BIBELTEKST ON-LINE

P R O S J E K T R A P P O R T

Nr. 1

for Bibel-prosjektet

12. november 1979

Per-Bjørn Pedersen Michael Gillow

Bjørn Eide

1. INNLEIING

Våren og sommaren 1976 vart det ved Nordisk institutt, Prosjekt for datamaskinell språkbehandling (PDS), Universitetet i Bergen arbeidd med dei nye omsettingane av Det nye testamentet for å gjere dei datamaskinelt tilgjengelege. Arbeidet baserte seg på trykkeribband og målet var å lage KWIC-konkordansar og ordlister av eit nytt og viktig materiale til språkvitskapleg bruk. Det Norske Bibelselskap (DNB) hadde gjeve si velsigning til dette og late oss få kopiar av trykkeribanda - eit avgjerande grunnlag for i det heile å tilrettelegge materialet innan rimeleg tid. DNB gjorde seg nytte av våre produkt ved å ta ut selektive lister til arbeidsgrunnlag for fastlegging av den endelege språkdrakta for nyomsettingane av heile Bibelen.

Hausten 1978 kom heile Bibelen i den nye omsettinga både på bokmål og nynorsk. Det reiste med eit gong spørsmålet om nye bibelordbøker. Frå språkvitskapleg hald vart det uttrykt sterke ønske om å gjere dette store og aktuelle språktilfanget tilgjengeleg for nærare studium og etterrøking. For båe oppgåver ville det ut frå 1976-løynslene vere naturleg å gå vegen om datamaskinene for å gjere tilfanget tilgjengeleg og brukeleg. Dette førte til at det vart gjord avtale mellom DNB og PDS om gjennomføring av eit liknande prosjekt for heile Bibelen som det som vart gjort med Det nye testamentet i 1976.

Dette arbeidet kom i gang fyrst i april i år og dei fyrste resultata ligg føre.

Bibeltekst On-line
Per-Bjørn Pedersen, Michael Gillow, Bjørn Eide
Proceedings of NODALIDA 1979, pages 77-88

2. PLANLEGGING

Listene våre i 1976 viste at kvart nytestamente inneheldt om lag 170 000 laupande ord (tokens). Eit grovt overslag for heile Bibelen gjev eit tilfang på nær 3/4 million ord.

Med våre 76-røynsler ville det bli ein enorm jobb å lage konkordansar og lister av ymse slag for ein heil bibel med det programsystemet vi då bygde opp. Eit program-system som vandra langs tradisjonelle KWIC-vegar og som var like effektivt som det meste tilsvarande så langt vi kunne dømme. Preet ved 76-systemet var at vi visste det fungerte. Vi kunne såleis komme i gang med produskjon nokså snart medan eit nytt opplegg ville krevje grundig planlegging og vurdering og ei uviss utviklingstid. Med dei store datamengdene dette nye prosjektet innebar fann vi det likevel viktig å leite nøye og grurdig etter nye vegar å gå.

Fleire modellar vart skisserte av arbeidsgruppa: Michael Gillow, Per-Bjørn Pedersen og Bjørn Fide med leiar for PDS - Kolbjørn Heggstad - som formell leiar.

Det einaste som i utgangspunktet var klårt var at vi måtte lage eit fleksibelt og dynamisk system for lesing av trykkeriband. Allereide i 1976 hadde vi sysla med tankar kring denne problematikken. Det vi då laga til vart eit einfelt lese-/styre-program med ein utskriftsdel og ein primitiv tilstandsgrammatikk. Sjølv i den primitive forma systemet fekk i 1976 hadde det vist seg som tenleg og brukeleg. Denne delen fann vi difor nyttig å byggje vidare på i vårt nye arbeid. Tilstandsgrammatikken er blitt utvikla til ein svært fleksibel og allmenn reiskap som også - etter vårt syn - er lettfatteleg og bruksvenleg. Men den gjev høve til å byggje så kompliserte strukturar og mønster at det vart heilt naudsynt å byggje inn i systemet gode og fleksible avlusings- og utprøvings-mekanismar.

For brukar-sida såg vi det viktig å lage eit interaktivt system som i stor grad skulle vere parameter-styrt og som skulle gje høve til å lage fleire ulike typar lister ut frå same grunndata, med eit absolutt krav om gode høve til å produsere KWIC-konkordansar etter behov utan fyrst å måtte lage fullstendige konkordansar for heile materialet.

Med slike krav for eit ny-opplegg var det sjølvsagt nærliggjande å tenkje på å bruke database-systemet DMS1100. Men ved vårt universitet var dette utelukka med dei vilkår det er for bruk av DMS1100 i dag. Vi måtte difor sjå oss om etter noko anna. Vårt val vart då å byggje vårt eige skreddarsydde database-system ved hjelp av såkalla MSAM-filar som er tilg-

jengeleg frå og med COBOL-74.

Det var alleredе i utgangspunktet semje om å bruke COBOL som programmeringsspråk fordi det var det språket som troikaen hadde best fellas kjennskap til.

Som nemnt ovanfor tok arbeidet for alvor til i april. I fyrste fase - frå april og fram til juli - gjekk utviklinga av systemet for lesing av trykkeri-data og fastlegginga av brukar-delen parallelt. Det vesentlege av mai månad vart brukt til å skissere og prøve ut tre-fire ulike modellar. Modellane bygde alle på MSAM-filar, men i ulik grad og omfang. Det vart gjort prøver med splitting av data i fleire og færre filar, lagring av ekte data eller i ei eller anna form for numerisk representasjon, skiljing av indeksstruktur og datastruktur etc.

Modellprøvingane gav som resultat at vi i fyrstninga av juni stod att med ein modell som syntes rimeleg og brukeleg for store datamengder. Den baserte seg på MSAM-filstruktur, men alt samla i ein fil - både data og indeks, og med data i ei blanding av ekte og numerisk representasjon.

Dei prøvene vi gjorde i juni viste at plassbehovet ville vere om lag ein tredjepart av det vårt gamle KWIC-system trong, og då hadde vi data som kunne bere i seg all informasjon frå trykkeri-bandet, men lagre denne informasjonen berre ein gong og såleis gje gode høve til andre utlistingstypar enn KWIC.

Å lage ein fullstendig KWIC-konkordans av materialet ville - frå våre prøver - truleg koste om lag det same ressursmessig som i vårt 76-system, men selektive konkordansar og andre typar lister ville koste svært mykje mindre. Dellister og seleksjon er det som brukarane spør etter. Dette var også den slags ønske som DNB fyrst og fremst hadde. Dessutan ytra både dei og språkfolk sterke ønske om at systemet måtte gje høve til tagging av tekster. I systemet er også dette bygt inn, og utan at det går på kostnad av plass eller tid for oppretting av grunndata. For kvart laupande ord er der høve til å setje på fleire millionar taggar.

Eit viktig, men hittil unemnt krav for eit eventuelt nytt programopplegg var at det burde vere bra høve til oppdatering/neddatering av data. Også dette har vi greidd å passe inn i systemet utan at det har gått på kostnad av tid eller plass for oppretting av grunndata.

Av dei mindre gode sider ved systemet kan nemnast: frekvensen for eit ord kjem vi fram til ved å trikse litt under lasting av data og under produksjon. Dette skuldast den måten MSAM-filane blir bygde opp på. Vi kunne ha gjort frekvensen for kvart einskilt ord lett og direkte tilgjengeleg, men det ville ha kosta etter måten mykje i tid og plass under op-

pretting av grunndata. Vi gjekk difor frå ein slik måte og ordnar det ved litt fusk. Ein annan litt bakvend ting som for brukaren vil verke uvant er at ved utlisting vil dei orda komme fyrst som er blitt lagra sist. Dette kan vi nokolunde lettvint bøte på ved å laste grunndata på ein litt annan måte, noko som vil koste lite eller ingenting.

3. OVERSYN OVER PROGRAM-SYSTEMET

Som ein vil sjå av diagrammet i Tillegg 1 har vi funne det naturleg å dele det i to: den delen som les trykkeribånd og den delen som legg opp utdata frå trykkeribanda i den endelege MSAM-strukturen og som driv produksjon på grunnlag av MSAM-data. Fyrste delen har vi kalla CROMWELL-delen og den andre EINSTEIN-delen. Heile systemet har vi nemnt MONOLITT fordi det endelege lagringsformatet baserer seg på berre ein fil i motsetnad til dei fleste andre modellane vi laga. Data-formatet ut frå CROMWELL har vi kalla GRANAT (GRafisk ANAlysert Tekst), og lagringsformatet BRUTE (BRUks TEkst).

Noko nærare inn på namngjevinga skal vi ikkje gå her, men den ville kunne fortelje ein del om korleis vi har tenkt og arbeidd. Alle namn spring direkte ut frå idear og røynsler undervegs.

3.1. CROMWELL-DELEN

Diagrammet i Tillegg 2 syner hovudkomponentane.

Dei to faste delane er styreprogrammet CROMWELL og parsar-programmet PURITAN. I tillegg finst dei applikasjonsavhengige: FORM og grammatikken. Grammatikken fortel korleis trykkeribandet skal tolkast og FORM tek seg av skrivinga av GRANAT, dvs. kva data frå trykkeribandet som skal gå vidare til grunndata.

I CROMWELL blir løsinga av 'råteksten' styrt ved hjelp av ein kommandostruktur. I denne strukturen er det bygt inn gode høve til avlusing/debugging. I Tillegg 3 er eit oversyn over dei kommandoar som no finst i CROMWELL. Kommandoar for testing av grammatikken vantar til dels, men skal byggjast inn.

3.1.1. PURITAN og grammatikk

Systemet inneheld eit analyseprogram for 'råteksten' som t.d. kan finnast på eit trykkeriband med alt det grafiske 'rusket' som er med der. Det er dette programmet som vi har kalla PURITAN. Det er eit generelt analyseprogram, som er styrt av ein grammatikk. Det leverer frå seg dei tekststykka som grammatikken definerer. I Bibel-prosjektet er det grafiske ord. Tekststykka vert leverte med den koden som grammatikken bestemmer. Dei syntagmatiske eigenskapane som tekststykket har kan takast vare på i ein bit-streng som vert levert med kvart tekststykke frå analyseprogrammet. I Bibel-

prosjektet er tekstegenskapane til dei grafiske orda t.d.: stor bokstav, hermeteikn (føre og etter), komma og spørjeteikn.

Grammatikken er sett saman av eit tal med tilstandstabellar. Talet på tilstandstabellar, dvs. talet på paragrafar i grammatikken, vil skifte etter kor innfløkt råteksten er, og etter kva analysestrategi ein må fylgje. I Bibel-prosjektet greier det seg med godt under 100 paragrafar.

Kvar grammatikk-paragraf er sett saman av liner. Logisk sett har kvar kode i råteksten si grammatikkline i alle paragrafane. Grovt sett har ei grammatikkline fylgjande format:

1) Inn-kode
2) Inn-attributt
3) Ut-kode (=< 2 teikn)
4) Ny tilstandstabell
5) Ut-attributt

'Inn-kode' er den verdien som finst i 'råteksten', kode A. 'Inn-attributt' er eigenskapar ved denne kodeverdien som påverkar sjølve analyseprogrammet, PURITAN, t.d. varsel om retur eller at kodeverdien skal brukast som 'inn-kode' ein gong til for ein ny grammatikk-paragraf eller kontroll av 'push-down' stakk. 'Ut-kode' er den verdien (dei verdiane) som finst i 'uttekst', kode B. 'Ny tilstandstabell' er nummeret til den grammatikk-paragrafen som skal brukast av PURITAN for neste kode. 'Ut-attributt' påpverkar ikkje PURITAN, men vert levert vidare i bit-strengen som attributt til 'uttekst'.

3.1.2. Nokre eigenskapar ved grammatikkar

Grammatikken er uavhengig av programmet. Med nye grammatikkar kan vidt ulike råtekst-format analyserast utan at program treng skrivast om. Grammatikken blir lagra som ein eigen datapost i ein fil og er ein av mange mulege grammatikkar som PURITAN kan gjøre seg bruk av i same analysen.

Grammatikken er ein tilstands-automat med ymse tillegg. Kvar paragraf kan representerast på to måtar:

1) 'FULL'-tilstand, som er ein indeksert tabell.
2) 'KORT'-tilstand, som er ein søkje-tabell.

FULL/KORT er heilt ut transparent for PURITAN.

Nye grammatikk-paragrafar er 'billege' å skrive fordi det er høve til indirekte representasjon av alle felt i grammatikk-linene, bortsett frå 'inn-kode'. Programmet finn då ikkje direkte dei verdiane det skal bruke, i inneverande grammatikk-paragraf, men finn i staden opplysning om kva paragraf verdien skal stå i.

Gjennom ein 'push-down'-stakk som er styrt av inn-attributta, kan vi i grammatikken bruke grammatikk-paragrafar som subrutinar.

På grunn av 'KOPT'-tilstandar og høvet til å bruke indirekte representasjon kan grammatikken også brukast til å kjenne att strenger. I Bibel-prosjektet er denne eigenskapen brukt til å kjenne att strengene i kontroll-språket på trykkeribandet. Strengene vert kjende att ved si særlege vandring gjennom paragrafane i grammatikken. Dette gjev t.d. grammatikk-skrivaren høve til å la andre aksjonar i PURITAN vere avhengige av heile teiknsekvensar. Ved å nytte denne eigenskapen saman med mekanismen for 'push-down'-stakk kan vi få ein avgrensa parameterstyrt utmating ved t.d. å gje ei gyldig attkjenning av t.d. strengen 'AB' dersom attkjenninga tok til i paragraf C, men ikkje gyldig dersom ho tok til i t.d. paragraf D.

Gjennom 'inn-attributta' kan grammatikk-skrivaren også la ein og same kodeverdi komme att som 'inn-kode' så mange gonger som han ønskjer. Han har dermed ein kraftig ein-til-fleire mekanisme til rådvelde.

Gjennom grammatikken kan også utmatings-handsaminga vørta utsett, slik at vi i røynda oppnår ein 'look-ahead' mekanisme.

3.2. EINSTEIN-DELEN

Heile dette programsystemet er fast og gjev ikkje rom for applikasjons-programmering. Det kan nærast samanliknast med ein editor.

EINSTEIN er i store drag bygd opp som CROMWELL: styreprogram med kommando-struktur.

Eit oversyn over EINSTEIN-kommandoane er å finne i Tillegg 3.

4. SAMANFATNING

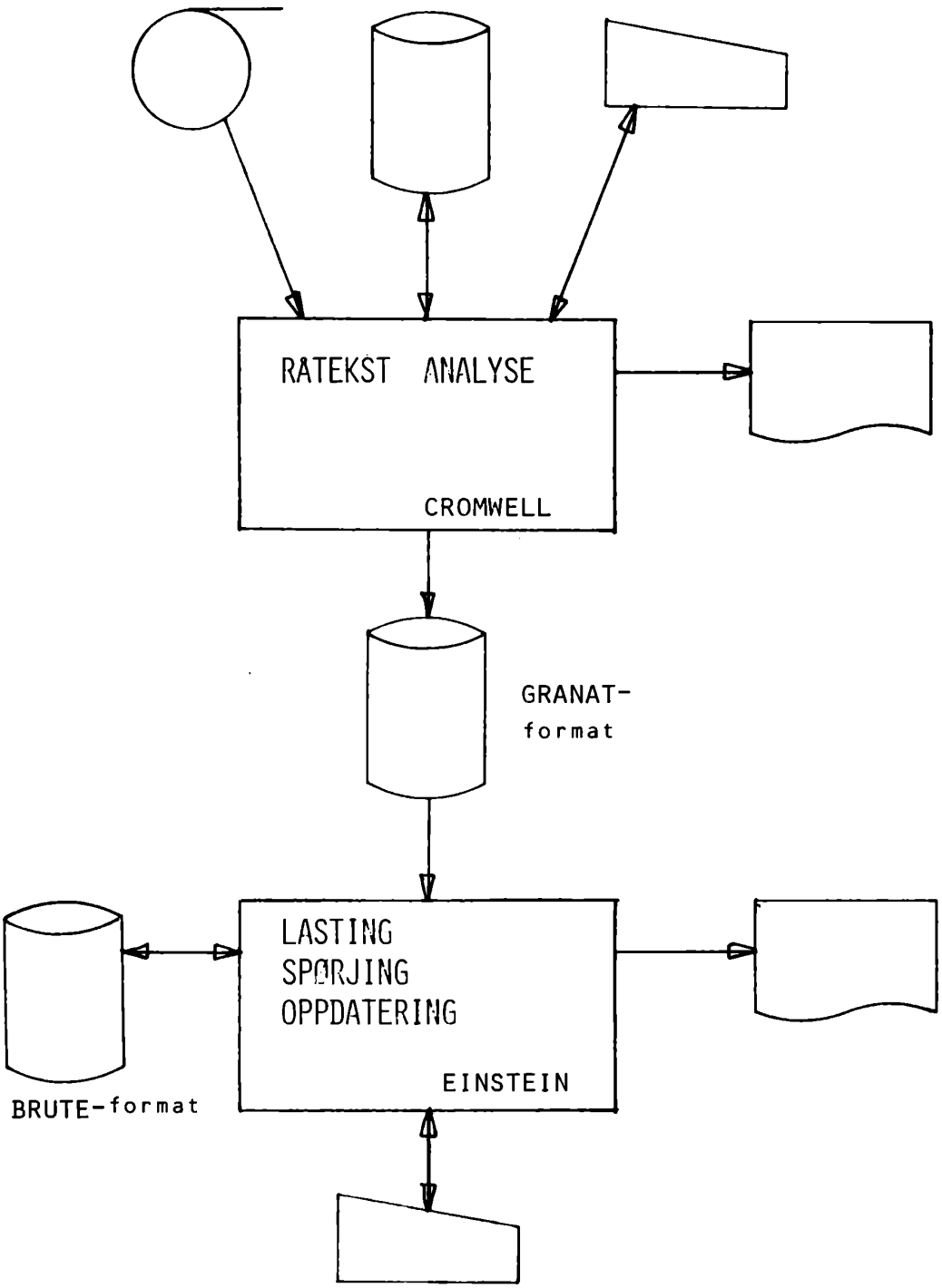
Program-systemet MONOLITT er no i full bruk i arbeidet for DNB med dei nye bibelordbøkene. Til no har vi lagt til rettes heile Det nye testamentet (bokmål). Som det går fram av Tillegg 4 er vårt nye system mindre tidkrevande og svært mykje mindre plasskrevande enn vårt gamle KWIC-system og når det dessutan har vist seg å fungere svært godt i produksjon, så meiner vi systemet tilfredsstiller ganske godt dei krav vi og andre stilte i utgangspunktet for at det skulle vere verd å ta fatt på ei nyutvikling.

Utviklingsfasen kan heller ikkje seiast å ha vore uromalt lang. Fyrste del av arbeidet byrja i april. Frå juli og i nær på to månader stod arbeidet nesten stilt på grunn av sommarferie og installasjon av ny maskin ved Universitetet. Frå september av og fram til no har det praktiske og daglege arbeidet blitt gjort av Bjørn Eide, men med regelmessige samtalar og konsultasjonar. Alt i alt kan arbeidet til no reknast å ha krevd seks-sju mann-månader, men av dei må eit par reknast som meir eller mindre spilt tid grunna dårlege og til dels urette og unøyaktige opplysningar frå trykkeriet om data og kodar på trykkeribandet, noko som førte oss på ville vegar i utforminga av grammatikken.

Mike Gillow har vore hjernen bak modellbygginga og utviklinga av det noverande opplegget medan Per-Bjørn Pedersen og Bjørn Eide har hatt hovudarbeidet med programskriving og uttesting.

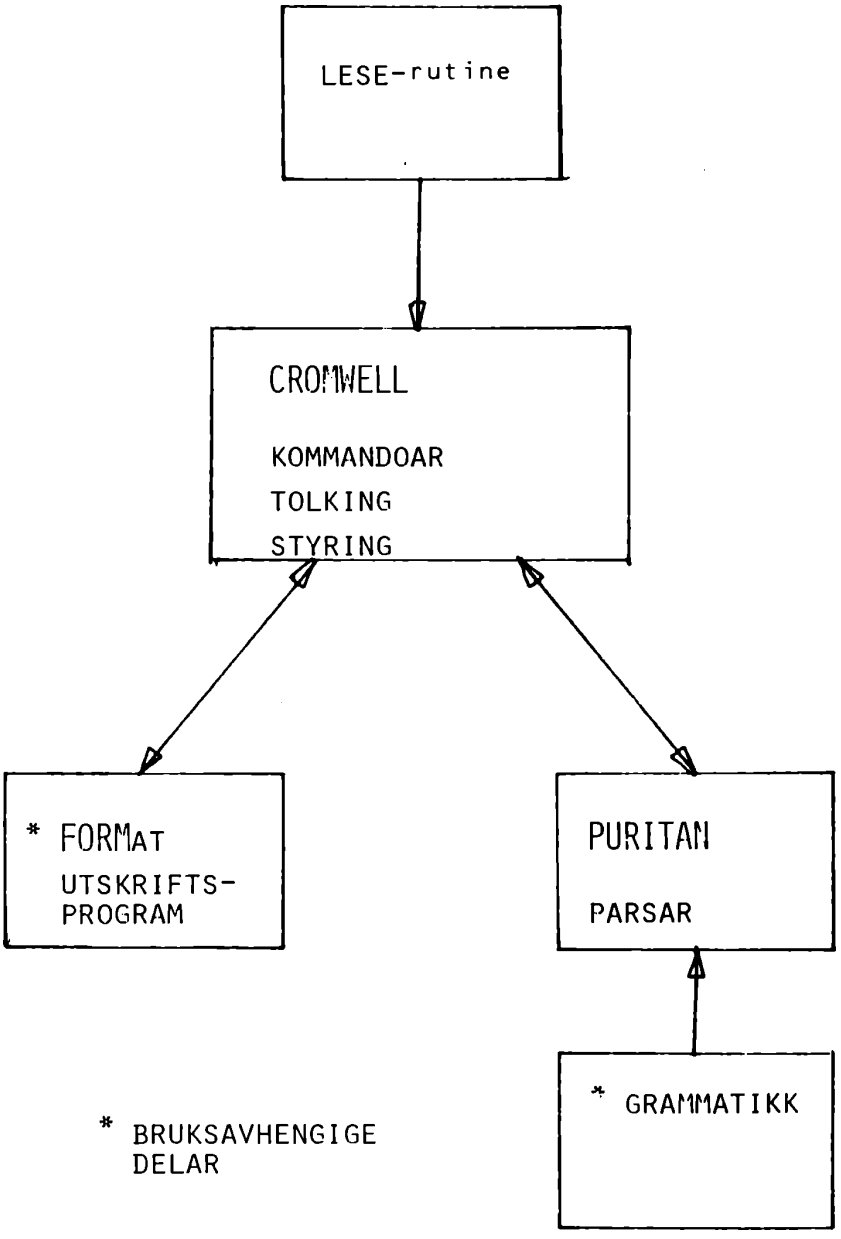
Men framfor alt må dette sjåast på som eit lagarbeid der vi alle tre har bryt våre idear og tankar mot kvarandre i seine nattetimar og tidlege morgonstunder. Ein ikkje uviktig del av vår filosofi og drivkraft for arbeidet har vore at vi alle skulle kjenne alle delar av systemet slik at vi skulle kunne bruke, utvikle og kritisere kvarandres program. Men for ein slik arbeidsmåte krevst det ein viss grad av stardardisering og felles arbeidsmetodar. Det har difor vore gledeleg og inspirerande å sjå at også slike tankar har kunna gjennomførast. Vi meiner også her å ha vandra langs vegar som ikkje er heilt vanlege, men som burde kunne gje gode og fruktbare resultat også i andre programmeringsoppgåver.

5. Tillegg 1



OVERSYN OVER KOMPONENTANE I MONOLITT

6. Tillegg 2



OVERSYN OVER KOMPONENTANE I CROMWELL

7. Tillegg 3

KOMMANDOAR I CROMWELL

INIT	initialiser TESTFIL
DDAT	lage testdata on-line (i desimalkode)
LIST	liste noverande TESTbuffer (i talformat)
SAVE	lagre buffer i TESTFIL
REST	motsett av SAVE
HIV	slette ein post i TESTFIL
FDAT	overføre røynlege datablokker til TESTFIL
STYR	nystille inn-peikar og tilstand
REF	nystille tekstreferanse
SET	styre avlusingsinformasjon
RUN	produksjon
UTOR	handsame eitt ord frå PURITAN

K O M M A N D O A R I E I N S T E I N

KWIC	lage KWIC-konkordans
FREK	lage frekvens-ordliste
VERS	liste tekstebetar
FIL	tilordnar datafilen
FORM	gjev nye verde til format-parameter
INDX	lagar ordindeks
LAST	lastar ny data

8. Tillegg 4

Samanlikning mellom vårt gamle KWIC-system og vårt nye MONOLITT-system.

Med det gamle systemet trongst det rundt 30 køyringar og kring 400 min. maskintid for å lage ein fullstendig KWIC. Denne KWICen hadde eit plassbehov på rundt 6000 spor. Med MONOLITT vil tilsvarande data få plass i litt over 2000 spor, faktisk då med ein heil del meir opplysningar lagra enn etter gamle metoden. For å laste eit heilt NT med MONOLITT trengst kring 220 min. maskintid. Då har vi rett nok ikkje ein KWIC-konkordans på bordet. Vi har enno ikkje gjort nøyaktige overslag over kva mskintid som trengst for å køyre ut ein full KWIC med MONOLITT, bl.a. av den grunn at den slags utlistingar er der sjeldan eller aldri spørsmål etter, men dei prøver vi har gjort viser at det går svært snøgt å skrive ut selektive KWICar, og det peikar i retning av at MONOLITT truleg vil vere i stand til å produsere ein full KWIC på den maskintida som står att før vi har teke ut like mykje maskinressursar som det gamle systemet.