelson, Alan S. og Patrick W. Nelson. Mathematical Analysis of HIV-1 Dynamics in Vivo. *SIAM Review*, 41(1): 3–44, mars 1999.

[90] Pevsner, Nikolaus. *Pioneers of Modern Design*. Penguin Books, London, 1991.

[91] Quackenbush, John. Computational analysis of microarray data. *Nature Reviews Genetics*, 2(6): 418–427, juni 2001.

[92] Rosenhan, David L. og Martin E. P. Seligman. *Abnormal Psychology*. 3. utgave, W. W. Norton & Company, New York, 1995.

[93] Silagy, D., Y. Demay og J. F. Agassant. Numerical Simulation of the Film Casting Process. *International journal for numerical methods in fluids*, 30(1): 1–18, mai 1999.

[94] Ефремов, Ю. Н. og А. Д. Чернин. Крупномаштабное звездообразование в галактиках. Успехи физических наук, 173(1): 3–25, januar 2003.

Tillegg A

ibibproc-kildekode

Dette tillegget presenterer noen av kildekodefilene fra implementasjonen av *ibibproc*. Av særlig interesse er DTD'en for referansedatabaser og tegnkartene for TEX-filteret, og disse presenteres i de neste avsnittene.

A.1 DTD for referansedatabaser

```
<!--

 ibibproc.dtd - DTD for ibibproc database files

 Copyright (C) 2003 Marius L. Jøhndal <mariuslj at ifi.uio.no>

 This file is part of ibibproc.

 ibibproc is free software; you can redistribute it and/or modify
 it under the terms of the GNU General Public License as published by
 the Free Software Foundation; either version 2 of the License, or
 (at your option) any later version.

 ibibproc is distributed in the hope that it will be useful,
 but WITHOUT ANY WARRANTY; without even the implied warranty of
 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 GNU General Public License for more details.

 You should have received a copy of the GNU General Public License
 along with ibibproc; if not, write to the Free Software
 Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA

 $Id: ibibproc.dtd,v 1.14 2003/07/24 15:20:11 mariuslj Exp $

   Public identifier:
```

```
     -//Marius Johndal//DTD ibibproc database V1.0//EN

   URL:

     http://ibibproc.sourceforge.net/1.0/ibibproc.dtd

   Namespace:

     http://ibibproc.sourceforge.net/xmlns/ibibproc

-->

<!ENTITY % ns
  "xmlns:ibibproc CDATA #FIXED
    'http://ibibproc.sourceforge.net/xmlns/ibibproc'">

<!ENTITY % country
  'country    NMTOKEN                          #REQUIRED
   state      NMTOKEN                          #IMPLIED'>

<!ENTITY % lang.master
  'xml:lang      NMTOKEN                             #REQUIRED'>

<!ENTITY % lang.inherited
  'xml:lang      NMTOKEN                             #IMPLIED'>

<!ENTITY % authorfunction
  'function      (author|editor|serieseditor|
                  publisher|translator|collector|
                  coauthor|collaborator|other)       #REQUIRED
   otherfunction CDATA                               #IMPLIED'>

<!ENTITY % data.abstr  '(#PCDATA)'>
<!ENTITY % data.static '(#PCDATA)'>
<!ENTITY % data.lang   '(#PCDATA)'>

<!-- Text with typographical mark-up. -->
<!ENTITY % data.typo   '(#PCDATA|protect|special)*'>

<!ELEMENT protect      (#PCDATA)>

<!ELEMENT special      (default, filter+)>

<!ELEMENT default      (#PCDATA)>

<!ELEMENT filter       (#PCDATA)>
<!ATTLIST filter       name CDATA #REQUIRED>
```

```
<!-- Translatable strings. -->
<!ENTITY % data.trans  '(#PCDATA|primary|alternative|protect|special)*'>

<!ELEMENT primary      %data.typo;>
<!ATTLIST primary      %lang.inherited;>

<!ELEMENT alternative  %data.typo;>
<!ATTLIST alternative  text  (transliteration|translation) #REQUIRED
                       %lang.inherited;>

<!-- Bibliography. -->
<!ELEMENT bibliography (entry+)>
<!ATTLIST bibliography %ns;>

<!-- Bibliographic entry. -->
<!ELEMENT entry        (((analytic, (monographic,
                          series?)?)|(monographic?, series?)),
                        description?)>
<!ATTLIST entry        id        ID      #REQUIRED
                       entrydate CDATA   #IMPLIED
                       type      NMTOKEN #REQUIRED
                       %lang.master;>

<!-- Compositional levels. -->
<!ELEMENT analytic     (authors, titles)>
<!ATTLIST analytic     %lang.inherited;>

<!ELEMENT monographic  (authors?, titles, scope?, imprint?)>
<!ATTLIST monographic  %lang.inherited;>

<!ELEMENT series       (authors?, titles, scope?, imprint?)>
<!ATTLIST series       %lang.inherited;>

<!-- Author information. -->
<!ELEMENT authors      (person|entity)*>
<!ATTLIST authors      completeness (complete|
                                     incomplete|
                                     anonymous)       'complete'
                       %lang.inherited;>

<!ELEMENT person       (name, affiliation?, email?, url?)>
<!ATTLIST person       %authorfunction;
                       inversion    (style|none)      #IMPLIED
                       %lang.inherited;>

<!ELEMENT name         (firstname*, preposition*, surname+,
                        lineage*, titula*)>
<!ATTLIST name         class     (actual|unrevealed|
                                  surmised|pseudonym) 'actual'
```

```
                      %lang.inherited;>

<!ELEMENT firstname    %data.trans;>
<!ATTLIST firstname    type      (full|initial|other) 'full'
                       othertype CDATA                #IMPLIED
                       %lang.inherited;>

<!ELEMENT preposition  %data.trans;>
<!ATTLIST preposition  %lang.inherited;>

<!ELEMENT surname      %data.trans;>
<!ATTLIST surname      %lang.inherited;>

<!ELEMENT lineage      %data.trans;>
<!ATTLIST lineage      %lang.inherited;>

<!ELEMENT titula       %data.trans;>
<!ATTLIST titula       %lang.inherited;>

<!ELEMENT affiliation  %data.trans;>
<!ATTLIST affiliation  %lang.inherited;>

<!ELEMENT email        %data.static;>

<!ELEMENT url          %data.static;>

<!ELEMENT entity       %data.trans;>
<!ATTLIST entity       class      (institution|company|
                                   comitee|school|
                                   university|organisation|
                                   unknown|other)            #REQUIRED
                       otherclass CDATA                     #IMPLIED
                       %authorfunction;
                       %lang.inherited;>

<!-- Title information. -->
<!ELEMENT titles       (title+)>
<!ATTLIST titles       %lang.inherited;>

<!ELEMENT title        %data.trans;>
<!ATTLIST title        class      (maintitle|subtitle|other) 'maintitle'
                       otherclass CDATA                      #IMPLIED
                       %lang.inherited;>

<!-- Part information. -->
<!ELEMENT scope        (unit)+>
<!ATTLIST scope        %lang.inherited;>

<!ELEMENT unit         (single|range)*>
```

```
<!ATTLIST unit          class       (volume|issue|page|book|
                                     part|chapter|appendix|
                                     section|footnote|note|
                                     paragraph|line|word|
                                     stanza|verse|scene|
                                     series|table|figure|
                                     example|illustration|
                                     plate|freetext|other)     #REQUIRED
                        otherclass CDATA                      #IMPLIED
                        %lang.inherited;>

<!ELEMENT single        %data.trans;>
<!ATTLIST single        %lang.inherited;>

<!ELEMENT range         (start?, end?)>
<!ATTLIST range         %lang.inherited;>

<!ELEMENT start         %data.trans;>
<!ATTLIST start         %lang.inherited;>

<!ELEMENT end           %data.trans;>
<!ATTLIST end           %lang.inherited;>

<!ELEMENT container     ((single|range), container?)+>
<!ATTLIST container     %lang.inherited;>

<!-- Publication information. -->
<!ELEMENT imprint       (place?, pubname?, edition?, version?,
                         pubdate?, pagination?, locators?, numbers?,
                         refs?, copyright?)>
<!ATTLIST imprint       %lang.inherited;>

<!ELEMENT place         %data.trans;>
<!ATTLIST place         %country;
                        %lang.inherited;>

<!ELEMENT pubname       %data.trans;>
<!ATTLIST pubname       %lang.inherited;>

<!ELEMENT edition       %data.trans;>
<!ATTLIST edition       class       (arabic|text|other) #REQUIRED
                        otherclass CDATA                #IMPLIED
                        %lang.inherited;>

<!ELEMENT version       %data.trans;>
<!ATTLIST version       %lang.inherited;>

<!ELEMENT pubdate       %data.trans;>
<!ATTLIST pubdate       class       (fulldate|monthyear|year|
```

```
                              text|other)              'fulldate'
                    otherclass CDATA                   #IMPLIED
                    %lang.inherited;>

<!ELEMENT pagination   EMPTY>
<!ATTLIST pagination   type       (consecutive|
                                   nonconsecutive)     #REQUIRED>

<!ELEMENT locators     (locator+)>

<!ELEMENT locator      %data.static;>
<!ATTLIST locator      class      (lccn|mrnumber|doi|isbn|
                                   issn|isrc|isrn|ismn|isan|
                                   sici|bici|
                                   coden|pmid|
                                   uri|bibcode|oai|pii|
                                   other)              #REQUIRED
                       otherclass CDATA                #IMPLIED>

<!ELEMENT numbers      (number+)>

<!ELEMENT number       %data.static;>
<!ATTLIST number       class      (standard|other)     #REQUIRED
                       otherclass CDATA                #IMPLIED>

<!ELEMENT refs         (ref+)>

<!ELEMENT ref          %data.static;>
<!ATTLIST ref          type          (ctan|url|other)  #REQUIRED
                       othertype     CDATA             #IMPLIED
                       function      (see|original|local|
                                      archive|copy|other) #REQUIRED
                       otherfunction CDATA             #IMPLIED
                       accessed      CDATA             #IMPLIED>

<!ELEMENT copyright    %data.trans;>
<!ATTLIST copyright    %lang.inherited;>

<!-- Extent and medium information. -->
<!ELEMENT extent       (price?, dimensions?, medium?)>
<!ATTLIST extent       %lang.inherited;>

<!ELEMENT price        %data.trans;>
<!ATTLIST price        %lang.inherited;>

<!ELEMENT dimensions   %data.trans;>
<!ATTLIST dimensions   %lang.inherited;>

<!ELEMENT medium       %data.trans;>
```

```
<!ATTLIST medium       %lang.inherited;>

<!-- Descriptive information. -->
<!ELEMENT description  (keywords?, categories?, contents?, abstract?,
                        annotations?, notes?)>
<!ATTLIST description  %lang.inherited;>

<!ELEMENT keywords     (keyword)+>
<!ATTLIST keywords     %lang.inherited;>

<!ELEMENT keyword      %data.trans;>
<!ATTLIST keyword      %lang.inherited;>

<!ELEMENT categories   (category+)>
<!ATTLIST categories   %lang.inherited;>

<!ELEMENT category     %data.trans;>
<!ATTLIST category     %lang.inherited;>

<!ELEMENT contents     %data.trans;>
<!ATTLIST contents     %lang.inherited;>

<!ELEMENT abstract     %data.trans;>
<!ATTLIST abstract     %lang.inherited;>

<!ELEMENT annotations  (annotation+)>
<!ATTLIST annotations  %lang.inherited;>

<!ELEMENT annotation   %data.trans;>
<!ATTLIST annotation   %lang.inherited;>

<!ELEMENT notes        (note+)>
<!ATTLIST notes        %lang.inherited;>

<!ELEMENT note         %data.trans;>
<!ATTLIST note         %lang.inherited;>

<!--
  Local Variables:
  coding: utf-8
  End:
-->
```

A.2 Tegnkart for TEX-filteret

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+0020		!	"	#	$	%	&	’	(	)	*	+	,	-	.	/
U+0030	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
U+0040	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
U+0050	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
U+0060	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
U+0070	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	
U+00A0		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬		®	¯
U+00B0	°	±	²	³	’	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
U+00C0	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
U+00D0	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
U+00E0	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
U+00F0	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ
U+0100	Ā	ā	Ă	ă	Ą	ą	Ć	ć	Ĉ	ĉ	Ċ	ċ	Č	č	Ď	ď
U+0110	Đ	đ	Ē	ē	Ĕ	ĕ	Ė	ė	Ę	ę	Ě	ě	Ĝ	ĝ	Ğ	ğ
U+0120	Ġ	ġ	Ģ	ģ	Ĥ	ĥ			Ĩ	ĩ	Ī	ī	Ĭ	ĭ	Į	į
U+0130	İ	ı	IJ	ij	Ĵ	ĵ	Ķ	ķ		Ĺ	Í	Ļ	ļ	Ľ	ľ	
U+0140		Ł	ł	Ń	ń	Ņ	ņ	Ň	ň		ŉ	Ŋ	ŋ	Ō	ō	Ŏ
U+0150	ŏ	Ő	ő	Œ	œ	Ŕ	ŕ	Ŗ	ŗ	Ř	ř	Ś	ś	Ŝ	ŝ	Ş
U+0160	ş	Š	š	Ţ	ţ	Ť	ť			Ũ	ũ	Ū	ū	Ŭ	ŭ	Ů
U+0170	ů	Ű	ű	Ų	ų	Ŵ	ŵ	Ŷ	ŷ	Ÿ	Ź	ź	Ż	ż	Ž	ž
U+01C0														Ǎ	ǎ	Ǐ
U+01D0	ǐ	Ǒ	ǒ	Ǔ	ǔ											
U+01E0			Ǽ	ǽ			Ǧ	ǧ	Ǩ	ǩ	Ǫ	ǫ				
U+01F0	ǰ				Ǵ	ǵ			Ǹ	ǹ						
U+0210															Ȟ	ȟ
U+0220							Ȧ	ȧ	Ȩ	ȩ					Ȯ	ȯ
U+0230			Ȳ	ȳ												

Tabell A.1: TEX-filterets definisjoner for U+0000–U+0240. Blokkene *Basic Latin*, *Latin-1 Supplement*, *Latin Extended-A* og *Latin Extended B* omfattes.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+02C0							ˆ	ˇ		ˉ	’	‵		ˍ		
U+02D0									˘	˙	˚	˛	˜	˝		
U+1E00			Ḃ	ḃ	Ḃ	ḃ	B̲	b̲			Ḋ	ḋ	Ḍ	ḍ	D̲	d̲
U+1E10	Ḑ	ḑ													Ḟ	ḟ
U+1E20	Ḡ	ḡ	Ḣ	ḣ	Ḥ	ḥ	Ḧ	ḧ	Ḩ	ḩ						
U+1E30	Ḱ	ḱ	Ḳ	ḳ	K̲	k̲	Ḷ	ḷ			L̲	l̲			Ḿ	ḿ
U+1E40	Ṁ	ṁ	Ṃ	ṃ	Ṅ	ṅ	Ṇ	ṇ	N̲	n̲						
U+1E50					Ṕ	ṕ	Ṗ	ṗ	Ṙ	ṙ	Ṛ	ṛ			R̲	r̲
U+1E60	Ṡ	ṡ	Ṣ	ṣ							Ṫ	ṫ	T̲	t̲	T̲	t̲
U+1E70													Ṽ	ṽ	V̲	v̲
U+1E80	Ẁ	ẁ	Ẃ	ẃ	Ẅ	ẅ	Ẇ	ẇ	W̲	w̲	Ẋ	ẋ	Ẍ	ẍ	Ẏ	ẏ
U+1E90	Ẑ	ẑ	Ẓ	ẓ	Z̲	z̲	h̲	ẗ	ẘ	ẙ						
U+1EA0	Ạ	ạ														
U+1EB0									Ẹ	ẹ			Ẽ	ẽ		
U+1EC0											Ị	ị	Ọ	ọ		
U+1EE0					Ụ	ụ										
U+1EF0			Ỳ	ỳ	Ỵ	ỵ			Ỹ	ỹ						
U+2000	NQ SP		MQ SP		EN SP	EM SP				TH SP						
U+2010				–	—				‘	’	,		“	”	„	
U+2020	†	‡	•				…									
U+2030	‰	‱								‹	›					
U+2190		↑	→	↓	↔	↕										
U+21C0													⇌			
U+21D0			⇒		⇔											

Tabell A.2: TEX-filterets definisjoner for U+0250–U+21F0. Blokkene *Spacing Modifying Letters*, *Latin Extended Additional General Punctuation Marks* og *Arrows* omfattes.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+2200	∀		∂	∃		∅	Δ	∇	∈	∉		∋	∌			∏
U+2210	∐	∑	−	∓		∕	∖	∗	∘	∙	√			∝	∞	
U+2220	∠			∣		∥		∧	∨	∩	∪	∫			∮	
U+2230							∶						∼			
U+2240	≀			≃	≄	≅			≈	≉				≍		
U+2250	�doteq															
U+2260	≠	≡	≢		≤	≥					≪	≫		≮		
U+2270											≺	≻				
U+2280			⊂	⊃	⊄	⊅	⊆	⊇							⊎	
U+2290		⊑	⊒	⊓	⊔	⊕	⊖	⊗	⊘	⊙						
U+22A0			⊢	⊣	⊤	⊥										
U+22C0	⋀	⋁	⋂	⋃	⋄	⋅	⋆		⋈							
U+22E0			⋢	⋣											⋮	⋯
U+22F0		⋱														
U+2300									⌈	⌉	⌊	⌋				

Tabell A.3: TEX-filterets definisjoner for U+2200–U+23F0. Blokkene *Mathematical Operators* og *Miscellaneous Technical* omfattes.

Ligatur	Glyf	Skrifttyper	Unicode-kodepunkt
--	–	Computer Modern, European Computer Modern	U+2013
---	—	Computer Modern, European Computer Modern	U+2014
'!	¡	Computer Modern	U+00A1
'?	¿	Computer Modern	U+00BF
?`	¡	European Computer Modern	U+00A1
!`	¿	European Computer Modern	U+00BF
''	”	Computer Modern, European Computer Modern	U+201D
``	“	Computer Modern, European Computer Modern	U+201C
<<	«	European Computer Modern	U+00AB
>>	»	European Computer Modern	U+00BB
,,	„	European Computer Modern	U+201E

Tabell A.4: Ligaturer i TEX/LATEX. Disse tegnsekvensene brytes opp av TEX-filteret slik at TEX og LATEX ikke oppfatter dem som ligaturer. En del av dem er allerede håndtert under oversettelsen fra UTF-8, men de er med i listen for kompletthet og i tilfelle fremtidige endringer i tegnoversettelsestabellene ikke skulle oppfange dem. Alle sekvensene deles ved hjelp av {} som settes inn mellom hvert tegn. Sekvensene brytes opp uavhengig av TEX-format, ettersom en oppbrytning har minimal effekt dersom sekvensen ikke er en ligatur. Tabellen viser hvilke av standardskrifttypene som definerer de ulike ligaturene. Definisjonene for Computer Modern finnes i comlig.mf, mens de for European Computer Modern finnes i excligtb.mf.

Symbol	Makro	LaTeX 2ε-utgave	Fontkoding	Unicode
—	\textemdash	1994/12/01	alle	U+2014
–	\textendash	1994/12/01	alle	U+2013
¡	\textexclamdown	1994/12/01	alle	U+00A1
¿	\textquestiondown	1994/12/01	alle	U+00BF
“	\textquotedblleft	1994/12/01	alle	U+201C
”	\textquotedblright	1994/12/01	alle	U+201D
‘	\textquoteleft	1994/12/01	alle	U+2018
’	\textquoteright	1994/12/01	alle	U+2019
•	\textbullet	1994/12/01	alle	U+2022
·	\textperiodcentered	1994/12/01	alle	U+00B7
\	\textbackslash	1995/12/01	alle	U+005C
|	\textbar	1995/12/01	alle	U+007C
<	\textless	1995/12/01	alle	U+003C
>	\textgreater	1995/12/01	alle	U+003E
^	\textasciicircum	1995/12/01	alle	U+005E
~	\textasciitilde	1995/12/01	alle	U+007E
®	\textregistered	1995/12/01	alle	U+00AE
™	\texttrademark	1995/12/01	alle	U+2022
«	\guillemotleft	1994/12/01	T1	U+00AB
»	\guillemotright	1994/12/01	T1	U+00BB
‹	\guilsinglleft	1994/12/01	T1	U+2039
›	\guilsinglright	1994/12/01	T1	U+203A
„	\quotedblbase	1994/12/01	T1	U+201E
‚	\quotesinglbase	1994/12/01	T1	U+201A
"	\textquotedbl	1994/12/01	T1	U+0022

Tabell A.5: Lange tegnmakroer i LaTeX 2ε.

Symbol	Makro	LaTeX 2ε-utgave	Fontkoding	Unicode
SS	\SS	1994/12/01	alle	?
Ð	\DH	1994/12/01	T1	U+00D0
ð	\dh	1994/12/01	T1	U+00F0
Đ	\DJ	1994/12/01	T1	U+0110
đ	\dj	1994/12/01	T1	U+0111
Ŋ	\NG	1994/12/01	T1	U+014A
ŋ	\ng	1994/12/01	T1	U+014B
Þ	\TH	1994/12/01	T1	U+00DE
þ	\th	1994/12/01	T1	U+00FE
IJ	\IJ	Defineres av babel	T1	U+0132
ij	\ij	Defineres av babel	T1	U+0133

Tabell A.6: Korte tegnmakroer i LaTeX 2ε. Disse kommer i tillegg til makroene som er definert i plain TEX (se tabell 8.1.)

Symbol	LaTeX 2ε-makro	plain TeX-makro/ligatur
—	`\textemdash`	`---`
–	`\textendash`	`--`
¡	`\textexclamdown`	`!'`
¿	`\textquestiondown`	`?'`
“	`\textquotedblleft`	`''`
”	`\textquotedblright`	``` `` ```
‘	`\textquoteleft`	`'`
’	`\textquoteright`	``` ` ```
•	`\textbullet`	`\bullet` (mat.modus)
·	`\textperiodcentered`	`\cdot` (mat.modus)
\	`\textbackslash`	`\backslash` (mat.modus)
|	`\textbar`	`\mid` (mat.modus)
<	`\textless`	< (mat.modus)
>	`\textgreater`	> (mat.modus)
^	`\textasciicircum`	`^` i `verbatim`-miljø
~	`\textasciitilde`	`~` i `verbatim`-miljø

Tabell A.7: plain TeX-makroer og -ligaturer for lange tegnmakroer.

Symbol	Makro	TeX-formater	Fontkoding
´	`\'`	plain TeX og alle LaTeX-utgaver	alle
`	`\``	plain TeX og alle LaTeX-utgaver	alle
ˆ	`\^`	plain TeX og alle LaTeX-utgaver	alle
˜	`\~`	plain TeX og alle LaTeX-utgaver	alle
¯	`\=`	plain TeX og alle LaTeX-utgaver	alle
˙	`\.`	plain TeX og alle LaTeX-utgaver	alle
¨	`\"`	plain TeX og alle LaTeX-utgaver	alle
¸	`\c`	plain TeX og alle LaTeX-utgaver	alle
˘	`\u`	plain TeX og alle LaTeX-utgaver	alle
ˇ	`\v`	plain TeX og alle LaTeX-utgaver	alle
˝	`\H`	plain TeX og alle LaTeX-utgaver	alle
.	`\d`	plain TeX og alle LaTeX-utgaver	alle
¯	`\b`	plain TeX og alle LaTeX-utgaver	alle
⁀	`\t`	plain TeX og alle LaTeX-utgaver	alle
°	`\r`	LaTeX 2ε 1994/12/01 og nyere	alle
˛	`\k`	LaTeX 2ε 1994/12/01 og nyere	T1

Tabell A.8: Diakritikamakroer i TeX og LaTeX.

Tillegg B

Bibliografiformater

Dette tillegget gir en oversikt over felter og typer som benyttes i BibTeX- og RIS-filer.

B.1 BibTeX

`address`	Utgivers addresse.
`annote`	Kommentar.
`author`	Forfattere.
`booktitle`	Boktittel.
`chapter`	Kapittel eller inndelingsenhet.
`crossref`	Kryssreferanse.
`edition`	Bokutgave.
`editor`	Redaktører.
`howpublished`	Utgivelsesmåte.
`institution`	Institusjon.
`journal`	Tidsskriftnavn.
`key`	Sorteringsnøkkel.
`month`	Utgivelsesmåned.
`number`	Tidsskriftnummer eller serienummer.
`organization`	Organisasjon.
`pages`	Sideavgrensning.
`publisher`	Utgiver.
`school`	Skole.
`series`	Serienavn.
`tittel`	Verktittel.
`volume`	Bind.
`year`	Utgivelsesår.

Tabell B.1: Felter i BibTeX. Tabellen er basert på [61].

`abstract`	Sammendrag av dokumentet.
`contents`	Dokumentets innholdsfortegnelse.
`keywords`	Nøkkelord som beskriver dokumentet.
`category`	Kategorisering av dokumentet.
`language`	Dokumentets språk.
`URL`	Peker til en HTML-, PostScript- eller PDF-utgave av dokumentet eller til en omtale av det refererte dokumentet.
`affiliation`	Forfatteres tilknytning.
`copyright`	Rettighetsinformasjon.
`location`	Et sted med tilknytning til publikasjonen, for eksempel stedet der en konferanse ble holdt.
`price`	Pris.
`size`	Et verks fysiske dimensjoner.
`ISBN`	International Standard Book Number.
`ISSN`	International Standard Serial Number.
`LCCN`	Library of Congress Call Number.
`mrnumber`	Mathematical Reviews Number.
`doi`	Digital Object Identifier.

Tabell B.2: Vanlige uoffisielle tilleggsfelter i BibTeX-filer. Filter med vilkårlige navn er tillatt i BibTeX, men bibliografistilene må tilpasses for å benytte seg av de nye feltene. Feltene i denne tabellen er tatt fra Dana Jacobsens oversikt over BibTeX (se `http://www.ecst.csuchico.edu/~jacobsd/bib/formats/bibtex.html`) og fra *The Collection of Computer Science Bibliographies* (se `http://liinwww.ira.uka.de/bibliography/index.html`).

`article`	En tidsskrift- eller magasinartikkel.
`book`	Bok med kjent utgiver.
`booklet`	Trykt og innbundet verk, men uten utgiver.
`inbook`	Del av en bok.
`incollection`	Del av en bok med egen tittel.
`inproceedings`[a]	En artikkel i en konferanseinnleggssamling.
`manual`	Teknisk dokumentasjon.
`mastersthesis`	Masteroppgave.
`phdthesis`	Doktorgradsoppgave.
`proceedings`	Konferanseinnleggssamling.
`techreport`	En rapport publisert av en institusjon.
`unpublished`	Dokument som ikke er formelt publisert.
`misc`	Samletype for alt som ikke passer under andre typer.

[a] `conference` kan brukes som synonym.

Tabell B.3: Referansetyper i BibTeX. Tabellen er basert på [61].

B.2 RIS

ABST	Sammendrag	ADVS	Audiovisuelt materiale
ART	Kunstverk	BILL	Lov, resolusjon
BOOK	Bok	CASE	Rettssak
CHAP	Bokkapittel	COMP	Dataprogram
CONF	Konferansenotatsamling	CTLG	Katalog
DATA	Datafil	ELEC	Elektronisk dokument
GEN	Generell type	ICOMM	Internett-kommunikasjon
INPR	Under trykking	JFULL	Tidsskrift
JOUR	Tidsskriftartikkel	MAP	Kart
MGZN	Magasinartikkel	MPCT	Film
MUSIC	Musikkstykke	NEWS	Avis
PAMP	Pamflett	PAT	Patent
PCOMM	Personlig kommunikasjon	RPRT	Rapport
SER	Seriepublikasjon	SLIDE	Lysark
SOUND	Lydopptak	STAT	Statutt
THES	Oppgave, avhandling	UNBILL	Resolusjon
UNPB	Upublisert verk	VIDEO	Videoopptak

Tabell B.4: Referansetyper i RIS. Basert på tabell i [38].

TY	Referansetype. Referansestart.	ER	Referanseslutt.
ID	Referanseidentifikator.	T1, TI, CT	Primær tittel.
T2	Sekundær tittel.	T3	Tertiær tittel.
A1, AU†	Forfatternavn.	A2, ED†	Redaktørnavn.
A3†	Serieredaktør.	Y1, PY	Primær publikasjonsdato.
Y2	Sekundær publikasjonsdato.	N1	Notater.
N2, AB	Sammendrag.	KW†	Nøkkelord.
RP	*Reprint*-status.	JF	Fullt tidsskriftnavn.
JO, JA	Tidsskriftforkortelse.	J1	Alternativ tidsskriftforkortelse.
J2	Alternativ tidsskriftforkortelse.	VL	Tidskriftårgang.
IS, CP	Tidskriftnummer.	SP	Første refererte side.
EP	Siste refererte side.	CY	Utgivelsessby.
PB	Utgiver.	SN	ISBN eller ISSN.
AD	Kontaktadresse.	AV	Tilgjengelighetsinformasjon.
UR	URL.	U1–U5, M1–M3	Brukerfelter.

Tabell B.5: Felter i RIS. Basert på tabell i [38]. Flere feltnavn adskilt med komma indikerer synonyme feltnavn. I tillegg er feltet `BT` identisk med `T1` for typene `BOOK` og `UNPB`, ellers identisk med `T2`. Felter merket med † kan gjentas for hver forekomst av opplysningen, for eksempel for hver forfatter.

Denne rapporten er satt med LaTeX og typesnittene Adobe Garamond og Optima. I tillegg til disse er ARPHIC PL KaitiM GB brukt for forenklede *hanzi*, ARPHIC PL KaitiM Big5 for tradisjonelle *hanzi*, Kochi Mincho for *kanji* og *kana*, Baekmuk Batang for *hangul* og *hanja*, Kerkis for gresk, Computer Modern Super for kyrillisk og matematikk, Vietnamese Computer Modern for vietnamesisk, Armenian Times fra ArmTeX for armensk, ETHIO for etiopisk og TeX TIPA for IPA-symboler. Siteringer og referanselister er generert med programpakken *ibibproc*.