

Softari

NOKIA DATAN

OHJELMISTOLEHTI 4/89



Tietotekniikan vuosikymmen alkaa!

Nokia Datan ohjelmistolehti

Päätoimittaja:

Terhi Ogbeide,
puh. (90) 567 2994
smail: Ogbeide—Terhi—NOK

Toimitus:

Jyrki J. J. Kasvi,
puh. (90) 567 4064
smail: Kasvi—Jyrki—FRL

SoftariTiimi:

Mikko Helander, Kari Kaipainen,
Outi Kuusela, Rami Raulas,
Tapio Sulkava, Anne Typpö,
Esko Uotila

Ulkoasu:

Jyrki J. J. Kasvi

Osoitteenmuutokset:

Takakannen palvelukortilla

Toimituksen osoite:

Nokia Data/Softari
PL 780
00101 HELSINKI

Painosmäärä:

12 000

Kirjapaino:

Helsingin Liikekirjapaino Oy

Julkaisija:

Nokia Data

ISSN 0782-8209

Sisällys:

Riittää tältä päivältä	2
SPARCstation 1 lupaa paljon	3
Valoa projektipäälikön elämään	6
Viitosten kaksintaistelu	10
High score	12
Learning Aclä — Part 6	14
SQL Windows-kurssi — osa 4	18
Mikä mihinkin ja miksi	24
Työasemasta superiin	28
Sunit verkottuvat	30
Tuoteuutiset	31
Mikkonen	32

Riittää tältä päivältä

Loppuvuodesta on nousut muotiasiaaksi löytää perheensä. Ilmiö täyttää valitettavasti kaikki muotivirtauksen tunnusmerkit eli ei aikaakaan kun väki taas unohtaa, mitä ja keitä varten he oikeastaan töitä paahtavat.

Suomalaisen atk-ammattilaisen työtahti on hirmuinen eikä tuntipalkkoja pahemmin lasketa, kun kuukausipalkkainen painaa toistakymmentä tuntia päivässä. Perhe on vain yksi kalenteriin kirjaistuista riennoista, ja lomat häiritsevät työrytmiä.

Juuri silloin, kun kaikki kulkee todella lujaa, olisi kuitenkin syytä huolestua. Ennemmin tai myöhemmin havahtuu nimittäin huomamaan, että vaikka kuinka puristaa itsestään tunteja, työ ei etene yhtään sen vauhdikkaammin — päinvastoin. Luovuus on kadonnut, ja motivaation tilalla on tottumus, mutta vaatimukset sen kun kasautuvat.

Pian adrenaliinilla ylläpidetty suorituskyky lähtee stressikäyrän mukana las-

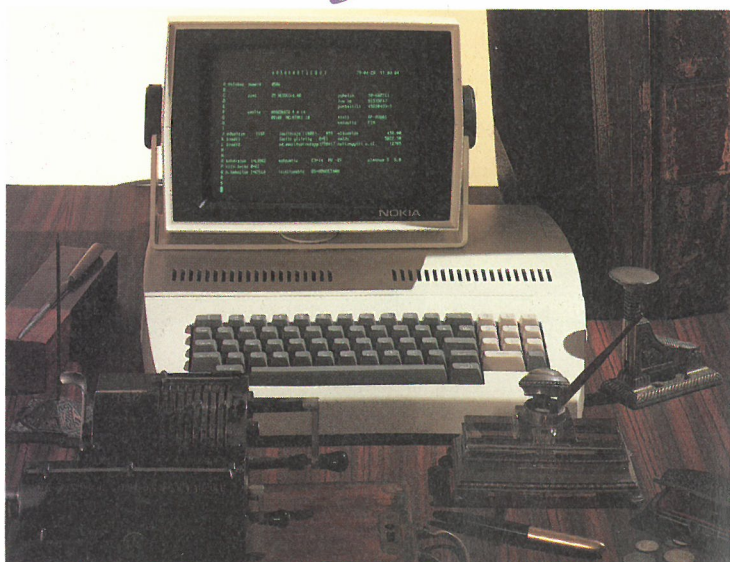
kuun. Mitä pidempään ja voimakkaampana prässä on jatkunut, sitä perusteellisempi pudotus on ja sitä kauemmin se kestää. Niin työyhteisö kuin kotiväkikin saavat tuta seuraamukset, mutta otammeko me opiksi...

Valitettavan useassa päässä asuu edelleen käsitys, että työ mitataan ajassa. Kuitenkin jos muistetaan sekä liiallisen työpaineen aiheuttamat fyysiset ja psyykkiset ongelmat että levänneiden aivojen suorituskyky, mitä mieltä on *antaa* ihmisten repiä itseään.

Kaikeksi onneksi yhä syvenevä työvoimapula on luonut alalle tekijän markkinat eli on varaa sanoa: "Riittää tältä päivältä, kotona odotetaan", ja tehdä siinä sivussa entistä parempaa tulosta kaikilla rintamilla.

Tajuammeko me tätä uraputkessamme nimeltä 42, on kokonaan toinen tarina. Ehkä olisi aiheellista pohtia tänä uutena vuotena, millä tässä elämässä oikein on merkitystä — ja elää sen mukaan.

Jyrki J. J. Kasvi



SPARC- station lupaa paljon

Pitkään on jaksettu jauhaa suppeata käskykanta käyttävien RISC-koneiden erinomaisuudesta. Puheeksi se on useimmiten jäänytkin, sillä harvinaisuutensa lisäksi RISC-koneet ovat olleet siinä määrin hinnakkaita, että tavallisen käyttäjän työpöydälle niitä ei ole pahemmin eksynyt.

Jo SPARCstation 300 on ravitsuttanut markkinoita melkoisesti. Tarjotessaan Sun 4-perheen hintaluokassa tuplasti enemmän laskentatehoa se on pannut monen laitehankinnoista päättävän pään pyörälle. Kun SPARCstation 1 tarjoaa liki yhtä paljon raakaa voimaa kolmanneksella hinnasta, ei ihme että päättä palelee.

RISC on halvempi ja nopeampi kuin CISC

Tavallaan voidaan sanoa ympyrän sulkeutuneen, sillä ensi alkuunhan kaikki tietokoneet olivat RISC-tyyppisiä (Reduced Instruction Set Computer). Vain osa mahdollisista käskysanoista merkitsi jotain. Kun kutakin uutta käskysanaa varten ei ole enää tarvinnut rakentaa huonetta täyteen elektroniputkia, käskykannat ovat kasvaneet täyttämään kaikki käytettävissä olevat bittiyhdistelmät. Niinpä suuri osa nykyisistä tietokoneista onkin CISC-tyyppisiä (Complex Instruction Set Computer).

Suppea käskykanta tarjoaa kahdenmoisia etuja. Ensinnäkin suorittimen valmistuskustannukset laskevat logiikan yksinkertaistumisen myötä. Hyvä esimerkki tällaisesta edullisesta suorittimesta on kotimikroista tuttu 6502, jota ei useinkaan osata mieltää RISC-prosessoriksi. Paljon huomattavampi etu saavutetaan, kun RISC-suorittimen suunnittelussa keskitytään nopeuteen.

Yleensä CISC-suorittimien käskykannasta käytetään jatkuvasti vain pientä osaa. Jos RISC-suorittimen suunnittelussa keskitytään vain tähän osaan kuuluviin käskyihin siten, että niiden suoritukseen kuluu vain yksi kellojakso, laskentateho kasvaa huomattavasti. Lisäksi keskeytysten hallinta helpottuu.



**Sun aloitti
RISC-koneiden
pudotuspelin**

Vaatimaton ilmiasu saa
helposti unohtamaan, että
pöydällä on 12 VAX 11/780:n
verran raakaa voimaa.

Pelkkä RISC ei kuitenkaan riitä

SPARCstation 1:n keskusyksikkönä hyrrää 20 megahertsin taajuudella kelloitettu SPARC Integer Unit (SPARC IU) 64 kilotavun virtuaalisella välimuistilla varustettuna. SPARC IU:n vaikuttavat 12,5 MIPSiä (Million Instructions Per Second) eivät kuitenkaan riitä pitkälle mitä liukulukuihin tulee. Niinpä SPARC IU:n vierestä löytyy vakiona liukulukusuoritin FPU, joka suoriutuu noin 1,4 miljoonasta liukulukuperiaatiosta sekunnissa (double Linpack MFlops).

Mikäli SPARC IU on SPARCstation 1:n sydän, 32-bittinen SBus-väylä on sen selkäranka. SPARCstation 1:n muistinosoitus on nimittäin hoidettu siten, että SBus-väylään liitettyjä laitteita voidaan käyttää suoraan virtuaaliosoitteiden kautta. Kaiken kaikkiaan väylälle voidaan liittää kolme lisäkorttia. Tieto myös liikkuu — SBus kykenee välittämään purskeina 66 megatavua sekunnissa.

Kaikki tämä ja muukin SPARCstation 1:n elektroniikka on saatu video-korttia lukuun ottamatta mahtumaan yhdelle 21,6×28,0 sentin kokoiselle piirilevyllä. Koska suurin osa piireistä on tehty erityisesti SPARC-sarjaa varten (ASIC — Application-Specific Integrated Circuit), kaikki on saatu hoitumaan 49 integroidun piirin voimin (muistipiirit poisluken).

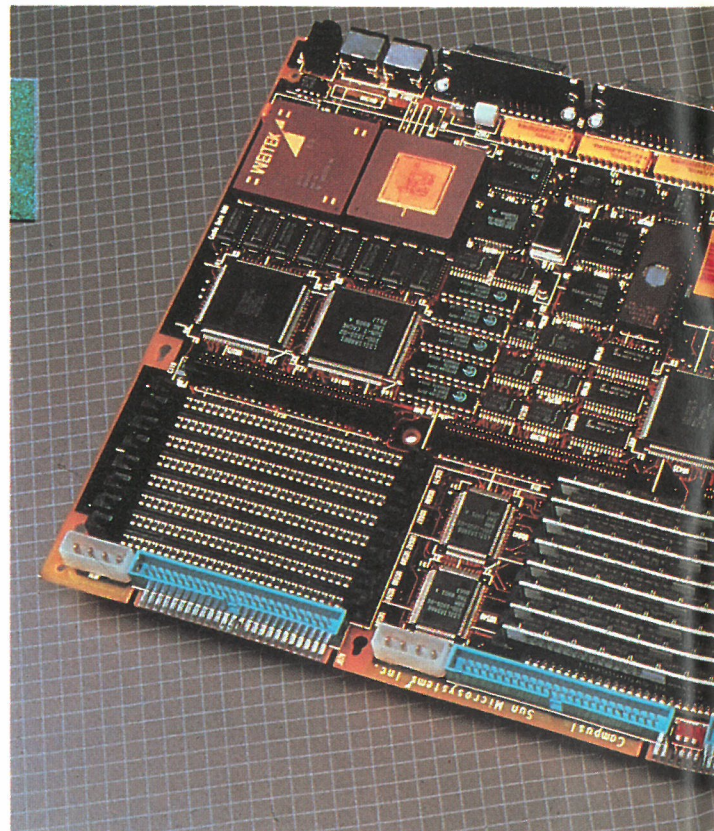
3,5 tuumaa eikä sitten muuta

SPARCstation käyttää sekä kiintolevy- että levykeasemissaan 3,5 tuuman tekniikkaa. Kuorien sisään mahtuu kaksi 105 megatavun kiintolevyasemaa, joiden haku-aika on 22 millisekuntia sekä yksi 1,44 megatavun levykeasema. Kiintolevykapasiteetti tup-

laantuu kuitenkin jatkossa oitis kun 200-megaiset asemat tulevat markkinoille.

Mikäli massamuistitarpeet eivät tästä vielä tyydyty, järjestelmään voidaan liittää uuden teollisuusstandardin mukaisen SCSI-2-liittimen kautta lisää SCSI-yhteensopivia muistilaitteita. Erityisesti SPARCstation 1:tä varten on jo valmiina ulkoinen 105 megatavun kiintolevyasema sekä 150 megatavun nauha-asema, joka syö neljännestuuman nauhoja. Myös varhaisempia SCSI-yhteensopivia Sun-oheislaitteita voidaan tuki käyttää, ne vain eivät ulkonäöltään istu SPARCstation 1:n viereen ihan sävy sävyyn...

Sun on todellakin onnistunut
survomaan kaiken tarpeellisen
21,6×28,0 sentin tilaan.



Sekunnissa syntyy 400.000 vektoria

Vanhojen 19-tuumaisten Sunin väri- ja positiivinäyttöjen lisäksi SPARCstation 1:een voidaan liittää sen oma 17-tuumainen positiivinäyttö tai Trinitronin 16-tuumainen värinäyttö. Näyttöjen erottelukyky on 1152*900 pistettä, ja näytön virkistystaajuus on 66 hertsia.

Virittämätön SPARCstation 1 piirtää 50.000—100.000 keskimäärin kymmenpisteistä vektoria sekunnissa. Jos kone varustetaan GX-grafiikkakiihdyttimellä, se piirtää kaksiulotteista grafiikkaa 400.000 vektorin sekuntivauhdilla ja kolmiulotteistakin grafiikkaa syntyy 175.000 vektoria sekunnissa.

Grafiikan lisäksi myös ääni-puoleen on panostettu. SPARCstation 1 sisältää erillisen äänijärjestelmän, joka mahdollistaa digitoitun äänen syötön ja tulostuksen.

Forth-tulkki hoitaa alustukset

SPARCstation 1:n käyttöjärjestelmänä on Sun Operation System (SunOS) viimeisin versio 4.0. Käyttöjärjestelmää on parannettu esimerkiksi syöttö- ja tulostustoimintojen osalta. Niinpä uusien oheislaitteiden asennointi onnistuu nykyään ilman koko kernelin linkkausta ynnä kääntämistä.

Käyttöjärjestelmää mielenkiintoisempi piirre löytyykin sitten alustus-PROMin puolelta. Siellä nimittäin majailee Forth-kääntäjä, joka ottaa järjestelmän hallintaansa heti käynnistyksen jälkeen. Kaikki järjestelmän alustusrutiinit on kirjoitettu Forthilla.

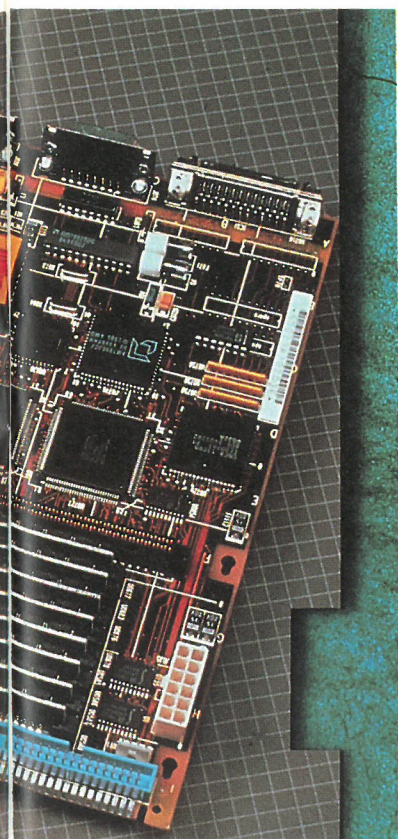
Alustusrutiinien suorituksen jälkeen järjestelmä tarkistaa vielä jokaisen SBus-korttipaikan osoitteen siltä varalta, että niistä löytyy lisää Forth-koodia. Nämä ohjelmanpätkät kertovat järjestelmälle, millaisia laajennuksia lisäkortit sisältävät, ja suorittavat korttien vaatimat alustukset. Forth-kääntäjää voidaan käyttää myös muuhun kuin alustuksiin eli esimerkiksi verkkoliitinkortti voi sisältää Forth-kielisiä rutiineja.

Huomaamaton joka pöydän työväline

Yhtenä SPARCstation 1:n suunnittelufilosofian tärkeimpänä periaatteena on ollut tehdä koneesta mahdollisimman huomaamaton monessakin mielessä. Jo vaatimaton ilmiasu saa helposti unohtamaan, että työpöydällä on kahdentoista VAX 11/780:n verran raakaa voimaa.

85 watin tehonkulutus ja alle 40 desibelin 'melu'taso pitävät omalta osaltaan huolta järjestelmän huomaamattomuudesta.

Pieni tehokulutus ja komponenttien vähyys pitävät puolestaan vikatiheyden miellyttävän matalana.



SPARCstation 1

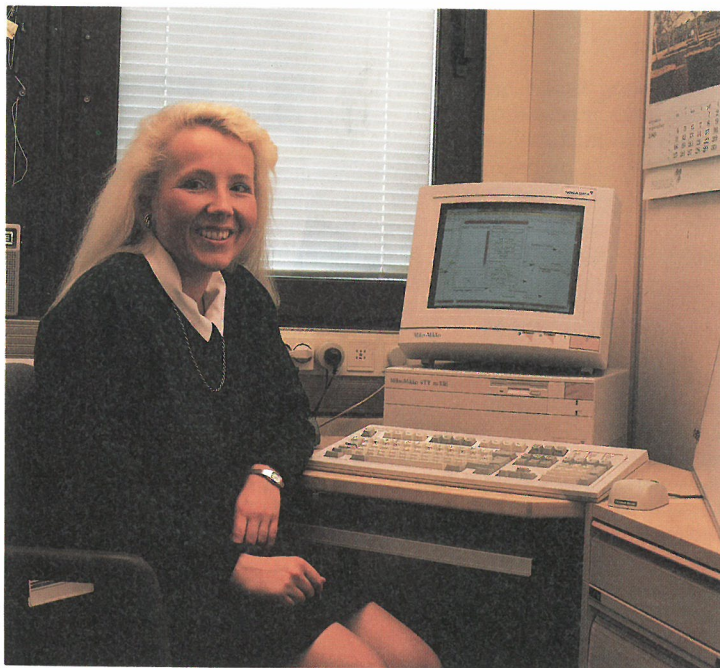
Keskusyksikkö:	SPARC IU
Liukulukusuoritin:	FPU
Kellotaajuus:	20 MHz
Suorituskyky:	12,5 MIPS Dhrystone
Liukuluvut:	1,4 MFLOPS Double Linpack
RAM-muistia:	8 Mt
maxRAM:	16 Mt
Liitännät:	Ethernet, SCSI-2, 2 sarjaporttia, Audio I/O
Massamuistit:	2 * 105 Mt kiintolevy, 3,5" / 1,44 Mt levykeasema

Valoa projektipäällikön

Nokia Planner hallitsee projektit

Nokian Tutkimuskeskus on TEKESin tuella tuomassa helpotusta projektien suunnitteluun ja seurantaan uuden Windows-ympäristössä pyörivän Nokia Planner -ohjelmiston avulla.

Esko Koskenranta



Nokia Planner 1.0 on ISO ohjelmisto — noin 50 000 riviä — ja kasvaa koko ajan, sillä kehitystyö jatkuu edelleen. Pakko ollakin, jos tarkoituksena on kyetä käsittelemään esimerkiksi vuoden mittaista ajanjaksoa parhaimmillaan 15 minuutin viipaleina, ihmisistä ja muista resursseista puhumattakaan.

Historia jakaa ohjelmistot kahtia

Projektien suunnittelu ja seuranta on työlästä ja ikävää puuhaa. Tietokoneiden ja ohjelmistojen kehityskään ei ole keventänyt taakkaa paljoakaan: alaa ovat perinteisesti hallinneet suurten keskusja minikoneiden ohjelmistot, joiden käyttö on usein raskasta ja hankalaa.

Tällä vuosikymmenellä markkinoille on tullut useita mikro-pohjaisia ohjelmistoja, joista osa

on kehitetty perinteisten ohjelmien pohjalta. Ne ovat siten perineet esikuviansa hyvät ja huonot ominaisuudet. Osa on kehitetty vastaavasti suoraan mikroille niiden ominaisuuksia hyödyntäen (esim. MS-Project: Softari 3/86).

Vaikka varsinkin suuryritysten ohjelmat (ja niistä kehitetyt mikro-versiot) tukevat teoreettisesti hienoja mallitusmenetelmiä (PERT, resurssien taseus, jne.), on niiden yleisimpänä kohtalona joutua hyllylle pölyttymään: käyttö on hyötyihin nähden liian työlästä.

Planner pyörii Windows-ympäristössä

Nokia Tutkimuskeskuksen Tietämystekniikkaosasto lähti kaksi vuotta sitten kehittämään eräiden konsernin elektroniikkayksiköiden tukemana (Mobira, Salora, Siirtojärjestelmät ja Datasiirto)

MS-Windows -pohjaista ohjelmaa, jonka keskeisenä tavoitteena oli helppokäyttöisyys. Windows-ympäristöön päädyttiin, koska ohjelman tuli toimia MikroMikko 3:ssa.

Pyrittiin siis luomaan ohjelma, jonka käyttö olisi intuitiivista ja helppoa. Projektihallinnon teoriapuolen painoarvo koettiin vähäisemmäksi. Projekti kulkee nimellä PMT (Project Management Tool), ja sen rahoitukseen on osallistunut myös TEKES sekä avustuksella että lainoituksella.

Kaksi vuotta myöhemmin ja monien vaikeuksien jälkeen Nokia Planner on saatu tuoteasteelle. Se on kirjoitettu tänä aikana vähintäänkin kertaalleen alusta uudestaan. Ensimmäiset versiot perustuivat eräaseen sinänsä hyvään tietueiden hallintaohjelmistoon, jonka nopeus ja muistin käytön tehokkuus eivät lopulta riittäneet Windows-ympäristössä.

Ohjelmisto on hyvin todennäköisesti laajin Suomessa (mahdollisesti Pohjoismaissakin) tehty Windows-sovellus (noin 50 000 riviä, 90 koodisegmenttiä ja 60 EXPORT -funktioita). Windows-ohjelmointia tuntevat arvaavat osan sen kehittämisen ongelmista, kiinnostuneet voivat kysellä lisätietoja allekirjoittaneelta. Ohjelmistoa on kehitetty objekti-orientoituneessa hengessä, vaikka se onkin kirjoitettu puhtaalla C:llä.

Näkymät ovat ikkuna projektiin

Nokia Planner tarjoaa projektin suunnittelijalle ja seuraajalle työpöydän, jonka näkymissä projektin tehtävät esitetään graafisina nuoliverkkoina (kuva 1). Planner tukee moniprojektityöskentelyä eli yhteen tiedostoon voidaan tallentaa esimerkiksi kaikki projek-

Problem-Solving Process

E. Engineer
K. Consultant

66, 00 Hours
96, 00 Hours

Functional Description

E. Engineer

20, 00 Hours

Testing

Knowledge Engineer
E. Engineer

40, 00 Hours
30, 00 Hours

elämään

tit, joilla on paljon yhteisiä resursseja. Projektien ajoitus perustuu Plannerissa kriittisen polun menetelmään (Critical Path Method).

Plannerilla projektit voidaan suunnitella hierarkkisesti, jolloin osaprojektien sisältöä voidaan tarkentaa menettämättä kokonaisuuden hallittavuutta. Projekteille voidaan määritellä omat kalenterit, joita aikataululaskenta käyttää eli tehtävien kestot yms. esitetään suoraan työaikana eikä kalenteriaikana.

Näkymien ja projektien välinen kytkeä on löyhä eli näkymään voi kuulua tehtäviä useista eri projekteista ja vastaavasti yksittäinen tehtävä voi olla esillä useilla eri näkymillä. Kerralla työpöydällä voi olla avoimena useita esimerkiksi saman projektin eri osia kuvaavia näkymiä. Työpöydällä voi olla myös tekstiraportteja tai automaattisesti päivittyviä resurssikuormituskäyrästä.

Näkymien sisällön määrää käyttäjä: hän voi luoda uusia tehtäviä ja näiden välisiä riippuvuuksia piirtämällä ne näkymään hiirellä. Eri tyyppisille tehtäville on omat symbolinsa ja esimerkiksi kriittiset tehtävät voidaan korostaa omilla väreillä.

Tehtävien yhteydessä voidaan esittää käyttäjän valitsemia esimerkiksi aikatauluun tai käytettyihin resursseihin liittyviä tietoja. Kerralla voidaan esittää neljä eri tietoa, jotka voivat olla myös listoja (esim. resurssit). Jokainen tieto voidaan esittää omalla värillä, fontillaan ja fontin koolla.

Näkymän suurennustasoa (ja näkymän yläosassa esitettyjen aika-asteikkojen määrää ja tasoa) voi helposti säätää samoin kuin symboleiden kokoa. Näkymiin voidaan liittää myös halutulla fontilla esitettävistä teksteistä, joilla voidaan kuvata vähemmän formaaleja mutta projektille olennaisia asioita. Tällaiset tekstit

voidaan sitoa tehtäviin tai ne voidaan liittää tiettyyn ajankohtaan.

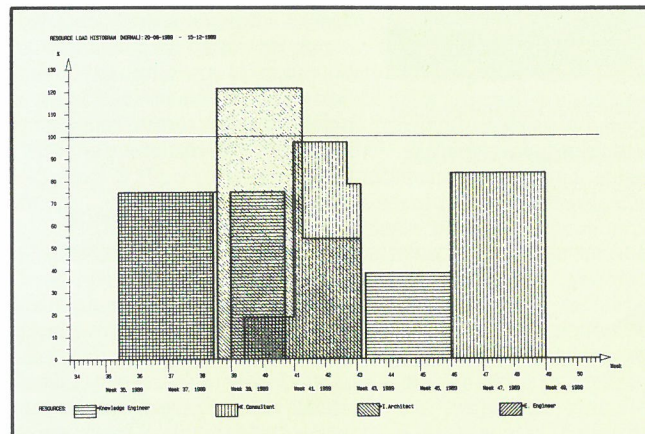
Tehtävien eteneminen esitetään näkymässä havainnollisesti eli projektin tilanne voidaan nähdä piirturille tai kirjoittimelle tulostetusta kuvasta kerralla. Musta-valkoiset päätteet ja tulostimet eivät joudu häpeään, sillä korostuksista suurin osa tulee näkyviin myös näillä.

Aikaa voi mitata eri yksiköillä

Kalenterit voidaan määritellä jopa 15 minuutin tarkkuudella. Määrittely tapahtuu luonnollisesti vuosi kerrallaan. Yksittäinen kalenteri voi olla yhtä aikaa usean eri projektin tai resurssin käytössä, joista kullakin voi olla vastavasti useita kalentereita käytössään, yksi kullekin vuodelle.

*Planner laskee
muutaman sadan
tehtävän verkon noin 10
sekunnissa eikä 3000
tehtävänkään laskenta
kestä kuin noin 30
minuuttia.*

Ilman kalentereitakin tulee toimeen käyttämällä selkeitä standardiarvomäärittelyksiä: h/p, p/vk, p/kk, p/v ja lomakausi. Tehtävien aikamäärien esitysasuja on useita, eikä viikkojen numeroita tai päivien nimiä ole unohdettu. Tehtävien kestotkin voidaan ilmaista useissa eri yksiköissä (min, h, p, vk, kk, v). Yksikköä voidaan vaihtaa (muunnos standardimäärittelysten avulla), kestot voidaan pakottaa haluttuun yksikköön tai ohjelman voidaan antaa valita 'sopiva' yksikkö tehtävien pituuden mukaan.



Resurssien seuranta mahdollistaa ennusteet

Resurssien kuormitushistogrammi piirturilla tulostettuna.

Myös resurssien hallinta tapahtuu hierarkkisesti. Yksittäiset resurssit kuuluvat resurssiluokkiin, jolloin uusien resurssien määrittely on helppoa: resurssi perii luokansa ominaisuudet. Resurssit voivat olla henkilöitä, laitteita, materiaaleja tms. Resurssien varausja kustannusyksikkönä voi olla aikayksikkö (ihmiset, koneet) tai jokin muu käyttäjän asettama yksikkö (kg, ltr tms.). Resurssien henkilökohtaisten kalentereiden avulla projektin aikataulua voidaan helposti tarkentaa resurssien kuormitusta tarkkaillaessa.

Resurssien kuormitusta voidaan seurata Plannerin avulla rinnan aikataulun suunnittelun kanssa. Kuormituksessa otetaan huomioon resurssikohtaiset kalenterit, joten ne heijastavat tarkasti todellisuutta ilman, että vapaa-aika olisi määriteltävä 'varelehtäväksi', kuten joissain ohjelmissa on tehtävä. Kerralla voidaan tarkastella viiden eri resurssin kuormituskäyriä, jotka päivittyvät automaattisesti aikataululaskennan aiheuttamien kuormitusmuutosten mukaisiksi. Kuormituskäyrät saadaan tietenkin myös tulostus-laitteelle.

Problem-Solving Process

E. Engineer
K. Consultant

66,00 Hours
96,00 Hours

Functional Surveying

I. Architect

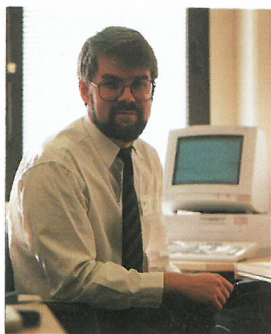
95,00 Hours

Introduction

K. Consultant

100,00 Hours

► Nokia Planner



Pekka Neva työskentelee Nokia Tutkimuskeskuksen Tietämys-tekniikkaosastolla projektipäällikkönä

Pääosa kustannussuunnittelusta ja -seurannasta perustuu tehtävien resurssien kustannuksille. Tehtäviin voidaan liittää myös muita kuin resurssien käytöstä johtuvia kustannuksia: hankinnat, vuokrat tms. Planner laskee kustannusten ja projektin etenemistilanteen perusteella projektin (osien) kokonaiskustannukset. Koska seurannan yhteydessä voidaan resurssien ja kustannusten toteumat ja arvioituid lisätarpeet kirjata, saadaan projektille myös sekä kustannusarvio että etenemistilannetta vastaava ennuste.

Raportit kertovat projektin tilan

Raportointia varten Planner antaa varsin kehittyneet välineet. Ohjelmisto tarjoaa kymmenkunta vakiomuotoista raporttia ja sisältää raporttigeneraattorin, jonka avulla käyttäjä voi laatia monipuolisia raportteja tehtävistä, resursseista, tehtävistä resurssineen tai resursseista tehtävineen.

Raportin rivien lajitteluun voidaan käyttää kolmea eri lajittelukenttää tehtäville ja resursseille erikseen. Poimintaehdoilla (ehto1, ehto1 AND ehto2 tai ehto1 OR ehto2) voidaan valita tulostettavaksi vain kussakin raportissa relevantit tehtävät ja resurssit. Poimintaehdot 'samanlainen' ja 'erilainen' osaavat myös tuttujen jokerimerkkien (*, ?) käytön.

Tieto kulkee paikasta toiseen

Planner tarjoaa jo versiossa 1.0 monipuoliset vaihtoehdot tietojen siirtämiseksi oman mikron Plannerista ulkomaailmaan ja takaisin. Graafiset raportit voidaan siirtää sellaisenaan (metafile) tiedostoon tai leikepöydälle, mistä ne voidaan siirtää edelleen muihin Windows-sovelluksiin kuten Designeriin tai jopa joihinkin Windowsin ulkopuolisiin sovelluksiin kuten Ventura 2.0:aan. Graafisista raporteista voidaan ottaa osia myös BITMAP-muodossa leikepöydälle.

Tekstiraportit voidaan siirtää sekä CSV- että ASCII-muodossa tiedostoon tai leikepöydälle, joi-

ta ne voidaan siirtää edelleen tekstinkäsittely-, julkaisu- tai taulukkolaskentaohjelmiin jatkokäsittelyyn. Lopulta itse tietosisästä voidaan poimia osajoukko toiseen Planneriin (oma tiedostorakenne) tai johonkin muuhun järjestelmään (CSV-muoto) siirtoa varten.

Planner osaa vastaavasti lukea tietoja sopivasti rakennetusta CSV-muodosta, jossa taulukon nimien sarakeotsikoiden avulla kerrotaan, mistä tiedoista on kyse. Lisäksi sillä voidaan poimia osia sillä itsellään tehdyistä muista tiedoista. Niinpä esimerkiksi resurssimääritysten siirtäminen tiedostosta toiseen on helppoa.

Paino on malleilla ja suurprojekteilla

Erityisesti on painotettu ohjelman kykyä ladata mallitietokantoja, joihin on tallennettu jokin yksikön projekteissa tyypillisesti toistuva osaprojekti. Latauksen aikana resursseiksi voidaan vaihtaa todelliset resurssit malliin tallennettujen aiempien resurssien tilalle, muuntaa osa tehtävien nimistä jne. Tämä mahdollistaa nopean projektisuunnittelun teon: osaprojektien tiedot ja niitä varten laaditut näkymät saadaan nopeasti koottua.

Toinen painopistealue on suurien projektien hallinta: osaprojektien vetäjille voidaan antaa ylätasoon suunnittelun tuottama projektitieto, jota voidaan tarkentaa ja seurata erillisissä mikroissa ja vetää myöhemmin yhteen ja saada yhteenvedoraportteja koko projektista, mikä vaatii luonnollisesti jonkinasteista kurinalaisuutta.

Opasteiden osalta Nokia Planner on ensimmäisiä Nokian ohjelmistoja, johon on integroitu Nokia Datan kehittämä ja joiltain osin tämän projektin yhteydessä laajennettu Help-järjestelmä: Text Display Manager. Plannerin käyttäjä saa niin halutessaan koko ajan opastusta valikkokomentojen merkityksestä ja ohjelman tilasta. Näkymän erilaisista tilanteista, virheilmoituksista ja dialogeista käyttäjä voi hakea tilannekohtaisia opasteita tai hän voi selailta vapaasti opastepuiden läpi.

Vain muisti rajoittaa käyttöä

Tietomäärän osalta Nokia Planner on joustava. Rajoittavana tekijänä on lähes kaikkien lukumäärien suhteen käytössä olevan muistin määrä. Ohjelmaa on olemassa kahtena versiona. Pelkkään RAM-muistiin tukeutuva versio käsittelee noin 300 — 1000 tehtävää (riippuen objektien määrästä, muista käytössä olevista sovelluksista, EMS:n määrästä yms.). Virtuaalimuistiversio (joka siirtää osan tiedoista tarvittaessa kiintolevylle) käsittelee helposti (tosin hitaammin) jopa 2000 — 3000 tehtävän tiedostoja. Käytännössä RAM-versiolla hoitaa muutaman kymmenen henkilön osaston projektien suunnittelun ja hallinnan.

Nokia Planner käyttää EMS-muistia (Softari 2/89) epäsuorasti Windowsin kautta. Ohjelmaa voi käyttää työryhmäverkossa, mutta se ei tue useiden käyttäjien rinnakkaista samojen tietojen käyttöä, joten verkkokäyttäjien on sovittava pelisäännöistä keskenään.

Nopeus on Plannerin hyviä puolia, se laskee muutaman sadan tehtävän verkon noin 10 sekunnissa eikä 3000 tehtävänkään laskenta kestä kuin noin 30 minuuttia.

Tutkimuskeskus hoitaa vielä markkinointia

Nokia Planner versio 1.0 saadaan markkinoille tammikuun kuluessa mennessä. Käyttöohjeita ja opastetekstejä lukuunottamatta tuote on lähes lopullisessa muodossaan — käyttö sujuu ilman käyttöohjeitakin. Betatestiversio on ollut käytössä Nokian sisällä ja useissa muissa yrityksissä. Ohjelman markkinointia on hoitanut pääasiassa Tutkimuskeskus itse, ja useita kauppajakin on jo tehty. Jatkossa ulkopuolinen myynti siirtynee Datalle ja muille jälleenmyyjille.

S.MAILissa Nokia Plannerilla on oma ilmoitustaulu, "Proj.hallinto Nokia Plannerilla" nro 179. Taulu on yksityinen, mutta käyttöoikeus annetaan kaikille kiinnostuneille. Keskustelu taululla

Prototyping

Knowledge Engineer

90, 00 Hours

Knowledge Acquisition

S. Expert
E. Engineer

35, 00 Hours
40, 00 Hours

K. Const

on ollut tähän asti varsin laimeaa, joten esimerkiksi ohjelmiston piirteistä tai sen jatkokehityksestä kiinnostuneet ovat tervetulleita esittämään näkemyksiään.

Plannerin soveltamis- ja kehitysmahdollisuuksia on paljon. Etsimmeikin kumppaneita sekä Nokian sisältä että muusta yritysmailmasta jatkokehityksen suuntaamiseksi entistä enemmän todellisen arkielämän tarpeisiin. Kiinnostuneille yksiköille ja yrityksille tarjotaan Nokia Planner site license -sopimusta, jolla toimipiste saa vapaan kopiointioikeuden Nokia Planneriin ja pääsee entisten jäsenten suostumuksella jäseneksi Technical Committeeen, joka päättää jatkokehityssuunnista. Nokia Plannerista kiinnostuneita pyydetään kääntymään allekirjoittaneen puoleen puh. 437 6435 tai ottamaan yhteyttä Risto Rajalaan, puh. 437 6325.

Kehitystyö ei loppu vielä vähään aikaan

Tarve siirtää tietoja erilaisten järjestelmien välillä on loppukäyttäjien yhä korostetummin esiin tuoma vaatimus. Edes jonkin ohjelmistoperheen sisäinen tiedonsiirto ei riitä, koska yritykset ovat jo aiemmin satsanneet hyvin kirjaamaan joukkoon valmisohjelmistojen, joiden yhteensopivuus on mitä on.

Tulevissa versioissa Plannerin liittymäkykyä tullaan edelleen kehittämään: Windows-ympäristössä yleistävä Dynamic Data Exchange (DDE) -yhteyksikäytäntö (Softari 4/88) sekä DIF-, dBase yms. tiedostorakenteet ovat tällöin keskeisesti esillä. Myös mahdollisuus loppukäyttäjän räätälöitävissä olevasta liittytäpinnasta hyvinkin eksotisiin (projektihallinto-, talous-, henkilöstöhallintotms.) sovelluksiin on kiinnostava kehitysmahdollisuus.

Resurssien saatavuuteen perustuva projektin ajoitus (resource scheduling) ja projekti aikatauluun perustuva resurssien käytön tasaus (resource leveling) ovat piirteitä, jotka sisältyvät useisiin markkinoilla oleviin ohjelmistoihin. Ratkaisumalli on vain usein suoraviivainen, teoriaan pohjaa-

Viime hetken lisäys

Nokia Planner -ohjelmiston markkinointi ja myynti siirtyy Nokia Datalle. Toistaiseksi Nokia Tutkimuskeskus vastaa edelleen ohjelmiston jatkokehityksestä ja yksityiskohtiin menevästä asiakastuesta.

va ja harvoin käytännön elämässä toimiva. Tämä osa-alue on jätetty 1.0-versiosta tarkoituksellisesti pois ja siihen kehitetään kokonaan uusia ratkaisuja, joissa sääntöpohjainen älykkyys ja hajautettu resurssien allokointi ovat eräitä avainsanoja.

Vuoden 1989 aikana on markkinoille tullut eräitä C++- ja OOP-ympäristöjä, joiden avulla voidaan ratkaista osa siirrettävyyssongelmista. Plannerinkin osalta harkitaan ohjelmiston siirtämistä tällaiseen ympäristöön ja sitä kautta OS/2:n Presentation Managerille (Softari 2/89) sekä myös työasemiin.

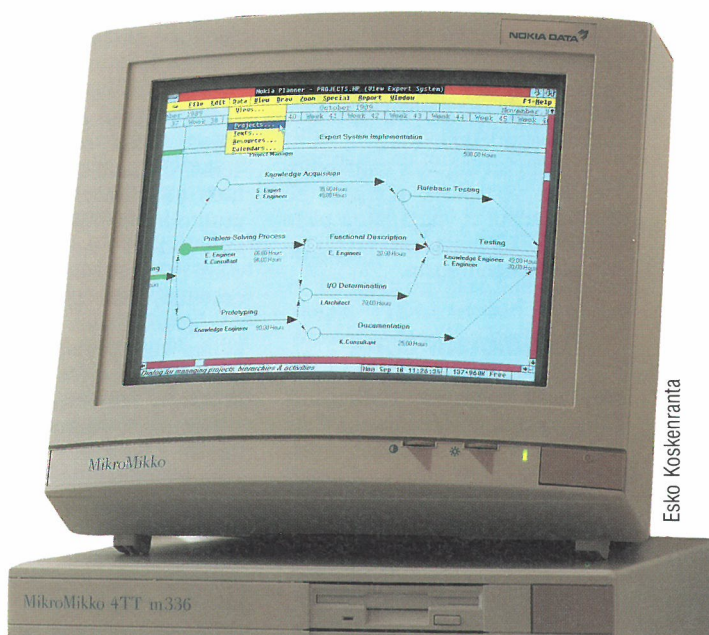
Nokia Planner on kehitetty osin myös siitä syystä, että tämän alueen sovellustuntemuksen on nähty olevan tyypillistä asiantuntijatietämystä. Eräänlaisena pitkän aikavälin tavoitteena on liittää Plannerin yhteyteen sääntöpohjaiseen tai muuhun soveltuvaan menettelyyn perustuvaa älykkyyttä, joka voisi auttaa kokemattomia projektien suunnittelijoita ja vetäjiä yksinkertaisimmillaan tarkistuslistatyypisistä ja laajemminkin suunnittelun ja seurannan vaikeuksissa.

Projektisuunnittelu ja -seuranta erillisenä järjestelmänä ei kaikissa tapauksissa saavuta riittä-

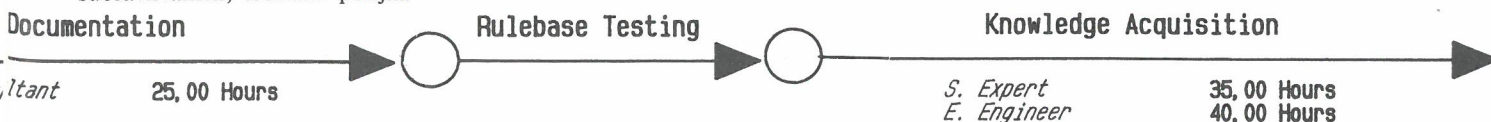
vää suosiota ja käyttöastetta. Varsinainen tuottava työkin on tehtävä, ja muut olemassa olevat järjestelmät saattavat usein sisältää (toistensa ja) projektihallinnon tietosisällön kanssa redundanttia tietoa. Tutkimuskeskuksessa onkin käynnistymässä tutkimusohjelma, jossa tuotekehitysympäristön projektitoimintaan liittyviä tietojärjestelmiä pyritään saamaan lähemmäs toisiaan.

Tavoitteena on rakentaa yhteyksiä sekä projektihallinnon että tuotteenhallinnan eri alueiden välille. Esimerkiksi projektia suunniteltaessa voitaisiin tuotteen (osien) ominaisuuksista lähtien laatia helposti järkevä projektisuunnitelma. Tehtävien tarvitsemilta resursseilta vaaditut ominaisuudet ja kyvyt voitaisiin ottaa huomioon suunniteltaessa resurssien käyttöä ja käynnistettäessä uutta projektia jo käynnissä olevien projektien rinnalle. Projektin seurannassa voitaisiin tehtävien varsinaisten tuotteiden valmistumista valvoma suoralta yhteydellä dokumenttien hallintaan. Henkilökohtainen tuntiseuranta yms. järjestelmät voitaisiin liittää esimerkiksi sähköpostin kautta projektiseurantaan. Näin lyötäisiin monta karpästä yhdellä ohjelmalla. ■

Kuva 2. Tyypillinen Planner-näkymä.



Esko Koskenranta



Viitosten kaksintaistelu

Vertailussa WordPerfect 5.0 ja MS Word 5.0



Ei ole helppoa vertailla sellaisia markkinajohtajan asemaan päässeitä tekstoreita kuin WordPerfectiä ja Wordia toisiinsa. Eroja kuitenkin löytyy, ja uutta ohjelmaa valitsevan on hyvä tuntea ne.

Vuonna 1988 tekstinkäsittelyohjelmien osuus koko MS-DOS-ohjelmistomarkkinoista oli noin 19 prosenttia (PC Week, Feb 27,89). Tästä kakusta suurimman siivun eli vajaan kolmanneksen vei WordPerfect, ja Microsoftin Word oli tiukassa tuntumassa kakkosena.

Vertailu on laadittu englanninkielisten versioiden pohjalta eli osa termeistä ei välttämättä päde ohjelmien olemassa oleviin tai tuleviin suomennoksiin.

Viimeisin sana

Noin viisi vuotta sitten Microsoft Corp. esitteli ensimmäisen PC:lle tarkoitetun tekstinkäsittelyohjelmansa. Tämä Wordin ensimmäinen versio sisälsi monia uusia ominaisuuksia, joita tuon ajan ohjelmista ei yleensä löytynyt.

5.0 on Wordin viimeisimmän version numero. Se on aiempia nopeampi, osaa hyödyntää expanded-muistia ja toimii myös verkko- ja OS/2-ympäristössä. Lisäksi se mahdollistaa tekstin ulkoasun muokkaamisen julkaisuohjelmien tapaan.

Tulostuksen esikatselu on yksi Word 5.0:n uusista piirteistä. Teksti saadaan siis näytölle lopullisen kirjoitintulostuksen tapaan muotoiltuna. Word ei kuitenkaan mahdollista WordPerfectin tapaan tulostusasetuksen tarkempaa tarkastelua, vaan näkyvissä on aina kokonainen sivu. WordPerfectilähän tulostusasetuksen tarkastaminen onnistuu myös sekä 100 % että 200 % suurennoksella.

Grafiikan ympäpäminen tekstiin on 5.0:ssa paljon helpompaa kuin aiemmissa versioissa. Kuvat saadaan mukaan Library Link-käskyn kautta taulukkolaskintiedostojen tapaan. Grafiikkatiedostot voivat olla TIFF-, HPGL-, PC Paintbrush PCX-, Windows Clipboard-, Lotus PIC-, PostScript-, encapsulated PostScript- tai Wordin omassa muodossa. 5.0 sisältää nimittäin mahdollisuuden kaapata koko näyttöruutu grafiikkatiedostoon. Erityisesti

Clipboardin kautta saadaan mukaan runsaasti eri Windows-ohjelmien avulla tuotettua grafiikkaa.

Tiloja ja toimintoja

Word 5.0:n Option-valikosta löytyy kaksi uutta toimintotilaa: Show Layout ja Show Line Breaks, jotka mahdollistavat tekstin ja sen tulostusasetuksen käsittelyn, vaikka teksti näytetäänkin paperitulosteen tapaan muotoiltuna. Tämä on erityisen kätevää kun lopputuloksen on tarkoitus olla monipalstainen. Uudet toimintotilat korvaavat aiempien versioiden Printer Display-valitsimen.

Mikä tahansa tekstin kappale voidaan ottaa Format Position-toiminnon kohteeksi. Tällainen kappale voidaan esimerkiksi ank-

kuroida sivulle johonkin tiettyyn paikkaan, muu teksti voidaan määritellä kiertämään se jne. Esimerkiksi logojen lisääminen sivujen ylälaitaan onnistuu tällä tekniikalla helposti.

Myös laatikoitu teksti on saanut uusia ominaisuuksia. Viivon mallivalikoima on laajentunut, minkä lisäksi tekstiä voidaan varjostaa eri harmausasteilla (0—100 %).

Viittaus kommenttiin

Uusi versio hallitsee myös viitemerkit ja kommentit. Viitemerkit tarkoittaa käytännössä sitä, että kirjoittaja voi määritellä tekstiin kohtia, joihin on helppo siirtyä tekstin muista osista. Viitemerkit itsessään ei tulostu, mutta sitä voidaan hyödyntää sivunumeroivissa. Jos esimerkiksi kappaleen ABC alkuun on sijoitettu viitemerkintä "page:ABC", muualla tekstissä voidaan kirjoittaa: "Lisää aiheesta ABC sivulla (Page:ABC)". Tulostusvaiheessa viittaus muutetaan todelliseksi sivunumeroksi.

Kommenteista on hyötyä erityisesti kierrätettäessä tekstejä eri henkilöillä lausuntoja varten. Lukijat voivat kirjoittaa sanottavansa suoraan tekstin oheen nimikirjaimineen kaikkineen. Word 5.0 osaa tarpeen niin vaatiessa lisätä kommentteihin myös päiväystiedot. WordPerfectistä tätä ominaisuutta ei löydy, vaan päiväykset on toteutettava makron avulla.

Rakennemuutos

Word 5.0:n sisäinen rakenne on muuttunut. Se osaa käyttää expanded-muistia, mikä mahdollistaa DOSin tuntemaa muistiavaruutta suurempien tekstien nopean käsittelyn.

Word toimii sekä MS-DOS- että OS/2-ympäristössä, vaikka se ei ole mikään Presentation Manager-sovellus. Niinpä Word on turvallinen valinta, vaikka järjestelmää oltaisiinkin muuttamassa. WordPerfect 5.0 ei sen sijaan tue kuin yhtä käyttöjärjestelmää kerrallaan eli siitä on olemassa erillinen MS-DOS- ja OS/2-versio. Se ei myöskään osaa hyödyntää työryhmäverkkoa ilman erillistä laajennusta.

Alustetut ohjeet

Wordin alustusrutiini on kehittynyt aiemmista versioista. Kaikki installoitavat kirjoittimet voidaan ottaa kerralla käyttöön ilman, että levykkeitä tarvitsee vaihtaa

joka kirjoittimen välissä. Kirjoitinvaihtoehtoja on kaikkiaan 200 kappaletta. Alustusohjelma tiedustele myös käytetyn koneen ja näyttölaitteen tyyppiä ehdotellen epävarmojen tapausten varalta yleisimpiä vaihtoehtoja.

Word 5.0:n uudet käyttöohjeet on jaettu useaksi paperikantiseksi kirjaksi, mikä ei ole välttämättä hyvä ratkaisu. Word 4.0:n ja WordPerfect 5.0:n kansioihin sijoitetut ohjeet ovat paljon helpompikäyttöisempiä. Paperikantisten kirjojen aukipitminen ja lukeminen on hankalaa mikäli samalla pitää päästä käsiksi tietokoneeseen. Lisäksi usein huomaa etsivänsä tietoa aivan väärästä kirjasta, sillä niitä on kaikkiaan neljä kappaletta.

Kirjat ovat kuitenkin hyvin kirjoitettuja ja sisältävät paljon hyvää tietoa. Etenkin vinkkejä antava ideakirja on hyvin käyttökel-poinen.

*Verrattaessa toimintoa
kohti tarvittavien
näppäilyjen määrää,
huomataan Wordin
käytön vievän
WordPerfectiä
vähemmän aikaa.*

Apua aloittelijalle

Word 5.0:n aputoiminto ei ole yhtä helpokäyttöinen kuin WordPerfect 5.0:n. Kaikilla valikoista löytyvillä online-aputeksteillä ei ole käytännössä mitään tekemistä sen asian kanssa, josta niiden pitäisi kertoa. Kaikki tieto on toki olemassa, sen esiin saaminen ei vain ole niin helppoa kuin esimerkiksi WordPerfectissä.

Aloittelijoille Word tarjoaa helpon tavan opetella ohjelman käyttöä. Käyttäjä voi käynnistää interaktiivisen opetusrutiinin mistä tahansa ohjelmaan liittyvästä aiheesta jopa kesken tekstin käsittelyn. WordPerfect 5.0 ei sisällä tällaista sisäänrakennettua opetusrutiinia, vaan sen opetusohjelma on täysin erillinen.

Aloittelijoille Word tarjoaa sekä hiirellä että näppäimistöltä hallittavan valikkopohjaisen käyttöliittymän. Kokeneemmat käyttäjät voivat ohjata teksturi-aan suoraan näppäimistöltä valikoiden ohi, mikä nopeuttaa ohjelman käyttöä melkoisesti. Todelliset asiantuntijat voivat luoda omilla makroillaan oman henkilökohtaisen käyttöliittymänsä.

Makrot mahdollistavat monimutkaisten tekstinkäsittelyrutiinien automatisoimisen. Word sisältää myös joitakin valmiiksi ohjelmoituja makroja. Koska Word 5.0:n valikkojärjestystä on muutettu, Word 4.0:n makrot on päivitettävä ennen niiden käyttöä 5.0:ssa. Word 5.0 sisältää onneksi päivitysrutiinin, jolla homma hoituu ilman ongelmia.

Kumpi onmpi parempi?

Wordin ja WordPerfectin käyttöliittymät ovat täysin erilaiset. WordPerfect on tarkoitettu käytettäväksi pääasiassa näppäimistöltä (vaikka valikoiden ohjaus hiirellä on myös mahdollista). Mitään päävalikkoa ei yksinkertaisesti ole olemassa. Sen virkaa hoitavat toimintonäppäimet.

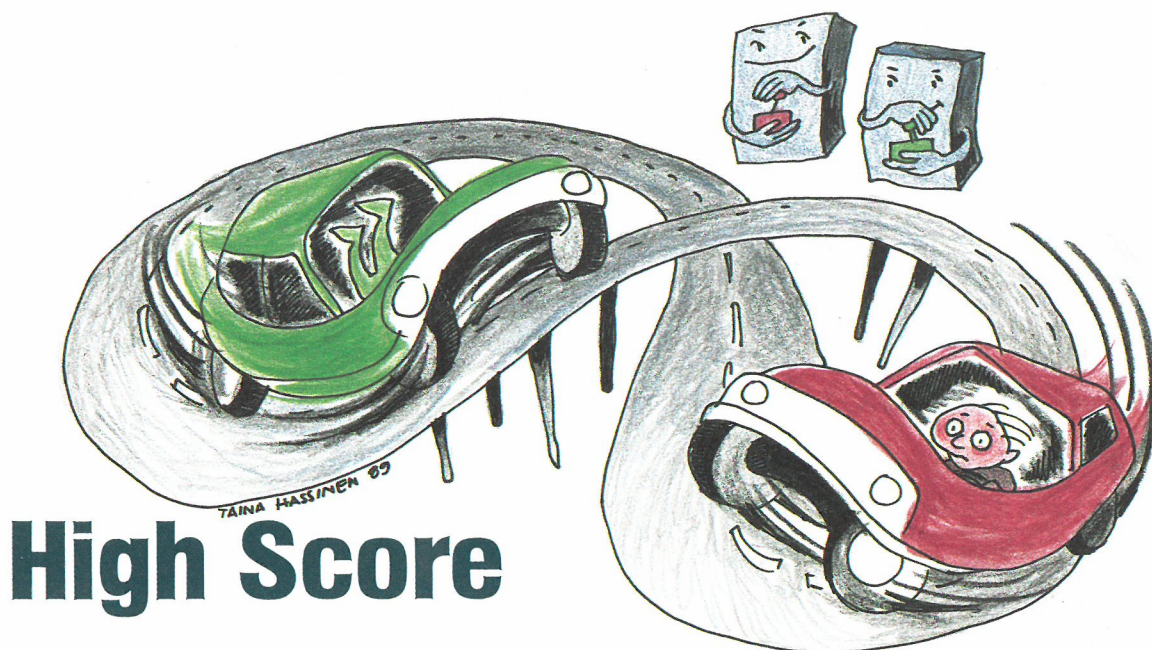
MS Wordin ohjaus onnistuu sen sijaan kahdella eri tavalla. Joko hiirellä tai näppäimistöltä. Wordista löytyy myös päävalikko kehotteineen. Omasta mielestäni Wordin valikoiden käyttö on helpompaa kuin WordPerfectin, jossa käyttäjän täytyy muistaa asioita ulkoa. Myös verrattaessa toimintoa kohti tarvittavien näppäilyjen määrää, huomataan Wordin käytön vievän WordPerfectiä vähemmän aikaa.

Yleisesti voidaan kuitenkin sanoa, että kummallakin ohjelmalla on omat etunsa ja haittansa, vaikka ne sisältävätkin lähes samat ominaisuudet. Se, kumpaa ohjelmaa kukin pitää parempana, on hyvin subjektiivinen kysymys. Tärkein tekijä on tietysti kokemus eli jos käyttäjällä on vuosien tuntuma jompaankumpaan, hän todennäköisesti valitsee sen myös tulevaisuudessa.

Samat sanat voidaan sanoa myös Perfect- ja Windows-tuoteperheistä eli mikäli käyttäjällä on kokemusta Perfect-tuoteperheestä, WordPerfect on häntä varten — ja sama Windows-perheelle ja Wordille.

Eli kuka on oikea henkilö sanoamaan: "Tämä teksturi on ehdottomasti paras"? Olisiko käyttäjä. . .

Ominaisuus	Word	WP
Sisäänrakennettu opetus	on	ei
Esikatselun tarkennus	ei	on
Ääriiviivänäkymä	on	ei
Grafiikan pyöritys	ei	on
Älykäs tulostus (merkistön vaihto)	on	on
Laatikoitu teksti näkyvissä	on	ei
Teksti saadaan kiertämään kuvat	on	on



High Score

Harri Kajava hyräili tyytyväisenä erään vanhan iskelmän melodiaa ja naputteli tahtia autonsa kojelautaan. Vekkuli keksintö tuo tietokoneohjattu kaupunkiliikenne. Kun iso keskustietokone ohjaili keskitetysti koko Helsingin moottoriajoneuvoliikennettä ja kevyt liikenne oli siirretty kokonaan omille väylilleen, oli työmatkasta tullut mukava rentoutumishetki ennen kiireisen työpäivän alkamista. Jonot liikkui nykyään paljon nopeammin, ja silti turvallisemmin. Harri ei muistanut oliko liikenteen tietokoneistamisen jälkeen Helsingin keskustassa tapahtunut enää yhtään onnettomuutta. Tietokoneella oli oma voimanlähteensä, moninkertaisesti varmistettu ja riippumaton kaupungin muusta sähköjakelusta. Vikatilanteiden varalta keskustietokoneesta oli olemassa täydellinen kopio, joka voitiin tarvittaessa laittaa remmiin sekunnin kymmenesosassa.

Harri yritti kuvitella, miltä Mannerheimintien ja Postikadun risteys näytti tietokoneen näkökulmasta; sen silminä toimivat lukuisat videokamerat ja magneettiset ilmaisimet kadun yläpuolella ja sivuilla, tuntoaistina paineanturit kadunpinnassa, sekä tasapainoelimiä kaarto- ja kallistusmittarit jokaisessa autossa. Ei sitä tietotulvaa kukaan ihminen kestäisi, mutta nämä uudet hermostokotietokoneet pystyivät helposti ja nopeasti käsittelemään suurempiakin tietomääriä. Olihan kaikkien maailman suurkaupunkien keskustaliikenne jo tietokoneistettu.

Vieläköhän ehtisi pirauttaa Liisalle? Siitä oli kohta kolme viikkoa kun viimeksi käytiin ulkona. Ensimmäinen oli ollut se Lontoon matka ja sitten sen takia myöhästynyt Siilinjärven projekti. Mahtikoihan tyttö olla vihainen, kun ei ollut edes soittanut?

— Liisa Hakulinen.

— Harri täällä, hei. Enhän herättänyt?

— Et herättänyt, olin juuri keittelemässä aamukahvia. Sä soitat autosta?

— Joo, mä oon Mannerheimintiellä, menossa töihin. Mitä sulle kuuluu?

— Ihan hyvää. Kävin eilen katsomassa äitiäni Lahdessa. Muuten olen ollut ihan kotosalla. Mullahan on vielä kaks viikkoa lomaa.

— Kuule, tuli tässä mieleen, kiinnostaisko sua lähteä illalla Espan lavalle tanssimaan, kun on niin upea ilmakehä?

— En oikein tiedä, kun mun sisko tulee...

— Mitä ihm... Voi ei! ... Se oli lähellä!

— Mitä se oli? Onko jokin hullusti?

— En tajua, tää auto, se menee ainakin sataa... Hei, tuohan tulee päälle!

— Oletko kunnossa? Sano jotain!

— Hengissä ollaan, ainakin vielä... Ei, ei tuosta välistä, ei siitä mahdu! ... Mahtui sentään, mutta oli peilit mennä. Se liikennetietokone on ihan seko! ... Ja taas menttiin! Tää on ihan kuin mistäkin amerikkalaisesta leffasta!

— Mitä ihmettä siellä tapahtuu? Keskustietokone seonnut?

Miksei ne heitä varakonetta päälle?

— Sitä mäkin ihmettelen! Tää on aivan sairasta menoa! Mittari näyttää sataakahtakymmentä ja oikealta on tulossa iso Volvo! ... Ehdittiin, hiuskarvan mitalla pois alta! Sen Volvon kuski tais pyöryä.

— Anna sille koneelle käsky, että ajaa sut lähimpään parkkiin! Sehän tappaa sut vielä!

— Hyvä idea! Odotas, tässä on ihan lähellä parkkitalo — jollei tää ehdi jo ohi siitä. — Noin. Hei, se ei ota komentoa vastaan! Se hullu kone ei huoli mitään komentoja! Ja nyt tää auto kääntyy vastaan tulevien kaistalle! Ei, nyt palattiin taas omalle.

— Laupias taivas! Soita liikennevalvontaan! Mutta soita sitten heti uudestaan mulle!

— Okei! Älä huoli, kyllä tästä selvität...ään. Hei.

— Hei.

Vapisevin käsin Harri paineli liikenteenvalvonnan numeron. Jonotusnumero 179 ja klassista musiikkia. Joku muukin halusi valittaa hulluksi tulleen liikennetietokoneesta.

Nyt sieltä tuli joku ainakin sataa vasemmalta! Harrin auto kurtasi renkaat ulvoen oikealle ja hän löi korvansa kipeästi ikkunaan. Samassa hän heilahti rajusti turvavöitä vasten auton tehdessä äkkijarrutuksen ja vihdoinkin pysähtyessä.

Kun Harri lopulta tajusi auton olevan pysähdyksissä, hän avasi turvavyönsä ja ryntäsi autosta ulos, ettei se ehtisi lähteä uudelleen liikkeelle. Vasta ulkona hän tajusi olevansa työpaikkansa pi-



Merja Sippola on 26-vuotias rakentajateekkari, joka opiskelee yhtenä syventymiskohteena tietojenkäsittelyä.

halla. Vieressä oli toinen auto, jonka kuljettaja juuri kömpi autosta ulos, yhtä kauhun kangistamana kuin hänkin. Harri tunnisti työtoverinsa Pertin.

— Tekikö se mielipuolet kone sulle saman tempun, Harri uteli.

— Siis sinullekin? vastasi Pertti. — Minä olin jäädä rekan alle Uudenmaankadun tunnelista tullessani!

— Ja mä sekä kuorma-auton että ison Volvon Mannerheimintiellä! Sen Volvon kuski pyörtyi!

— Tauti, kun laukaisi migreenikohtauksen! Pertti hieroi ohimoi- taan.

Äkkiä Harri muisti, että oli lu- vannut soittaa Liisalle.

— Mun täytyy mennä soitta- maan, hän huikkasi juostessaan pääovelle. Autoon hän ei ihan he- ti astuisi.



Tapani Matala juoksi loppu- metrit pomon ovelle ja astui ko- puttamatta sisään.

— Mitä nyt on tapahtunut, kun lomilta hälytitte?

— Se sinun lemmikkisi, liiken- teen keskustietokone. Se on tullut aivan hulluksi. Täällä on ollut langat kuumana koko aamun. Kone on kuulemma ohjannut kuin humalainen: mieletöntä pu- jottelua yli satasen nopeuksilla keskellä Helsinkiä.

— Onko tapahtunut onnetto- muuksia? Mihin varakone unoh- tui?

— Varakone meni päälle nor- maalisti, mutta se jatkoi samaa ruljanssia! Ei, onneksi ei ole syn- tynyt muutamaa mustelmaa ja hermoromahdusta pahempia on- nettomuksia, ja nyt kone käyt- täytyy taas ihan säädyllisesti, mutta mikään ei takaa, ettei se aloita kohta uudestaan.

— Mitä se sitten oikeastaan teki?

— Siitä on vähän vaikea saada selkoa. Puhelinsoitoista ei saa irti päätä eikä häntää, eikä kone suostu vastaamaan mihinkään kysymyksiin, jotka koskevat sen tänäaamuja toilailuja.

— Ja minun on nyhdyttävä se tieto ulos?

— Mistä arvasit? Painu hom- miin, ja pidäkin kiirettä! Minun niskassani on jo puoli Helsinkiä aina presidentistä asti. Se pahuk- sen kone pysäytti presidentin saattueen äkkijarrutuksella, ja Ruotsin pääministeri kolautti päänsä.

— Olen menossa jo.



— No niin, Alfa, annahan tula, mitä sellaista tämän aamun lii- kenteessä tapahtui mitä ei muina aamuina ole sinun ohjattessasi ta- pahtunut?

— Eräs punainen Lada kävi puolitoista senttimetriä lähem- pänä reunakiveä kuin koskaan aiemmin, siis minun ohjauksessa- ni. Betan ohjauksessa tuo ennätys on kyllä kerran ylitettykin puolel- la senttimetrillä.

— Tarkoitin sellaista, millä on jotain merkitystä autoissa mat- kustavien ihmisten turvallisuus- dille tai viihtyvyydelle tai molem- mille, ja mihin sinä voit vaikut- ta!

Äläkä katso minua noin.

Se on hardiskavereiden moka, ei minun.

— Säädin Uudenmaankadun tunnelin ilmanpuhaltimet uudella tavalla, nyt tuuletus on Heinisen mitan mukaan noin puoli pro- senttia parempi. Lisäksi käytin niitä uusia antureita Albertinka- dulla. Ne toimivat mainiosti; Al- bertinkadun liikenne etenee nyt keskimäärin 0.003 prosenttia no- peammin, kun voin optimoida paremmin kuin ennen. Vielä...

— Tiedät, että nuo ovat nor- maaliin päivärutiiniin kuuluvia asioita, lopeta niiden latelu ja esi- tä ne päiväraportissa! Kerro, olet- ko ohjannut autoja alle kahden- kymmenen senttimetrin etäisyy- deltä toisten autojen keulan edit- se? Entä oletko kiihdyttänyt nii- den nopeuden yli 80 kilometriin tunnissa? Oletko toiminnallasi välittömästi tai välillisesti aikaan- saanut sen, että autojen matkus- tajat — ihmiset — ovat kolhineet päänsä sivuikkunoihin tai etuis- tuimen niskatukiin?! Oletko ha- vainnut autoissa istuvien pyörty- neen?! Vastaa!

Alfa ei sanonut sanaakaan.



— No, selvisikö?
— Ei! Beta ei puhu mitään, ja Alfa pitää mykkäkoulua.

— Hemmetti!

— Olen sitä mieltä, että ajetaan

liikenne seis ja otetaan molem- mista muistianalyysi. Haluan tie- tää, mitä niiden sisuskaluissa on tänä aamuna liikkunut.

— Satutko tietämään, mitä se maksaa?

— Kyllä, mansikoita, mutta tässä on koko kaupungin turvalli- suudesta kyse. Sitä paitsi liikenne on jo osittain seis, sillä monet ei- vät yksinkertaisesti uskalla astua autoon.

— Hyvä on, kai se on tehtävä. Mutta eikö se pahuksen kone voi sotkea muistijälkiänsä ja salata kaikkea mitä tapahtui?

— Ei. Minä pidän persoonalli- sista tietokoneista, mutta koska tiedän millaisia ne voivat olla, en luota niihin liikaa. Mekanismi, joka ottaa koneen muistijäljet, on suojattu koneelta itseltään. Vali- tettavasti niitä tietoja ei saa ulos kaatamatta konetta. Äläkä katso minua noin. Se on hardiskaverei- den moka, ei minun.



Tapani Matala astui jälleen esi- miehensä huoneeseen. Hänen kainalossaan oli paperirulla ja kasvoilla kummallinen ilme.

— No, joko se selvisi?! Matti Hyrkäs oli yleensä perin rauhalli- nen mies, mutta nyt hänen her- monsa olivat ratkeamispisteessä. Hänet oli tänään jo kolmasti ero- tettu. Onneksi erottajista kukaan ei ollut hänen esimiehensä.

— Selvisihän se.

— No, mitä tapahtui?

— Katsohan. Kuusitoista autoa käyttäytyi tänään omituisesti. Olen piirtänyt tähän niiden reitit ja merkinnyt viereen nopeudet se- kä risteäviltä kaduilta tulevien ja ohitettujen autojen pienimmät etäisyydet niihin.

— No, voi sun halvattu! Sehän on kuin mikäkin formularalli!

— Jep. Kaikki kuusitoista au- toa painelivat yli sadan kilomet- rin tuntinopeudella Helsingin halki. Lyhin välimatka risteävältä kadulta tulevaan autoon oli 3 cm. Autot liikkuvat eri reittejä, muo- dostaen kuitenkin kahdeksan kil- pailevaa paria. Perille ne saapu- vat vain vajaan sekunnin eroilla.

— Mitä hiivattia se kone oikein kuvitteli tekevänsä?! Voisi luulla että se on katsellut liikaa poliisi- sarjoja holotelkusta.

— Ei aivan. Selitys on vieläkin yksinkertaisempi: Alfa ja Beta vain pelasivat keskenään videope- liä aikansa kuluksi. Alfa voitti, puolentoista sekunnin erolla. ■

Learning Ada

Part 6 — Generic Parameters

In this course we have dealt with data abstraction and operational abstraction. In Part 6 we introduce a way of forming general templates for these abstractions. Input/output is also described in greater detail.

In designing a software system, we often have to write packages or subprograms that are similar in purpose. Take sorting of the contents of an array as an example. This operation can easily be provided with a subprogram — say as follows:

```
type NAME__ARRAY__TYPE is
  array (INTEGER range < >) NAME__TYPE;

procedure SORT
  (NAMES: in out NAME__ARRAY__TYPE);
```

To sort an array containing some other type of components, we need a separate subprogram. The strong typing rules of Ada require us to specify the type of every object at compilation time. So it seems we are forced to write an additional procedure that is identical except for the parameter type. This is inefficient, because it involves creating separate subprograms to process objects of different types, even if the algorithm is the same for every type.

There is a solution: Ada permits generic packages and subprograms that define templates which combined with generic parameters, enable the templates to be tailored to specific needs. Generic units resemble the macro facility found in many lower level languages. The difference is that a generic unit is checked and compiled only once, whereas a conventional macro is only a piece of text that has to be recompiled every time it is used.

Since generic packages and subprograms are only templates, they cannot be executed; we cannot use them directly. Instead, we must first **instantiate** the generic package or subprogram. After that we can use the resultant program unit as if it were an ordinary unit.

To write a generic package or subprogram, we simply take the specification and prefix it with a generic part that defines the **generic parameters**. For example, we could start out from the following subprogram for interchanging two integers:

```
procedure
  SWAP__INTEGERS (A, B: in out INTEGER) is
  TEMPORARY: INTEGER := A;
begin
  A := B;
  B := TEMPORARY;
end SWAP__INTEGERS;
```

If we need to swap other types of elements later on, it is not necessary to create a new subprogram for every type. As it is clear that the type of the values being swapped does not affect the logic in any way, we can add the following generic part to the procedure specification:

```
generic
  type ELEMENT is private;
  procedure SWAP (A, B: in out ELEMENT);
```

The corresponding body of the subprogram is:

```
procedure
  SWAP (A, B: in out ELEMENT) is
  TEMPORARY: ELEMENT := A;
begin
  A := B;
  B := TEMPORARY;
end SWAP;
```

This procedure is the same as the SWAP__INTEGERS subprogram except for the data types. Note that in constructing a generic subprogram or package, we always have to write the specification as a separate unit. The word **generic** cannot appear at the beginning of a body.

The generic procedure SWAP is the template, from which the actual procedures can be instantiated. We have to define the data type in each case. To create a SWAP__CHARACTERS procedure for example, we need to write:

```
procedure SWAP__CHARACTERS is
  new SWAP (CHARACTER);
```

It is not necessary to repeat the subprogram body. The compiler simply substitutes the identifier CHARACTER for the identifier ELEMENT wherever it occurs in the body of the subprogram.

Note: the type CHARACTER is a parameter associated with the instantiation process. The resulting procedure SWAP__CHARACTERS has two **in out** parameters A and B of the CHARACTER type.

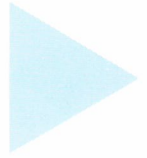
In addition to generic subprograms, there are also generic packages. Typical examples include stack and queue packages, where the type of the items in a data structure is parameterized by generics.

Generic Type Parameters

Data types are the most common class of generic parameters.

To check the validity of an instantiation, the Ada compiler must have some information on the types that can be used in different kinds of instantiations.

The keyword **private** in the previous example tells the compiler that any type that permits assignment and testing for equality can be used as an actual parameter. Writing **limited private** instead of **private** allows any type to be a generic parameter, but in that



Using a generic type in the body of a generic unit, we can utilize all the operations permitted by the formal type definitions.

case the operations that are possible within the generic unit really are limited. Other possibilities include **access** to pointers, (**<>**) for discrete (enumeration or integer) types, **range** **<>** for integer types, **digits** **<>** for floating point types, and **delta** **<>** for fixed point types.

To take an example:

```
generic
  type T is (<>);

function
  NEXT (X : T) return T;

function NEXT (X : T) return T is
begin
  if X = T'LAST then
    return T'FIRST;
  else
    return T'SUCC(X);
  end if;
end NEXT;
```

Using a generic type in the body of a generic unit, we can utilize all the operations permitted by the formal type definition. Since all the Ada discrete types have **FIRST**, **LAST** and **SUCC** attributes we can safely use them in the body of the function.

We can then instantiate new functions

```
type DAY__TYPE is
  (MON, TUE, WED, THU, FRI, SAT, SUN);
subtype LETTER__TYPE is
  CHARACTER range 'A' .. 'Z';

function
  TOMORROW is new NEXT (DAY__TYPE);
function NEXT__LETTER is
  new NEXT (LETTER__TYPE);
function
  NOT__OPERATION is new NEXT (BOOLEAN);

-- TOMORROW (SUN) = MON
-- NOT__OPERATION (TRUE) = FALSE
```

Array types, too can be used as generic type parameters. For practical purposes we usually have to give the index and component types as generic parameters, too. A typical example is:

```
generic
  type ITEM is private;
  type INDEX is (<>);
```

```
type
  VECTOR is array (INDEX range <>) of ITEM;
package ITEM__STORE is
...
end ITEM__STORE;
```

In this case the array type given as the third actual parameter must be a one-dimensional unconstrained array type. Its index type must be given as the second parameter, and its component type as the first parameter.

```
type INT__ARRAY__TYPE is
  array (POSITIVE range <>) of INTEGER;

package INT__STORE is
  new ITEM__STORE
    (INTEGER, POSITIVE, INT__ARRAY__TYPE);
```

Generic Object Parameters

Generic parameters can also be variables and constants. The form of their declarations is like that of a variable, but an extra keyword is included for each generic parameter: **in** **out** for variables, and **in** for constants (default).

When writing a terminal package, we can fix the screen size with two generic constants:

```
generic
  ROWS    : in INTEGER := 24;
  COLUMNS: in INTEGER := 80;
package TERMINAL is ...
```

Note that we can use default values for generic parameters. We can then create a normal terminal package (screen size 24×80) by writing:

```
package NORMAL__TERMINAL is new TERMINAL;
```

or we can create a terminal with a large screen size by writing:

```
package LARGE__TERMINAL is
  new TERMINAL (66, 132);
```

Generic Subprogram Parameters

Ada also enables subprograms to be transmitted as parameters to generic units. For instance, we can define the **SEND** and **RECEIVE** procedures outside a terminal package so as to implement different kinds of protocols. We can then treat **SEND** and **RECEIVE** as generic subprogram parameters:

```
generic
  ROWS    : in INTEGER := 24;
  COLUMNS: in INTEGER := 80;
  with procedure SEND (VALUE: in CHARACTER);
  with procedure RECEIVE (VALUE: out CHARACTER);

package TERMINAL is ...
```


Ada

On instantiating a new terminal, we have to specify the SEND and RECEIVE procedures. For example:

```
procedure MICRO_SEND (V: in
  CHARACTER);
procedure MICRO_RECEIVE (V: out
  CHARACTER);

package MICRO_TERMINAL is new TERMINAL
  (ROWS => 24,
   COLUMNS => 40,
   SEND => MICRO_SEND,
   RECEIVE => MICRO_RECEIVE);
```

Here we have used a named parameter association instead of a positional association, as we have been doing so far.

Defaults can be used for generic subprogram parameters. Here we will skip the details, and give only a typical example of a queue package in which the maximum queue length, element type, and output procedure are generic parameters.

```
generic
  MAX: INTEGER := 100; -- Default max length
  type ELEMENT_TYPE is private
  with procedure OUTPUT is DEFAULT_OUTPUT;
package GENERIC_QUEUE_STRUCTURE is ...

package STRING_QUEUE is
  new GENERIC_QUEUE_STRUCTURE (STRING);
```

By default, the procedure DEFAULT_OUTPUT is used in the package body to print the contents of one queue element.

In Ada, non-generic subprograms cannot have parameters that are subprograms. Subprograms can only be parameters of generic units. This, too, ensures that all type checking can be done at compilation time.

Using generic type parameters is a kind of a shorthand that reduces inconvenience and mistakes in editing the program text. The possibility of using subprograms as generic parameters gives us a very powerful programming tool because generic units can behave in very different ways depending on the actual parameters. This facility must not be misused, because it is liable to confuse the program logic.

Generic packages can be made very "general-purpose" by parameterization, so they are good candidates for reusable program components.

Input and Output

So far we have used only the TEXT_IO package for I/O in our program examples. As we have seen, TEXT_IO is used for reading and writing characters and character strings. It also contains the generic subpackages INTEGER_IO, FLOAT_IO, FIXED_IO, and ENUMERATION_IO for the corresponding data types.

Here we also present the basics of two other predefined I/O packages, which handle values of any type: SEQUENTIAL_IO for sequential files and DIRECT_IO for random access files. These packages, too, are generic.

Subpackages of TEXT_IO

Since the subpackages of TEXT_IO mentioned above are generic, they cannot be used directly. The actual packages must first be instantiated. To be able to read or write numeric values, we have to instantiate packages for every different type. For example,

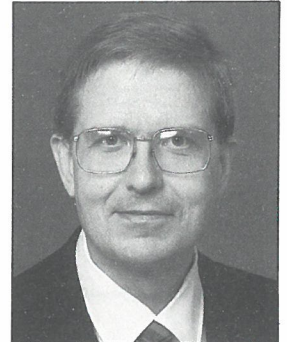
```
with TEXT_IO;

...
type COUNT_TYPE is range 0 .. 10000;
package COUNT_IO is
  new TEXT_IO.INTEGER_IO (COUNT_TYPE);
COUNTER: COUNT_TYPE;
```

Following this

```
COUNT_IO.PUT (COUNTER);
```

outputs the value of COUNTER in human-readable form.

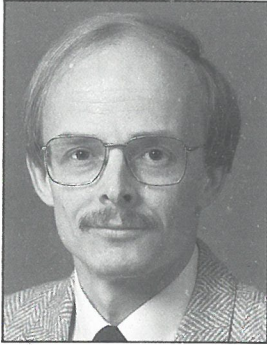


Veikko Kehä is a Software Engineering Manager in Nokia Data Product Management & Development.

END_ERROR:	reading past the end of a file.
NAME_ERROR:	opening a non-existing file or creating a file with a improper name.
STATUS_ERROR:	operating a file that has not been opened, or opening a file that has already been opened.
DATA_ERROR:	reading an element that is not of the required type.
MODE_ERROR:	reading from an output file or writing into an input file.
DEVICE_ERROR:	malfunction of the hardware or system.
USE_ERROR:	non-valid access right, creating a file that already exists, non-permissible overwriting etc.
LAYOUT_ERROR:	Setting column and line numbers in excess of the maximum maximum in TEXT_IO, etc.

Figure 1: Error-messages of SEQUENTIAL_IO.

*To be able to read or write
numeric values, we have to
instantiate packages for every
different type.*



Lassi Juutinen is a Senior Software Engineering Specialist in Nokia Data Product Management & Development.

Another example:

```
package BOOLEAN_IO is
  new TEXT_IO.ENUMERATION_IO (BOOLEAN);

BOOLEAN_IO.PUT (TOGGLE);
```

FALSE or TRUE is printed out, depending on the value of the Boolean variable TOGGLE.

SEQUENTIAL_IO and DIRECT_IO

Here is an example, if we want to read and write a file with personal records, we write:

```
with SEQUENTIAL_IO, PERSONS;
package PERSON_IO is
  new SEQUENTIAL_IO (PERSONS.PERSON_TYPE);
```

This instantiation creates a package that handles the I/O of personal files. PERSON_TYPE is declared in the package PERSONS.

To use this PERSON_IO package, we must first declare the file variable(s). We can then use the predefined procedures and functions to open and use the file(s). The example in Figure 2 shows how to copy a personal file.

Note that exceptions can always be raised by the system when the programmer operates files. So we have to write the exception handlers. It is good programming practice at least to close files in all error situations.

In the predefined package IO_EXCEPTIONS, the following exceptions are declared: DATA_ERROR, DEVICE_ERROR, END_ERROR, MODE_ERROR, NAME_ERROR, STATUS_ERROR, USE_ERROR, and LAYOUT_ERROR (see box). These can be raised in any Ada I/O package.

The SEQUENTIAL_IO package includes the procedures CREATE, OPEN, WRITE, READ, RESET, CLOSE, and DELETE. There are also the functions END_OF_FILE, IS_OPEN, MODE, and NAME, among others. The DIRECT_IO package declares the procedure SET_INDEX and the functions SIZE and INDEX. These two packages contain all the basic operations for handling files.

References

- /1/ Reference Manual for the Ada Programming Language, January 1983.
- /2/ Booch, G.: Software Engineering with Ada, Benjamin/Cummings, 1987.
- /3/ Barnes, J. G. P.: Programming in Ada, Addison-Wesley, 1989.
- /4/ Cohen, N.: Ada As a Second Language, McGraw-Hill, 1986.

Additional Properties of TEXT_IO

We could also use SEQUENTIAL_IO to perform I/O for text files, but the predefined package TEXT_IO declares additional operations to facilitate the use of text files.

The additional features are pages, lines, and columns. They include the procedures SET_PAGE_LENGTH, SET_LINE_LENGTH, SKIP_PAGE and NEW_PAGE for skipping pages, and correspondingly SKIP_LINE and NEW_LINE.

The function COL returns the current column number, and the procedure SET_COL sets the current column. The line position is located by the function LINE, and the procedure SET_LINE sets the current line. The functions END_OF_LINE and END_OF_PAGE return a boolean value FALSE or TRUE. Complete descriptions of these features are given in the Reference Manual. ■

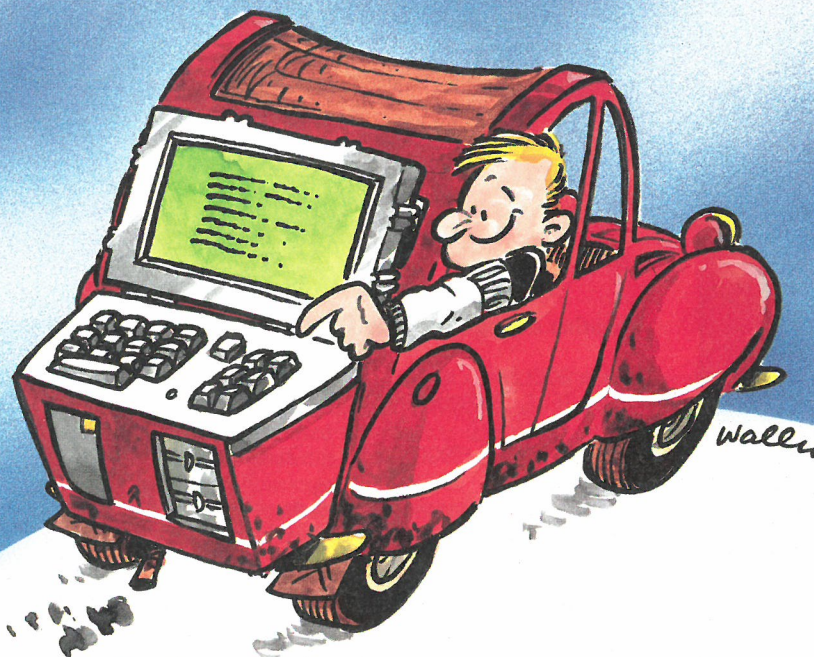
Additional Features in TEXT_IO

```
with PERSON_IO, PERSONS;

procedure COPY_PERSON_FILE is
  PERSON : PERSONS.PERSON_TYPE;
  INPUT_DATA,
  OUTPUT_DATA : PERSON_IO.FILE_TYPE;

begin
  PERSON_IO.OPEN (INPUT_DATA,
    PERSON_IO.IN_FILE, "CUSTOMERS.OLD");
  PERSON_IO.CREATE (OUTPUT_DATA,
    PERSON_IO.OUT_FILE, "CUSTOMERS.NEW");
  COPY_PERSONS:
  while not
    PERSON_IO.END_OF_FILE (INPUT_DATA)
  loop
    PERSON_IO.READ
      (INPUT_DATA, PERSON);
    PERSON_IO.WRITE
      (OUTPUT_DATA, PERSON);
  end loop COPY_PERSONS;
  PERSON_IO.CLOSE (INPUT_DATA);
  PERSON_IO.CLOSE (OUTPUT_DATA);
exception
  when others =>
    PERSON_IO.CLOSE (INPUT_DATA);
    PERSON_IO.CLOSE (OUTPUT_DATA);
    PERSON_IO.DELETE (OUTPUT_DATA);
end COPY_PERSON_FILE;
```

Figure 2: How to use SEQUENTIAL_IO.



SQLWindows-kurssi

Softarin numerossa 1/89 alkoi neliosainen SQLWindows-kurssi. Kolmessa edellisessä numerossa olemme saaneet aikaiseksi jäsenrekisterisovelluksen valikkorakenteen, keskusteluikkunan uuden jäsenen tietojen syöttämistä ja tietokantaan viemistä varten sekä List box -rakenteen jäsenten nimien selausta varten. Tässä viimeisessä osassa tarkastellaan SQLWindowsiin kehitettyä uutta rakennetta nimeltä Table Window.



Tiina Mikonmäki vaikuttaa Mikonmäki Oy:ssä p. (915) 49 388. Hän toimii verailevana kouluttajana mm. Nitecissä aiheinaan C-kieli, SQLWindows ja OS/2 Presentation Manager.

Kuvassa 1 nähdään Table Window -rakenne ikkunan alalaidassa. Tällä kertaa sovellus toimii siten, että Table Window käynnistyy kun käyttäjä valitsee Jäsenistö-valikosta Jäsenluettelo-vaihtoehtoon eli siihen ilmestyy selattavaksi kaikkien jäsenten kaikki tiedot.

Viime numerossa saatiin aikaan List box, johon tulostui pelkät jäsenten nimet. Siihen liitettiin painonappi poistettiin ikkunasta. Sen tilalle on tehty toinen List box, josta käyttäjä voi valita nähtäväksi esimerkiksi vain niiden jäsenten tiedot, jotka omistavat sätkän. Tarkastellaan tätä List boxia myöhemmin, mutta ensin on vuorossa Table Window.

Table Window vai List box?

Viime kerralla jo todettiin, että List box on tarkoitettu merkkijonon rivien selaamiseen tai valitsemiseen. Jos tieto ei ole merkkimuotoista, se on muutettava selaiseksi. Näitä merkkimuotoisia

tietoja voi tietysti näyttää useilta riveiltä käyttämällä merkkijonon yhdistelymerkkiä (|) esimerkiksi:

```
SalListInsert  
(hWndNimiOs, -1, strNimi || strOs)
```

Table Window on tarkoitettu nimenomaan erityyppisten tietojen näyttämiseen (tai muokkaamiseen tai valintaan). Tiedot sijaitsevat yleensä tietokannassa, mutta se ei ole välttämätöntä.

Table Windowhin määriteltävät sarakkeet (kuvassa 1 esim. nimeltään NRO, NIMI, K, 1 ja 2) ovat kukin kuin erillisiä Data Field -kenttiä. Toisin sanoen niille voi määritellä yksilölliset tietotyypit sekä Format- ja Country-tiedot. Tästä johtuu myös se olennainen ero, että Table Windown sarakkeita voi editoida, jos ohjelmassa niin halutaan.

Table Window -rakennetta tarvitaan myös, jos List boxia ei voida piirtää niin leveäksi, että kaikki tieto mahtuu siihen. Table Windowsissa on nimittäin vaakataulon vieritysmahdollisuus valmiina.

SQLWindows

Nopeuksissa on merkittävä ero, sillä Table Windowhin järjestelmä lukee vain ne rivit tietokannasta, jotka käyttäjä haluaa nähdä, mutta List boxin tapauksessa luetaan kaikki siihen tulevat rivit, halusipa käyttäjä niitä nähdä tai ei. Lisäksi Table Windowlle voidaan määritellä kätkömuistin (cache) määrä eli montaako viimeksi käsiteltyä riviä pidetään työmuistissa valmiina uudelleen-käsittelyä varten.

Table Windown voi määritellä myös top-level-ikkunaksi eli ikkunaksi, jossa ei ole muita olioita. Tällöin siihen saa myös otsikkotekstin ja valikon. Top-level-ikkunaksi määrittäminen käy samoin kuin Dialog boxin määrittely: View-valikosta Outline Options-vaihtoehto, jonka jälkeen Outlinen ohjelmasta valitaan Form Window -tasoinen otsikko ja sitten Outline Optioneista Table Window. Ikkunan käynnistäminen kävi funktiolla SalCreateWindow.

Table Windown käyttämisen haittapuolena on se, että se varaa itselleen yhden tietokannan kahvan (käytettävissä kaikkiaan neljä). Kannattaa myös tarkistaa, kuinka paljon enemmän koodia Table Window rakenteena tuottaa List boxiin verrattuna.

Table Windown käsittelytavat

Table Windowta voidaan käyttää joko pelkkänä tulostusalueena, valinnan tekemiseen tai sitten siinä näkyvien tietojen muokkaamiseen.

Kaksi ensin mainittua tapaa ovat käytössä myös List boxeissa. Jotta Table Windowsta voitaisiin valita yksi tai useampia rivejä, on rivin alkuun määriteltävä sarakke, joka ei ole editoitavissa. Muuten hiiren näppäimen napautus rivin alussa muuttaisi kohdistimen pystyviivaksi eli editointitilaan.

Jos Table Window on määriteltä siten, että joitakin siinä olevia tietoja voidaan muuttaa, on muutosten tietokantaan viemiseen kaksi vaihtoehtoa: joko välittömästi tai eräajan tapaan.

Muutokset viedään tietokantaan välittömästi, kun kyseisen sarakkeen Message Actions -kohdassa reagoidaan esimerkiksi SAM_FieldEdit-sanomaan.

Table Window mahdollistaa eräajotyylisen käsittelyn pitämällä kirjaa kaikista riveihin kohdistuneista muutoksista. Jokaisella rivillä, jota on muutettu, on ROW_Edited-bitti asetettuna. Vastaavasti uusilla riveillä on ROW_New- ja käyttäjän valitsemilla riveillä ROW_Selected-bitti asetettuna. Funktiolla SalTblFindNextRow avulla voi etsiä esimerkiksi kaikki ne rivit, joilla on ROW_Edited-bitti asetettuna ja tehdä vastaaville riveille muutokset tietokantaan.

Piirretään jäsenrekisterisovellukseen kuvan 1 mukainen Table Window.

Table Windown piirtäminen

Piirretään Table Window Pääikkunaan (Form Window). Jos Pääikkuna ei ole näkyvässä, sen saa esiin valitsemalla Outline-ikkunan ohjelmasta Form Window -rivin ja sen jälkeen View-valikosta Show Window.

Nyt Draw-valikosta voidaan valita vaihtoehto Table Window ja viedä hiirikohdistin Pääikkunan päälle siihen kohtaan, johon Table Window aiotaan piirtää. Paina hiiren vasen näppäin alas, mutta älä vapauta sitä ennenkuin olet venyttänyt Table Windown suunnilleen sopivan kokoiseksi. Kokoa voi muuttaa myöhemminkin Table Windown reunoista (oikeastaan liukusäätimien sisäreunoista, joissa hiirikohdistin muuttuu kaksipäiseksi nuoleksi).

Sitten piirretään sarakkeet. Vie hiirikohdistin piirtämäsi Table Windown ylimmälle riville, jolloin kohdistimen muoto muuttuu Col-kohdistimeksi. Voit painaa hiiren vasemman näppäimen alas ja samalla venyttää sarakkeen suurin piirtein sopivan levyiseksi, minkä jälkeen voit vapauttaa alaspainetun näppäimen. Leveyttä voi myöhemmin muuttaa ainoastaan Outlinessa olevan ohjelman puolelta.

Seuraavaksi kannattaa täyttää sarakkeen tiedot Outlineen. Jos et sarakkeen piirtämisen jälkeen tehnyt muuta, pystyviivakohdistin odottaa Column-rivin lopussa sarakkeen nimeä. Sarakkeiden nimet alkavat col-etuliitteellä, jonka jälkeen kannattaa ilmaista sarakkeen tyyppi, esimerkiksi str=String tai n=Number. Esimerkki-

```
Column: colNro
Title: NRO
Visible? Yes
Editable? No
Maximum Data Length: 5
Data Type: Number
Justify: Left
Width: 5
Format: Unformatted
Country: Default
Message Actions
```

Kuva 1 : Jäsentiedot tulostettuna Table Window -rakenteeseen.

ratkaisussa on käytetty pelkkää col-etuliitettä.

Kuvassa 2 näkyy, miten juuri piirretyn sarakkeen tietoja pitäisi muuttaa. Kannattaa muistaa, että esimerkiksi Editable-rivin tietoa pääsee muuttamaan painamalla rivillä hiiren OIKEAA näppäintä ja sen jälkeen ottamalla vaihtoehtot esiin painamalla näppäimistöltä nuoli ylös -näppäintä.

Tärkeitä tietoja sarakkeesta ovat Editable — saako käyttäjä muuttaa sarakkeen tietoja vai ei, Maximum Data Length — kuinka monta merkkiä kenttään voi syöttää, Data Type — tiedon tyyppi (syötteen oikeellisuus tarkistettava itse) sekä Width — kuinka leveältä sarakke näyttää näytöllä.

Tämän jälkeen voidaan muuttokin sarakkeet tehdä kuvan 1 mukaisiksi viemällä hiirikohdistin Table Windown ylimmälle riville ja toimimalla ylläolevien ohjeiden mukaisesti. Tietokannan JASENET-aulussa on seuraavanlaiset sarakkeet: NRO integer, NIMI char(30), KULJIN char(1), VARI1 char(1) ja VARI2 char(1).

Kuva 2 : Table Windown ensimmäisen sarakkeen määrittely.

SQLWindows

Kolmen viimeisen sarakkeen osalta Table Windowta voisi tehdä selkeämmäksi kirjoittamalla sarakkeiden nimet kokonaisina ja määrittelemällä Width-tiedon vastaavasti.

Table Windowta voi kelata oikealle ja sinne jää tyhjää kelattavaa, jonka poistamiseen en ole vielä keksinyt keinoja. Ehkä tulevissa versioissa tämä on mahdollista.

Koko Table Windowta koskevat määrittäykset

Kun olet saanut sarakkeet piirrettyiksi, sulje Table Windown rivit napauttamalla hiiren vasenta näppäintä kahdesti. Avaa sitten seuraavan tason otsakkeet samalla tavalla.

Anna Table Windowlle nimi. Esimerkkiohjelmassa on käytetty Jasenluettelo-nimeä, mutta Nokian suosituksessa kehoitetaan käyttämään jotain tbl-alkuista kuten tblJasenluettelo.

Table Windowlle on Title-, Icon File- ja Menu-otsakkeet, jotka eivät kuitenkaan ole käytössä, sillä Table Windown piirtäminen alkoi Draw-valikosta. Tällöin Table Window tulee jonkin ikkunan Contents-kohtaan eikä top-level-ikkunaksi. Top-level-ikkunaksi sen olisi saanut View-valikon Outline Options-vaihtoehdosta.

Tärkeitä Table Windown määrittäjiä löytyy Memory Settings-otsakkeen alta. Kohdassa Maximum Rows in Memory määritellään, montaako viimeksi käsitellyä riviä pidetään työmuistissa. Muuta sen arvo Dynamic esimerkiksi arvoksi 50 ja tarkista myös, että kohdassa Discardable? on arvona Yes. Tällöin, jos työmuistin kätkömuistitila loppuu, annetaan lupa viedä rivejä levyille, josta ne taas luetaan tarpeen vaatiessa uudelleen.

Jos Table Windowhin määriteltäisiin editoitavia sarakkeita, pitäisi ottaa huomioon, että järjestelmää pitää muutetut rivit kätkömuistissa ja siis varauduttava sanoman SAM__CacheFull tulemiseen.

Tämän jälkeen avaa Table Windown Contents-kohta ja sieltä yksi sarake. Jokaisella sarakkeella on oma Message Actions -kohdansa. Jos Table Windowta tullaan käyttämään vain tulostus-alustana, kuten meidän tapauk-

sessamme, nämä kohdat ovat tyhjiä. Sen sijaan koko Table Windown Message Actions -kohta on erittäin tärkeä ohjelman toiminnalle.

Table Window toimimaan!

Table Windown Message Actions -kohdassa on aina samannäköinen reagointi sanomaan SAM__fetchrow. Se näkyy kuvassa 3. Siinä Table Window on aina sanoman SAM__FetchRow tullessa

Table Window: Jasenluettelo

Title: (untitled)
Icon File: (none)
Display Settings
Memory Settings
Menu
Contents
Variables
Message Actions
On SAM__FetchRow
If not SqlFetchRow(hSqlTable, lParam, nInd)
Return FALSE
Return TRUE

Kuva 3 : Table Windown Message Actions -kohdan vakiorakenne.

Application Actions

```
On SAM__AppStartup
  Call SqlConnect(hSql)
  Call SqlConnect(hSqlTable)
On SAM__AppExit
  Call SqlDisconnect(hSql)
  Call SqlDisconnect(hSqlTable)
... Table Windown käynnistys
... (jossakin Actions- tai
... Message Actions -kohdassa)
Call SqlClose (hSqlTable)
Call SqlPrepare (hSqlTable, 'SELECT NRO, NIMI FROM
  ASIAKAS INTO :colNro, :colNimi')
! Let us open a workspace (cursor)
Call SqlOpen (hSqlTable, 'ASIAKKAAT')
! For which table?
Call SalTblReset(Asiakkaat)
! Let us open the range of the table in order to
! start sending of SAM__FetchRow-messages
! when needed
Call SqlImmediate('SELECT COUNT(NRO) FROM
  ASIAKAS INTO :nTblRows')
Call SalTblSetRange(Asiakkaat, 0, nTblRows-1)
```

Table Window: Asiakkaat

Title: (untitled)
Icon File: (none)
Display Settings
Memory Settings
Menu
Contents
Column: colNro
Column: colName
Variables
Message Actions
On SAM__FetchRow

If not SqlFetchRow(hSqlTable, lParam, nInd)

Return FALSE

Return TRUE

riviä ei löytynyt, älä piirrä
rivi löytyi, piirrä se sarakkeistani

SAM__FetchRow
lParam=0
lParam=1

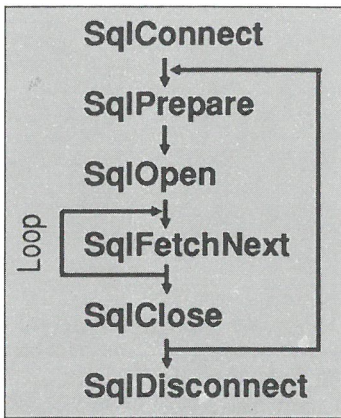
SQLWindows

Täytä Table Window SqlPreparaessa ilmoitetuilla tiedoilla

OK, lähetän SAM__FetchRow-sanomia

OK! En piirrä
OK! Piirrän

Kuva 6 : Rivien lukeminen tietokannasta Table Windowhin.



Kuva 4 : Funktiot, joita tarvittiin tietokannan rivejä List boxiin luettaessa.

valmis hakemaan yhden rivin tietokannasta. Tarkastellaan vähän myöhemmin tätä koodinpätkää tarkemmin.

Tämä koodi Table Windown Message Actions -kohdassa vastaa List box -esimerkissämme Loop-silmukan sisällä ollutta SqlFetchNext-funktion kutsua. Siinä tarvittavia funktioita voi kerrata kuvan 4 avulla.

Kuvassa 5 on lueteltu ne funktiot, joita tarvitaan Table Windown varten.

Kuvassa 6 on esimerkki ohjelmanpätkästä, jossa luetaan tietokannan ASIAKAS-taulusta asi-

akkaiden numerot ja nimet Table Windowhin.

Application Actions -kohtaan on lisätty uusi Connect tietokantaan käyttäen kahvaa nimeltä hSqlTable, joka on määritelty globaaliksi muuttujaksi.

Table Windown käynnistys

Table Windown käynnistys tapahtuu jostakin valikon vaihtoehdosta tai painonapista, jolloin Actions- tai Message Actions -kohdassa ensin varmistetaan, että hSqlTable-kahva on käytettävissä (SqlClose). Se kahvahan jätetään aina auki, sillä käyttäjä voi milloin vain kelata Table Windowta alaspäin, jolloin täytyy olla avoin kahva, jonka perusteella SqlFetchRow-funktiolla haetaan tietokannasta rivi.

Sen jälkeen SqlPreparella käännetään SELECT-lause, jonka INTO-osassa luetaan ne Table Windown sarakkeet, joihin tiedot tietokannasta luetaan. Sen jälkeen tarvitaan funktion SqlOpen kutsu, joka lukee vastausjoukon tietokannasta kätkömuistissa olevaan työtilaan eli kursoriin palvelinkoneelle.

Seuraavat käynnistysfunktiot liittyvät Table Windowhin, sillä ne alkavat SalTbl-etuliitteellä. SalTblReset antaa ilmoituksen, että Table Window aiotaan täyttää uudelleen. Samalla se nollaa niin kutsutun Rangen. Se on lyhenne Scroll Rangesta, joka kertoo, montako riviä käyttäjä voi kelata Table Windowssa alaspäin. Jos Scroll Range on nolla, SQLWindows ei lähetä Table Windowlle yhtään SAM_FetchRow-sanomaa. Scroll Range avataan funktiolla SalTblSetRange, joka esimerkissovelluksessa asettaa Scroll Rangen nolasta ASIAKAS-taulussa olevien rivien lkm miinus 1:ksi. Rivien lukumäärä lasketaan käyttämällä SELECT-lauseessa SQLBasen COUNT-funktiota.

Rivien lukumäärän laskemiseen käytetään funktiota SqlImmediate, mutta siinäkin voisi käyttää omalla kahvalla Connectattua SqlPrepare-SqlExecute-paria, jolloin toiminta olisi nopeampaa.

Tässä vaiheessa SQLWindows tietää siis, että sen on lähetettävä SAM_FetchRow -sanomia Table Windowlle nimeltä Asiakkaat. Se

lähettääkin SAM_FetchRow-sanoman jokaiselle riville, jonka pitäisi sillä hetkellä näkyä näytöllä ja joka ei vielä ole luettuna Table Windown kätkömuistissa. Aina, kun se lähettää sanoman, se myös asettaa sovelluksen globaaliin, valmiiksi määriteltyyn lParam-muuttujaan sen rivin numeron, joka pitäisi hakea tietokannasta.

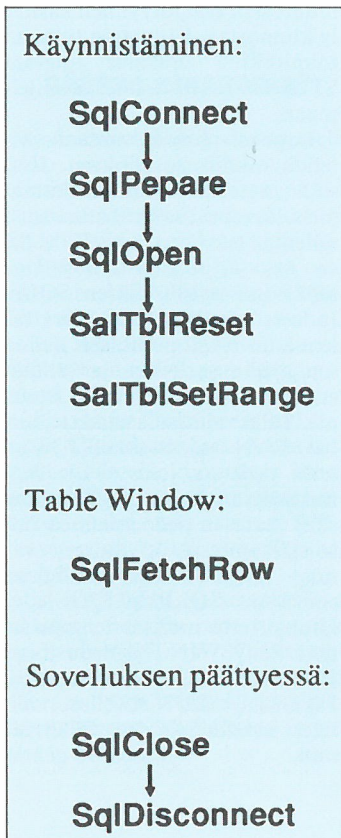
Table Window herää henkiin

Table Window toimii aina samoin saatuaan SAM_FetchRow-sanoman. Se kutsuu funktiota SqlFetchRow, jonka ensimmäisenä parametrina on tietokannan kahva. Tämän parametrin avulla Table Window liittyy Table Windown käynnistysvaiheessa tehtyyn SqlPrepareen. Seuraavana parametrina annetaan lParam, jolle SQLWindows on siis asettanut arvoksi haettavan rivin numeron. Sovelluksessammehan emme tiedä esimerkiksi sitä, onko käyttäjä kelannut rivejä ylös- vai ko alaspäin, ja mitkä rivit ovat jo kätkömuistissa.

Paluutiedon SqlFetchRow-funktio välittää kahdella eri tavalla. Ensinnäkin koko funktio palauttaa arvonaan (ks. Global Declarations: External Functions: Library name: SqlFetchRow: Returns) FALSEn, jos kaikki rivit jo luettu eli riviä ei löytynyt tietokannasta. Tätä mahdollisuutta on hyödynnetty myös esimerkkisovelluksessa.

Toisaalta funktio palauttaa tietoa onnistumisesta myös kolmantena parametrina annettuun muuttujaan (esimerkin nInd). Kolmas parametri on funktion määrittelyssä asetettu tyypiksi Receive Number eli funktio saat- taa muuttaa parametrina annetun muuttujan arvoa kutsun aikana. Toisin sanoen kutsussa on oltava siinä kohtaa muuttuja, vaikkei sen arvoa tutkittaisikaan.

Muuttujan nInd arvoa tulee tutkia, jos käytössä on monen käyttäjän järjestelmä, jossa yhteiskäyttötasoksi on määritelty Cursor Stability. Tällöin nInd kertoo, onko joku muu ehtinyt muuttaa tietokannan tietoja sen yhden kilotavun kokoisen sivun osalta, joka on ollut ladattuna kätkömuistiin käsittelyä varten (nInd arvot 2 ja 3). Tätä ei yleensä tutkita kuin vasta sitten, kun tie-



Kuva 5 : Funktiot, jotka tarvitaan tietokannan rivejä Table Windowhin luettaessa.

Application Description: SQLWindows-kurssin sovellus

Design-time Settings

Global Declarations

External Functions

Constants

System

User

```
String: INSERT = 'insert into jasetnet
(nro, nimi, kuljin, vari1, vari2)
values (:nro, :nimi, :MikaKuljin, :Vari1, :Vari2)'
String: SELECT_COUNT = 'SELECT COUNT(*)
FROM JASENET INTO :nTblRows'
String: SELECT_TO_TABLE = 'SELECT
NRO,NIMI,KULJIN,VARI1,VARI2
FROM JASENET INTO
:colNro,:colNimi,:colKuljin,:colVari1,:colVari2'
```

Variables

...

Sql Handle: hSqlTable

Internal Functions

Application Actions

On SAM_AppStartup

Call SqlConnect(hSqlTable)

On SAM_AppExit

Call SqlDisconnect(hSqlTable)

Form Window: MainForm

Title: Kurssisovellus, Jäsenluettelo - TABWB.APP

Icon File: (none)

Display Settings

Menu

Popup Menu: &Jäsenistö

Menu Item: &Uusi jäsen...

Menu Item: Jäsen&luettelo

Keyboard Accelerator: (none)

Menu Settings

Actions

! Avataan tablelle kursori

Call SqlClose (hSqlTable)

Call SqlPrepare (hSqlTable, SELECT_TO_TABLE)

Call SqlOpen (hSqlTable, 'TABLEQRYCUR')

! Käynn. SAM_FetchRow-sanomien lähetys tablelle

Call SalTblReset(Jäsenluettelo)

! Lasketaan tablen maksimikoko

Call SqlImmediate(SELECT_COUNT)

Call SalTblSetRange(Jäsenluettelo, 0, nTblRows-1)

Menu Separator

Menu Item: E&xit

Menu Item: A&bout...

Contents

Table Window: Jäsenluettelo

Title: JASENLUETTELO:

Icon File: (none)

Display Settings

Memory Settings

Menu

Contents

Column: colNro

...

Variables

Message Actions

On SAM_FetchRow

If not SqlFetchRow(hSqlTable, lParam, nInd)

Return FALSE

Return TRUE

Background Text: JASENLUETTELO

Background Text: K = kuljin, 1 = vari1 (on/ei), 2 = vari2

Variables

Number: nInd

Number: nTblRows

Message Actions

SQLWindows

tokantaan aiotaan viedä päivityksiä.

Esimerkissä siis riittää sen tutkiminen, että saadaanko työtilasta luetuksi SQLWindowsin pyytämä rivi vaiko ei. Kontrolli palautetaan tuloksen mukaan sanoman käsittelystä SQLWindowsille ja paluuarvolla TRUE tai FALSE ilmoitetaan, onko Table Windown sarakkeissa piirrettävän rivin tiedot vai ei. SQLWindows hoitaa sarakkeisiin piirtämisen muun näytön ylläpidon ohessa.

Table Windowta toteuttamaan

Tee viimeistään tässä vaiheessa esimerkisovellukseemme vastaavat koodinpätkät, niin pääset näkemään Table Windown toiminnan havainnollisemmin. Koodaa Table Windown käynnistys vaikka valikossa olevaan Jäsenluettelo-vaihtoehtoon.

Kun saat ohjelmasi toimimaan, laita se User Modeen ja aseta vielä animaatio päälle. Usein tässä vaiheessa tulee ajoaikaisia SQL-virheilmoituksia. Niissä ilmoitettu Position tarkoittaa, monenhenko merkin kohdalla SQL-lauseen alusta lukien virhe havaittiin. Sovelluksessasi voit puuttua näiden virheilmoitusten tulemiseen tai muotoon reagoimalla sanomaan SAM_SqlError manuaalissa opastetulla tavalla.

Jos näytölläsi on vielä List box, vertaa rivien hakemisvaihetta toisaalta List boxiin, toisaalta Table Windowhin. Valitse sitten valikosta Jäsenluettelo-vaihtoehto ja tarkkaile, miten SAM_FetchRow -sanomia tulee. Kelaa Table Windowta alaspäin muutama rivi ja sen jälkeen takaisin ylöspäin.

Huomaat, miten sanomia tulee vain riveille, joita ei ole vielä luetu kätkömuistiin.

Ratkaisu edellä esitettyyn tehtävään löytyy kuvasta 7.

Lisäharjoituksia

Jos aikaa riittää, Table Windown kätkömuistin kokoa voi yrittää pienentää ja tutkia tällöin SAM_FetchRow -sanomien tuleamista.

Kuvassa 1 näkyvän List boxin voi myös lisätä sovellukseen ja täyttää Table Windown List boxista tehdyn valinnan mukaisesti. Eräs ratkaisu tähän löytyy kuvasta 8.

Jos Table Windown käyttö myös tietokannan editointiin ja muutosten eräajotyylinen käsittely kiinnostaa, kannattaa tutustua toimituksen mukana tulevaan STUDENT.APP-esimerkkiohjelmaan.

Lopuksi sovelluksestaan voi tehdä valmiin sovelluksen. Jotta siitä saataisiin mahdollisimman pieni ja nopea, se kannattaa ensin tallentaa tekstimuotoon (File: Save As) ja poistua sen jälkeen SQLWindowsista. Sitten SQLWindows käynnistetään juuri tallennetun tekstimuotoisen tiedoston avulla uudelleen ja valitaan ensimmäiseksi File: Make Runtime. Tällä valinnalla saadaan luotua .RUN-laajennusosalla varustettu tiedosto, joka ei ole enää luettavissa takaisin SQLWindowsiin, vaan sen tiedostonimeä kaksoisklikattaessa käynnistyy valmiin SQLWindows-sovelluksen suorittaja SQLRUN.EXE edellyttäen, että moinen temppu on määritelty WIN.INI-tiedostossa. Poistu siis SQLWindowsista ja käynnistä. RUN-sovellus, niin näet sovelluksesi käyttäjän silmin.

Kuva 7 : Table Window -tehtävän ratkaisu.

Loppu hyvin, kaikki hyvin?

Tässä vaiheessa olemme käyneet läpi jokseenkin kaikki SQLWindowsin oliot ja sinulla pitäisi olla jonkinlainen näkemys SQLWindows-ohjelmoinnista, jotta pääset jatkamaan harjoituksia. Jos olet jaksanut seurata kurssia, on sinulta mennyt siihen varmastikin useita tunteja. Jos nyt tekisit kaiken uudelleen, hommaan käytetty aika putoaisi murto-osaan. Ko-keile, jollet usko!

Samalla voit miettiä, onko tekemämme esimerkkiohjelma esimerkillinen vai voisiko sen käyttöliittymää korjata ja miten. Ehkä tämä on esimerkki siitä, ettei Windows sinänsä takaa hyvää käyttöliittymää, vaan kaikki on ohjelmoijasta kiinni.

Miten tästä pääsee eteenpäin?

Tämän kurssin jälkeen on vielä koko joukko opetettavaa. Viimeistään tässä vaiheessa pitäisi tutustua Microsoftin Application Style Guideen tai nokialaisille tarkoitettuihin, kehityksessä tuotettuihin käyttöliittymän tekoon liittyviin dokumentteihin.

Jos Windows-ohjelmoijalle on Windowsin tuntemus välttämätöntä, SQLWindows-ohjelmoijan pitäisi tuntea lisäksi vielä SQLBase. SQLBasetuntemuksen saa helpoiten Nitecin kurseilta. Tässä artikkelisarjassa ollaan aina käytetty helpointa tietä SQL-lauseissa eli kaikki virhetilanteet on jätetty huomiotta jne.

Yleensä tietokantasovelluksista halutaan erilaisia raportteja. Ne laaditaan SQR- tai SQRW- ohjelmalla, joka voidaan käynnistää suoraan SQLWindows-sovelluksesta. Myös SQR-kursseja järjestetään Nitecissä.

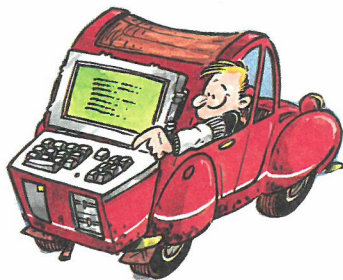
Itse SQLWindowsissakin on vielä melko lailla tutkittavaa. Esimerkiksi vakioiden ja omien SAL-tai C-kielisten funktioiden käyttöä ei ole lainkaan tarkasteltu.

Melko pian tulee eteen myös sovelluksen jako pienempiin sovelluksiin, sillä DOS-versiossa tulee aina olemaan muistiongelmia, ellei täytä konettaan expanded-muistilla.

Kun käytetään useita sovelluksia, joutuu tutustumaan myös käsitteeseen nimeltä Dynamic Data Exchange (DDE). DDE avaa ai-
van uusia mahdollisuuksia Windows-ohjelmoijalle. Otetaan esimerkiksi vaikka tilanne, jossa SQLWindows-sovellus päivittää tietokantaa ja lähettää samanai-
kaisesti viestejä Excelille, joka välittömästi näyttää tietokantaan tulleet muutokset graafisena käy-
ränä.

Tämän artikkelisarjan sisältö oli suurin piirtein sama kuin Nitecissä pidettävän SQLWindows-kurssin. Kurssilla on kuitenkin mahdollista saada henkilökoh-
taista opastusta.

Toivottavasti tapaamme joskus kursseilla! Hauskoja hetkiä SQLWindowsin parissa! ■



Kuva 8 : Vain valitut rivit Table Windowhin List boxin avulla.

Application Description: SQLWindows-kurssin sovellus

Design-time Settings

Global Declarations

External Functions

Constants

System

User

String: WHERE = 'WHERE kuljin ='

Variables

Internal Functions

Application Actions

Form Window: MainForm

Title: Kurssisovellus, Vain valitut - TABWC.APP

Icon File: (none)

Display Settings

Menu

Popup Menu: &Jäsenistö

Menu Item: &Uusi jäsen...

Menu Item: Jäsen&luettelo

Keyboard Accelerator: (none)

Menu Settings

Actions

! Katso, mikä kuljin valittu list boxista

Set lInd = SalListQuerySelection(hWndKuljinLista)

Call SalListQueryText(hWndKuljinLista,lInd,Text)

Call SalStrLeft(Text,1,Selection)

! Avataan tabellelle kursori valinnan mukaan

Call SqlClose (hSqlTable)

If Selection = '0'

Else

Call SqlOpen (hSqlTable, 'TABLEQRYCUR')

! Käynn. SAM_FetchRow-sanomien lähetyks tabellelle

Call SalTblReset(Jasenluettelo)

! Lasketaan tablen maksimikoko

Call SqlImmediate(SELECT_COUNT)

Call SalTblSetRange(Jasenluettelo, 0, nTblRows-1)

Menu Separator

Menu Item: E&xit

Menu Item: A&bout...

Contents

Background Text: VALITSE KULJIN:

List Box: KuljinLista

Window Location and Size

Multiple selection? No

Sorted? Yes

Vertical Scroll? Yes

Message Actions

On SAM_Create

Set hWndKuljinLista=hWndItem

Call SalListClear(hWndKuljinLista)

Call SalListInsert(hWndKuljinLista,-1,'0 Kaikki')

Call SalListInsert(hWndKuljinLista,-1,'1 Sätkä')

Call SalListInsert(hWndKuljinLista,-1,'2 Ami')

Call SalListInsert(hWndKuljinLista,-1,'3 Paku')

Call SalListSetSelect(hWndKuljinLista,0)

Set Selection = '0'

Table Window: Jasenluettelo

Variables

Window Handle: hWndKuljinLista

Number: lInd

Number: nInd

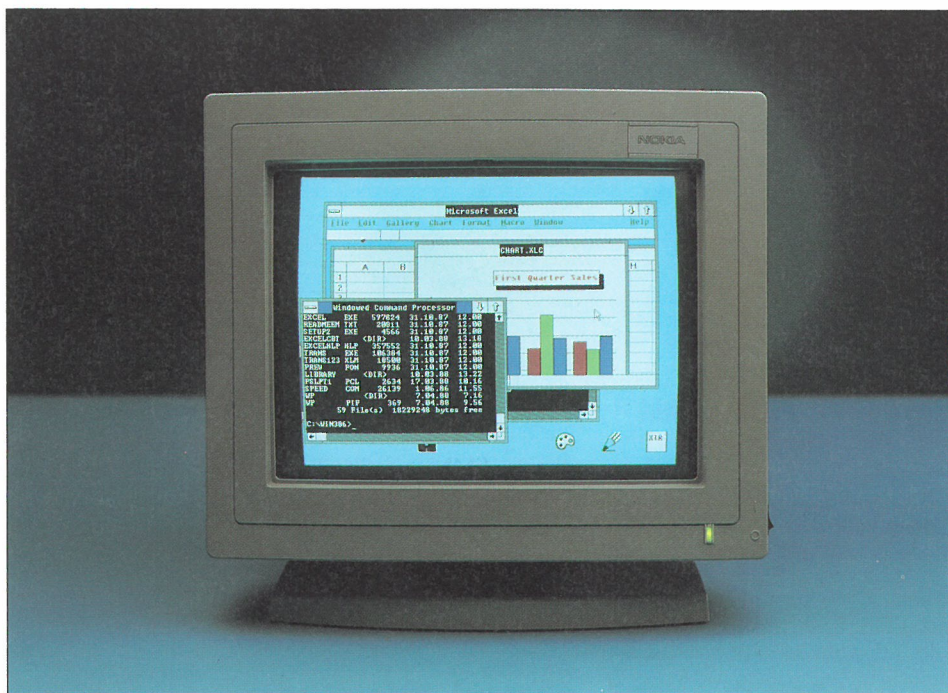
String: Selection(1)

String Text(30)

Number: nTblRows

Message Actions

Windows 2.11 ja muistin mysteerit



Mikä mihinkin ja miksi

Windows on yleistynt vauhdilla AT- ja 386-yhteensopivien mikrojen tahdissa. Moni ei kuitenkaan käytä järjestelmäänsä täysipainoisesti, vaan osa hyödyllisistä ajuriohjelmista tai muistialueista on unohtunut alustuksista pois, eikä niitä näin ollen ole Windowsin mielestä lainkaan olemassa.

Artikkeli jakautuu käytännössä kahteen osaan. Tässä numerossa perehdytään muistin periaatteelliseen toimintaan Windows-ympäristössä. Softari 1/90:ssä annetaan puolestaan käytännön ohjeita siitä, miten oman tietokoneensa ja Windowsinsa saa määriteltä hyödyntämään muistiin mahdollisimman tehokkaasti. Suurin osa tässä ja seuraavassa osassa esitetyistä numeerisista tiedoista ja suosituksista pohjautuu Jürgen Zieglerin tänä vuonna tekemään tutkimukseen.

Mega miinus 640 tekee tasan 384

Muistin 640 ensimmäistä kilotavua muodostavat perusmuistin (conventional memory). Nimensä se on saanut siitä, että sen suurempaa muistitilaa MS-DOS ei kykene hyödyntämään. Tämä ei ole suinkaan MS-DOSin vika, sillä se kirjoitettiin alun perin Intelin 8086- ja 8088-suorittimille, jotka kykenivät osoittamaan vain yhden megatavun (1024 kt) kokoista muistivarausta. Suorittimet sanelivat MS-DOSin suorituskyvyn rajat eli

kun muistin loppupäästä varatiin 384 kilotavua järjestelmämuistiksi (system memory), jäljelle jäi 640 kilotavua (1 Mt - 384 kt / 640 kt). Kaikkien ohjelmien on periaatteessa mahdollista pyöräyttää tässä tilassa, mutta Windows-sovellukset voivat onneksi kiertää tämän rajoituksen.

On tärkeää muistaa, että tavallisen 640 kilotavun muistilla varustetun koneen osoitevaraus on itse asiassa kokonainen megatavu. Ylimääräiset 384 kilotavua on varattu sellaisille järjestelmän ylläpitämiselle asioille kuin videomuistille ja BIOSille (Basic Input Output System). Tämä systeemimuisti sijoittuu aina samalle paikalle 640 kilotavun ja yhden megatavun väliin, vaikka varsinaista muistia olisi vain esimerkiksi 256 kilotavua. (Kaikki näiden artikkeleiden yhteydessä käsiteltävät tietokoneet sisältävät täydet 640 kilotavua perusmuistia.)

Kaiken takana on extended-muisti

Extended-muisti on laajennusmuistityypeistä toiminnaltaan kaikkein yksinkertaisin. Se alkaa

siitä, missä systeemimuisti loppuu eli sen ensimmäinen osoite on aina megatavu plus yksi.

80286-suorittinta käyttävien tietokoneiden osoitevaraus on kaikkiaan 16 megatavua. Koska MS-DOS ei kuitenkaan osaa hyödyntää siitä kuin pientä osaa, extended-muistin käyttömahdollisuudet ovat 80286-koneissa varsin rajalliset.

80386-koneissa extended-muistista on kuitenkin enemmän iloa, sillä 80386-suoritin osaa itse sivuttaa neljän gigatavun kokoisen osoitevarauutensa pienempiin osiin.

Kurkitaan rei'istä expanded-muistia

Expanded-muistin ideana on ohittaa MS-DOSin muistille asettamat rajat jaotteleamalla käytettävissä oleva muisti sivuihin ja käyttämällä näitä sivuja vuorotellen yksi kerrallaan (bank switching, memory mapping).

Nokian vanhoissa 80286-koneissa expanded-muistin käyttö vaatii erillisen expanded-kortin (suositus: Intel Above Board). Uudessa malleissa expanded-muisti

voidaan sijoittaa emokortille. Joka tapauksessa on kuitenkin käytettävä asianmukaista muistinhallintaohjelmaa (Expanded Memory Manager, EMM).

Mikäli 80386-ympäristössä pyöritetään Windows/286:ta, tarvitaan erillinen expanded-muisti-emulaattoriohjelma (Expanded Memory Specification Emulator, EMS), esimerkiksi 386MAX tai 386LIMIT. Windows/386 sisältää oman EMS-emulaattorinsa, joka automaattisesti muuntaa kaiken käytettävissä olevan extended-muistin expanded-muistiksi.

Oli sitten kyseessä 80286-mikrossa toimiva EMM-muistinhallintaohjelma tai jokin EMS-simulaattori, se luo systeemiin loogisen reiän (hole). Tämän reiän kohdalle sijoittuu niin kutsuttu EMS-ikkuna (systeemiin suuressa määrin EMS-ikkunoista lisää tuonempana), jonka kautta järjestelmä pääsee käsiksi expanded-muistiin. Sivutustekniikan ansiosta EMS-ikkuna näyttää MS-DOSin mielestä olevan systeemiin alueella, vaikka sijaitseekin käytännössä aivan muualla.

Tietoa ei suinkaan siirretä fyysisesti EMS-ikkunan kautta näkyvän expanded-muistin ja systeemiin muistissa olevan reiän välillä, sillä se olisi aivan liian hidasta pelkan sivurekisterin käsittelemiseen verrattuna. 80286-koneissa muistinhallintaohjelma yksinkertaisesti vaihtaa laajennusmuistin sivurekisterin osoittamaan halutulle sivulle, joka sitten näyttää EMS-ikkunan läpi katsottuna sijaitsevan systeemiin muistissa. 80386-mikroissa sama hoituu 80386:n sisäänrakennetun sivutustoiminnon kautta.

Expanded-muistia on olemassa kaksi Lotus/Intel/Microsoft-määrittelyn (LIM) erittelemää tyyppiä: LIM 3.2 ja LIM 4.0.

LIM 3.2 mahdollistaa vain pienen, 64 kilotavun kokoisten EMS-ikkunoiden käytön sivutuksessa (small frame EMS). Systeemiin muistista on siis varattava 64 kilotavun kokoinen reikä tätä tarkoitusta varten. Pieniä EMS-ikkunoita voidaan käyttää vain tiedon varastointiin.

LIM 4.0:n sivut voivat olla teoriassa jopa yhden megatavun kokoisia (large frame EMS). Tämä mahdollistaa käytännössä useiden ohjelmien yhtäaikaista ajamisen, mikäli ne pidetään kurissa ja järjestyksessä esimerkiksi Windowsin toimesta. LIM 4.0:n EMS-ikkunat voivat sijaita myös perusmuistissa, tai kuten Windowsin

suhteen on asian laita, niiden voi olla jopa pakko sijaita perusmuistissa. EMS-ikkunoiden koko määritellään 16-kilotavun suuruisina paloina, joita kutsutaan myös EMS-sivuiksi.

Intel Above Boardin, DT215:n ja DT216:n muistinhallintaohjelmat tukevat vain LIM 3.2:ta, kun taas 386MAX, 386LIMIT ja Windows/386:n sisäänrakennettu EMS-emulaattori osaavat myös LIM 4.0:an vaatimat toimenpiteet.

Ylämuistialueelta 64 kiloa lisää tilaa

Ylämuistialuemäärittely eli HMA (High Memory Area) on osa Lotuksen, Intelin, Microsoftin ja AST Researchin yhteistyönä määrittelemää extended-muistimäärittelyä (eXtended Memory Specification, XMS). HMA kattaa XMS:n ylämuistialueella (1024 kt — 1088 kt) sijaitsevia muistilohkoja koskevat muistinvaraus-, vapautus- ja koonmuutosrutiinit

*WDRV-ajuriohjelman
on oltava uusi versio
eli sen päiväyksen
on oltava vuoden
1989 tammikuun
18:ttä tuoreempi.*

Jotta HMA:sta olisi jotain hyötyä, täytyy käyttää erityistä ylämuistialueen ajuriohjelmaa (high memory driver), joka antaa MS-DOS-sovellusten käyttää tätä extended-muistiin kuuluvaa aluetta. Näin MS-DOS-sovellukset voivat saada käyttöönsä 64 kilotavua ylimääräistä muistia — tai oikeastaan noin 50, sillä myös ajuriohjelma vie oman osansa. Sekä MS Windows että Lan Manager voivat käyttää ylämuistialuetta, mutta eivät kuitenkaan samanaikaisesti.

Käytettävä ajuriohjelma voi olla joko HIMEM.SYS, jolloin se on määriteltävä CONFIG.SYS:issä tai sisällytettynä EMS-emulaattoriin. Sekä 386MAX, 386LIMIT että Windows/386:n sisäänrakennettu EMS-emulaattori sisältävät ylämuistialueen käyttöön ottavan ohjelman eli CONFIG.SYSin ei niitä käytettäessä tarvitse sisältää mitään lisämäärittelyjä.

Ylämuistialueesta voi olla usein hyvin paljon hyötyä eli mikäli se on olemassa, se kannattaa myös ottaa käyttöön.

SMARTDRIVE — älykästä tiedonsiirtoa

SMARTDRIVE on Windowsin mukana toimitettava ajuriohjelma, joka mahdollistaa kätkömuistin (cache) käytön. Mikä tahansa levytiedostoja toistuvasti käsittelevä MS-DOS-sovellus voi käyttää sitä lyhentämään tietojen lukemiseen kuluvaa aikaa. Toiminnon käyttämiseksi CONFIG.SYSin on sisällettävä SMARTDRIVEN määrittely (SMARTDRV.SYS).

Koska Windows lataa jatkuvasti uusia segmenttejä, SMARTDRIVE:sta on Windows-sovellusten yhteydessä käytännössä aina hyötyä. SMARTDRIVE:lle varattavan muistialueen koon voi käyttäjä määritellä itse, ja se voi sijaita joko extended- tai expanded-muistissa. Valinta tapahtuu SMARTDRV.SYSin yhteydessä annettavalla valitsimella.

Esimerkiksi haluttaessa varata megatavun verran expanded-muistia SMARTDRIVE:lle, CONFIG.SYSin tulisi sisältää seuraava rivi:

```
DEVICE= <path> SMARTDRV.  
SYS1024 /a
```

missä valitsin /a tarkoittaa expanded-muistia. Jos halutaan määritellä extended-muistiin sijoittuvan SMARTDRIVEN kooksi 960 kilotavua, CONFIG.SYSiin kirjoitetaan seuraavaa:

```
DEVICE= <path> SMARTDRV.  
SYS960
```

/a -valitsimen puuttuminen tarkoittaa extended-muistia. Lisää tietoa SMARTDRIVE:sta löytyy esimerkiksi Windows User's Guidesta.

RAMDRIVEstä ripeä levyasema

RAMDRIVE toimitetaan SMARTDRIVEN tapaan Windowsin mukana. Se varaa osan muistista ja mahdollistaa kyseisen alueen käytön kiintolevyaseman tapaan. Tällainen työmuistilevy (RAMdrive) toimii kuitenkin huomattavasti kiintolevyä nopeammin sekä tyhjenee aina virran katketessa. Niinpä työmuistilevyn sisältämät säilytettäväksi tarkoitettut tiedot on kopioidava ennen koneen sammuttamista työmuistilevykkeeltä todelliseen kiintolevyille tai levykkeelle.

Työmuistilevyn määrittely (RAMDRIVE.SYS) sisällytetään

Windows 2.11



tavalliseen tapaan CONFIG.SYS-tiedostoon. Työmuistilevyn koko voidaan määritellä käyttäjän tarpeiden mukaan, ja se sijaitsee extended-muistissa, vaikkakin jotkin uusien Windows-sovellusten yhteydessä toimitetut versiot osaa- vat käyttää myös expanded-muistia, mutteivät kuitenkaan Windows 2.11:n alaisuudessa.

Lisää tietoa työmuistilevystä ja RAMDRIVEstä voi etsiä Windows-levykkeiltä löytyvästä RAM-DRIVE.TXT-tiedostosta.

Valmistuspäiväys takaa tuoreuden

Mikäli 80386-koneesta löytyy SCSI-massamuistin sovitinohjelma eli halutaan käyttää 80386--

suoritinta virtuaalikonetilassa (Virtual Machine Mode, esimerkiksi Windows/386 sekä 386MAX ja 386LIMIT käyttävät sitä), CONFIG.SYSin on sisällettävä WDRV-ajuriohjelman määrittely.

Mikäli käyttäjä ei ole varma siitä, onko koneessa SCSI-sovitinta vai ei, hän voi tarkistaa asian koneen käynnistyksen aikana. Näytölle ilmestyy nimittäin alustusvaiheessa ilmoitus sovitinien läsnäolosta, jos SCSI on käytettävissä.

WDRV-ajuriohjelman on oltava uusi versio eli sen päiväyksen on oltava vuoden 1989 tammikuun 18:tta tuorempi.

EMS-emulaattoreita kaksin kappalein

386MAX on EMS-emulaattori, joka sallii MS-DOS-ohjelmien käyttää extended-muistia expanded-muistin tapaan. Se sisältää LIM 4.0:n mukaiset toiminnot eli osaa käyttää isoja EMS-ikkunoita.

386MAX toimii vain 30386-pohjaisissa koneissa, ja CONFIG.SYSin on sisällettävä 386MAX.SYS-ajuriohjelma. Koska 386MAX.SYSin on oltava käytettävissä ennen muita expanded-muistia käyttäviä ajuriohjelmaa, sen määrittelyn on oltava CONFIG.SYSissä ennen muita.

386MAX ja 386LIMIT ovat MS-DOS 4.0:lle saatavilla olevaan EMM386.SYSiin verrattuina suositeltavia vaihtoehtoja, sillä ne vievät huomattavasti vähemmän muistitilaa. 386MAXin valmistaja on Qualitas Inc., P.O. Box 386, Cabin John, MD 20818, USA. Puh, int +1-301-469-8848. Fax, int +1-301-469-5810

386LIMIT on itse asiassa yhteneväinen 386MAXin kanssa, paitsi että sen ominaisuuksia on riisut- tu etenkin ajuriohjelmien lataami- sen ja systeemimuistissa sijaitse- vien TSR-ohjelmien osalta. 386LIMIT sisältää kuitenkin kaik- ki Windows-järjestelmän määrit- teleminen kannalta oleelliset toi- minnot.

Hiiret myös mukaan kuvioihin

Useimmat Microsoftin hiirtä käyt- tävät ohjelmistot tarvitsevat toi-

miakseen MOUSE.SYS-ajurioh- jelman, joka on sisällytettävä CONFIG.SYSiin. Ajurin aseme- ta voidaan käyttää varsinaista hiiriohjelmaa MOUSE.COM, joka poistuu muistista varsinaisen so- velluksen suorituksen loppuessa. Windows sisältää kuitenkin oman hiiriajuriohjelmansa eli Window- sin yhteydessä ei tarvita mitään erityistoimenpiteitä hiiren käyttä- miseksi.

Miten Windows-sovellus todella toimii

Vaikka Windows-sovellus koostuu MS-DOS-sovelluksen tapaan yh- destä linkatusta EXE-tiedostosta, se sisältää itse asiassa useita erilli- siä osia, joista kullakin on oma merkityksensä. Tästä seuraa, että Windows-sovellusta ladattaessa koko ohjelmatiedostoa ei välttä- mättä haeta muistiin kerralla. Yksi tai useampia kulloinkin suoritetta- via ohjelman osia (koodisegment- tejä) ladataan muistiin tarpeen niin vaatiessa. Tarpeettomat seg- mentit pysyvät levyllä kunnes niitä tarvitaan. Poistettaviksi merkitty- jen segmenttien viemä muistitila otetaan uuteen käyttöön, jos uu- det segmentit eivät muuten mahdu muistiin.

Ei siis ihme, että SMARTDRI- VESTA on niin paljon hyötyä Windows-sovellusten yhteydessä. Koska Windows-ympäristössä la- dataan jatkuvalla syötöllä uusia segmenttejä, mahdollisuus ladata osa niistä suoraan kätkömuistista nopeuttaa ohjelman suoritusta huomattavasti.

Tämän segmentoinnin kautta Windows-sovellukset voivat olla suurempia kuin käytettävissä ole- va perusmuistiavaraus. Kyse ei siis suinkaan ole expanded-muistissa pullistelevista megatavuista vaan jatkuvasta koodinpätkien lataa- misesta ja poistamisesta.

Mihin ihmeeseen EMS:ää sitten tarvitaan?

Kuten muistetaan, LIM 3.2:n mu- kaisten 64 kilotavun kokoisten EMS-ikkunoiden kautta ei voida tehdä muuta kuin sijoittaa ohjel- mien tarvitsemia tietoja expan- ded-muistiin. LIM 4.0:n isot EMS-ikkunat sen sijaan mahdol- listavat myös ohjelmien sijoittami-

Windows, joka käyttää isoja EMS-ikkunoita

100000h
=1024 kt

F000h
=960 kg

A0000h
(=640 kg)

ROM BIOS

Toissijainen
kartoitusalue MA2

Näyttömuisti
NäyttöROM

Pääkartoitusalue
MA1

Windowsin omaan
käyttöön varaama
muistialue

DOS, ajurit, TSR:t

systeemimuisti

perusmuistialue

sen expanded-muistiin. Moiseesta mahdollisuudesta on Windowsin yhteydessä huomattavasti hyötyä.

Windows kykenee siis ajamaan perusmuistia suurempia sovelluksia. Mitä sitten käy, jos Windows yritetään saada käynnistämään uusi sovellus, kun edellinen on vielä käytössä?

Jos muistia riittää, toinen sovellus ladataan ensimmäisen sovelluksen rinnalle perusmuistiin. Tarpeen niin vaatiessa osa ensimmäisen sovelluksen turhiksi merkityistä segmenteistä poistetaan tilan vapauttamiseksi. Järjestelmä toimii siis aivan samalla tavalla kuin yhtä sovellusta ajettaessa.

Jos tilaa ei löydy riittävästi, käyttäjä saa virheilmoituksen: "Not enough memory to run application". Näin ei kuitenkaan tarvitse tapahtua, jos käytettävissä on LIM 4.0:n mukainen EMS isoine EMS-ikkunoineen. Ne kun voivat toimia perusmuistin laajennuksina.

Pienet ja isot EMS-ikkunat erikseen

Pienten 64 kilotavun kokoisten EMS-ikkunoiden käyttö onnistuu siis sekä LIM 3.2:n että LIM 4.0:n avulla. Windows itse ei kuitenkaan tee käytännössä mitään tiedon varastoimiseksi expanded-muistiin, vaan sovelluksen täytyy itse käyttää pieniä EMS-ikkunoita käsitelläkseen expanded-muistissa sijaitsevia tietoja.

Isojen EMS-ikkunoiden käsitteily onnistuu siis joko 386MAXilla, 386LIMITillä, Windows/286:lla tai Windows/386:lla.

Isoja EMS-ikkunoita käytettäessä Windows varaa osan muistista pitääkseen kirjaa siitä, missä on mitään. Tämä kirjanpitoalue jakautuu edelleen kahtia. Pääkartoitusalue (Mapping Area 1, MA1) sijaitsee perusmuistialueella ja toinen (MA2) systeemimuistin puolella. Kuva 1 havainnollistaa tilanetta.

Yhteensä nämä kaksi aluetta vievät MA kilotavua muistia.

MA = MA1 + MA2

Mitä sitten tapahtuu, kun käyttöön otetaan muistissa jatkuvasti pysytteleviä ohjelmia (memory resident program) kuten erilaisia ajureita?

Windowsin latausosoite kasvaa, ja koska Windowsin itselleen varaaan muistin määrä on Windows-versiota kohden vakio, MA1 (eli MA) pienenee. MA pienenee myös MA2:n kutistumisen myötä esimerkiksi silloin, kun osa systeemimuistista on varattu lisäkorttien lukumuisteille. MA ei kuitenkaan voi kutistua kuin tiettyyn rajaan asti:

Jos kartoitusalue (MA) pienenee liikaa eli alle Windows-versiosta riippuvan vakioarvon, Windowsin kyky käyttää isoja EMS-ikkunoita katoaa (kartoitu-

saluetta ei enää ole).

Windows määrittelee latauksen yhteydessä sen, käytetäänkö isoja EMS-ikkunoita eli ei. Päätös riippuu perus- ja systeemimuistin puolella vapaana olevan muistin määrästä.

Jos Windows päättää olla käyttämättä isoja EMS-ikkunoita, se voi käyttää vain perusmuistia sovellusohjelmien ohjelmakoodin sijoittamiseen. Tuloksena on ymmärrettävästi melkoinen suorituskyvyn pudotus (vain yksi suuri sovellus voi olla kerrallaan käytössä).

Tässä yhteydessä on tärkeää muistaa, että käyttäjä ei suoraan ole tietonen siitä, käyttääkö Windows isoja EMS-ikkunoita vai ei. Tilanne käy ilmi vasta siinä vaiheessa, kun hän yrittää käyttää esimerkiksi Excel 2.1:tä ja PageMaker 3.0:aa yhtä aikaa. Jos käytössä ei ole isoja EMS-ikkunoita, kyseisten sovellusten yhtäaikaisen hyödyntämisen voi suosiolla unohtaa.

Eli Windowsilla tulisi olla riittävästi muistitilaa sovellusten kartoittamiseen sekä perus- että systeemimuistin puolella.

Windowsin kahtiajakaisuus pienten ja isojen EMS-ikkunoiden suhteen pätee sekä Windows/286:een että Windows/386:een.

Pienet ja isot EMS-ikkunat yhdessä

Windows voi käyttää sekä pieniä että isoja EMS-ikkunoita yhtä aikaa. Muistitilan käytön optimoimiseksi pienen ikkunan tulisi osua päällekkäin suuren ikkunan toissijaisen kartoitusalueen kanssa. Tällöin Windows tietää miten hallita tilanne ja osaa ratkaista pienten ja isojen EMS-ikkunoiden välille mahdollisesti muodostuvat ristiriidat.

Systeemimuistikartta

100000h	ROM BIOS	1024 kt
F0000h	Reikä	960 kt
D000h	Reikä	832 kt
CE000h	Madge I RAM buffer	824 kt
CDC00h	Reikä	832 kt
CC000h	Reikä	816 kt
C8000h	SCSI ROM	800 kt
C6800h	VGA ROM	794 kt
C6000h	2 kt reikä (älä käytä)	792 kt
C0000h	VGA ROM	768 kt
A0000h	VGA RAM	640 kt

Tämän artikkelin tiedot pätevät seuraaviin Nokian tietokoneisiin:

80286	MikroMikko 3 (ASC)
	MikroMikko 3TT (AWS)
	PC286, WS286 (Nokian vanhat 286:t)
	Alfaskop DT215, DT216 (uudet 286:t)
80386	MikroMikko 3/386
	MikroMikko 3TT/386
	Alfaskop DT326, DT336, DS338, DS348

**TKK:n
verkoista
löytyy
mylly
joka
lähtöön**

Työasemasta superiin

Kerrotaan, että jos maailmassa on valmistettu kaksi kappaletta mikroja, toinen niistä on Teknillisellä korkeakoululla Otaniemessä. Tarinat ovat tarinoita, mutta skaalaa riittää tyhmistä paperipäätteistä työasemien kautta aina naapurissa VTKK:ssa seisovaan superkone Crayhin asti. Ja kaikki on saatu verkotettua toimimaan kauniisti yhdessä.

Kaikki ei ole kuitenkaan ollut niin yksinkertaista kuin mil-tä se ensi lukemalla vaikut-taa, sillä nykytilanteeseen pääse-miseksi on tehty melkoinen työ-rupeama eri tahojen tarpeiden selvittämiseksi ja tietoturvan ta-kaamiseksi. Siinä missä verkko-jen on annettava opiskelijoille pe-rusedellytykset opintojensa läpi-viemiseksi, niihin kytketyn super-koneen käytön on oltava tutkijoi-lle mahdollisimman sujuvaa.

Verkko & verkko

Verkkoja on itse asiassa kaksi. Nelikierteisiin pohjautuva data-verkko hoitaa pääteyhteydet, ja osittain optisesti toteutettu Ether-net yhdistää korkeakoulun tieto-koneet. Laskentakeskuksen labo-ratorioinsinööri Markku Suva-nen kertoo, että TKK:n Laskenta-keskuksesta lähtee kahdeksan va-lokaapelia Otaniemen tärkeim-piin rakennuksiin. Valokaapelit haaraautuvat edelleen tavallisiksi Ethernet-kaapeleiksi, jotka yh-distävät verkkoon kaiken kaikkiaan 17—18 korkeakoulun raken-nusta.



Teknillisellä korkeakoululla on käytössä toista tuhatta tavallista ja insinööriyöasemaa. Kaikki mikrot eivät ole kuitenkaan vielä verkossa, koska turvallisuussyistä ei haluta rakentaa useita suoria verkkoyhteyksiä.

Ulospäin yhteydet hoituvat FUNETin, suomalaisten korkeakoulujen ja tutkimuslaitosten yhteisen verkon kautta. FUNETin avulla saadaan yhteys paitsi muihin suomalaisiin tutkijoihin, opiskelijoihin ja asiantuntijoihin myls ulkomaisiin verkkoihin, sillä FUNETista pääsee pohjoismaisen NORDUnet-korkeakouluverkon kautta esimerkiksi EUnetiin, UUCP-verkon Euroopan segmenttiin asti.

FUNET mahdollistaa muun muassa kansainvälisen sähköpostin lähettämisen, USENET-uutisten lukemisen, tietokone-neuvottelut, tiedostojen siirron ja verkossa olevien koneiden etäiskäytön. Palveluja on itse asiassa niin paljon, että moni TKK:n opiskelija ja tutkija on huomannut sompailevansa verkossa monta tuntia päivässä. Pelkästään uutisten lukemiseen saa helposti hukattua kokonaisen iltapäivän.

FUNETin .fi-loppuiset käyttäjä-tunnukset tulevat myös USENETin uutisissa usein vastaan eli kansainvälistä keskustelua paitsi seurataan myös osallistutaan ahkerasti.

FUNET tukee useita eri arkkitehtuurivaihtoehtoja mukaan lukien DECnetin, TCP/IP:n ja BSC-linjoilla käytetyn NJE:n. Verkosta löytyy jos jonkinmoista bitinpyörintä mukaan lukien PC-yhteensopivat, VAXit, Sunit, Apollot, HP:t, CDC:t, Honeywellit, AT/T:t, Altosit sekä tietysti uusi superkoneemme Cray X/MP EA 416.

Mikrot työryhmäverkoiksi

Teknillisen korkeakouluun on hankittu Ylioppilaskunnan aloitteesta kuusi noin kymmenen mikron itsepalveluluokkaa. Tarpeita kartoitettaessa TKK:n laitokset toivoivat PC-tyyppisiä mikroja, joista piti löytymän peruspalvelut ja tarpeeksi kapasiteettia vuosien varrella hankittujen ohjelmistojen pyörittämiseen. Hankkeen koordinoi Laskentakeskus.

"Superkoneen käyttö työasemalta on joustavaa. Ellei verkkoa olisi välissä, tutkija voisi ajatella, että hänen oma työasemansa on superkone", osastopäällikkö Olli Serimaa Tieteellisen laskennan palvelusta toteaa.

Laskentakeskuksen johtajan Kristel Sarlinin mukaan tekniikan opiskelussa on tapahtumassa rakennemuutos. "emmin tietotekniikkaan tutustuminen aloitettiin ohjelmointikielten opettelusta. Nyt lähdetään mikrojen käyttöön tutustumisesta ja opetellaan hyödyntämään niitä opiskelun työvälineinä. Tekstinkäsittelyä, taulukkolaskentaa ja sähköpostia käytetään opiskelun apuna alusta alkaen, ja Laskentakeskus huolehtii siitä, että tarvittavat resurssit ovat olemassa."

Tietoturvan on pidettävä

Tietojenkäsittelykapasiteetin ohella mikroluokkahankkeen toteutukselta vaadittiin ennen kaikkea turvallisuutta. Työasemilta ei saa päästä vakoilemaan runkoverkon liikennettä, eivätkä paikalliset palvelut tai mahdolliset häiriöt saa mitenkään häiritä runkoverkon toimintaa. Niinpä mikrot yhdistetään työryhmäverkoiksi, joiden runkoverkon suuntaan tapahtuva kommunikaatio hoidetaan keskitetysti palvelimen välityksellä.

Mikroja testannut Vesa Ylä-Jääski kertoo, että mikroluokkien verkot rakentuvat standardoitujen tietoliikenneyhteyksien varaan. Käyttäjärjestelmänä on Ethernetissä olevien koneiden tapaan UNIX eli verkkojen käytön ei pitäisi tuottaa liiemmästi ongelmia. Korkeakoululla vallitsee jo ennestään hyvin vahva UNIX-kulttuuri.

Luokkien mikrot ovat Nokian 80386-pohjaisia MikroMikkoja.

TKK ei kuitenkaan halua sitoutua yhteen valmistajaan, vaan saada erilaiset UNIX-käyttöjärjestelmää käyttävät koneet toimimaan verkon kautta yhteistyössä.

"Vastaavaa näin laajaa mikroverkkoratkaisua ei Suomessa ole, mutta otimme riskin. Hankintaa edelsivät kuitenkin perusteelliset vertailut, joissa erilaisten ratkaisujen toimivuutta ja kuormitettavuutta testattiin eri näkökulmista", Sarlin, Suvanen ja Ylä-Jääski toteavat.

Sunit Crayn ympärillä

Tieteellisen laskennan palvelu on Valtion tietokonekeskuksen tiloissa Otaniemessä toimiva 40 tutkijan yksikkö, joka pitää huolta superkone Cray X/MP:stä ja opastaa sen käyttäjiä. Menetelmäosaston päällikkö Olli Serimaa määrittelee Tieteellisen laskennan palvelun kaikkien korkeakoulujen, Ilmatieteen laitoksen ja VTT:n tutkijoiden yhteiseksi laskennalliseksi supertietokonekeskukseksi.

Vuosi sitten käyttöön otettu superkone eroaa tavallisesta suurtietokoneesta vektoriarkkitehtuurinsa takia. Kone voi suorittaa useita samantapaisia numeerisia toimenpiteitä yhtä aikaa rinnakkain.

"Cray on tutkimuksen erikoisinstrumentti niille, jotka tarvitsevat numeronmurskausta. Sitä ei kannata käyttää mihin tahansa", Olli Serimaa muistuttaa. Niinpä kaikki valmistelevat toimenpiteet tehdään muualla kuin Crayssä.

Tutkija voi käyttää Craytä esimerkiksi työasemaltaan, jolla hän tekee kaikki valmistelevat toimenpiteet ennen kuin lähettää työn FUNETin välityksellä Craylle. Toinen vaihtoehto on tehdä ennakkovalmistelut suoraan Crayn edustakoneella.

Crayn laskemat tulokset ovat epähavainnollisia numerosarjoja, jotka on viimeisteltävä ja jatkokäsiteltävä erikseen esimerkiksi työasemalla graafiseen muotoon ja edelleen diaksi tai videoksi esitystä varten.

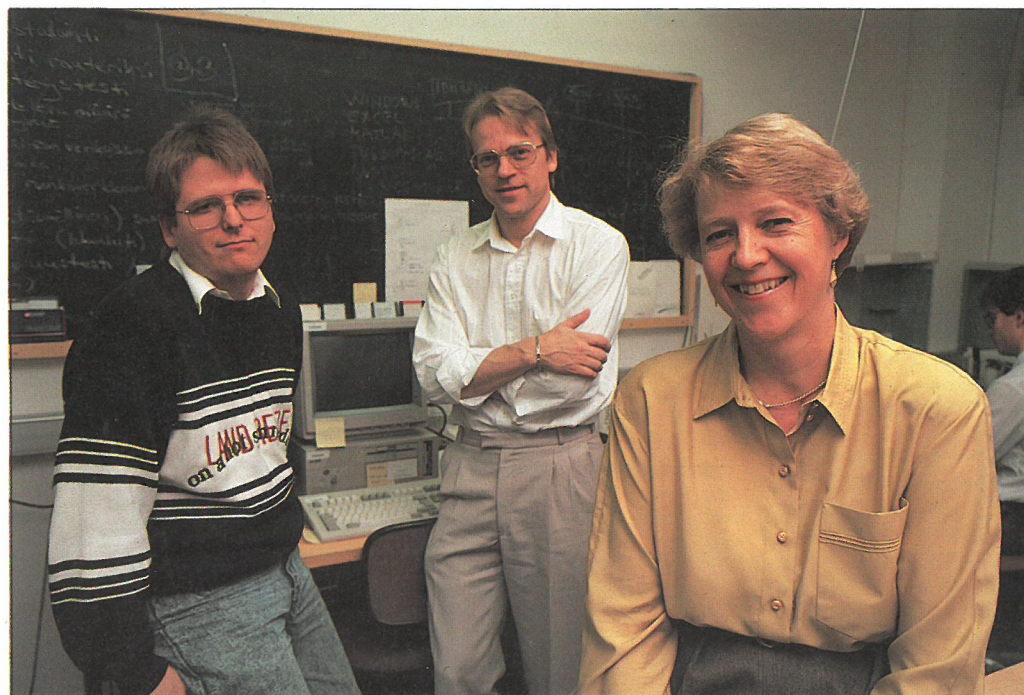
Toinen Crayn edustakoneista ja kaksi sen työasemista ovat Suneja. Olli Serimaa selostaa laitevalinnan syitä: "Sunista oli kokemuksia ulkomailta. Se tiedettiin toimivaksi, luotettavaksi ja helpohoitaiseksi ratkaisuksi. UNIX-ympäristöön oli saatavilla myös sopivia ohjelmistoja." Hän toteaa superkoneen käytön olevan työasemilta käsin niin joustavaa, että ellei verkkoa olisi, tutkija voisi luulla työasemaansa superkoneeksi.

Aika on kortilla

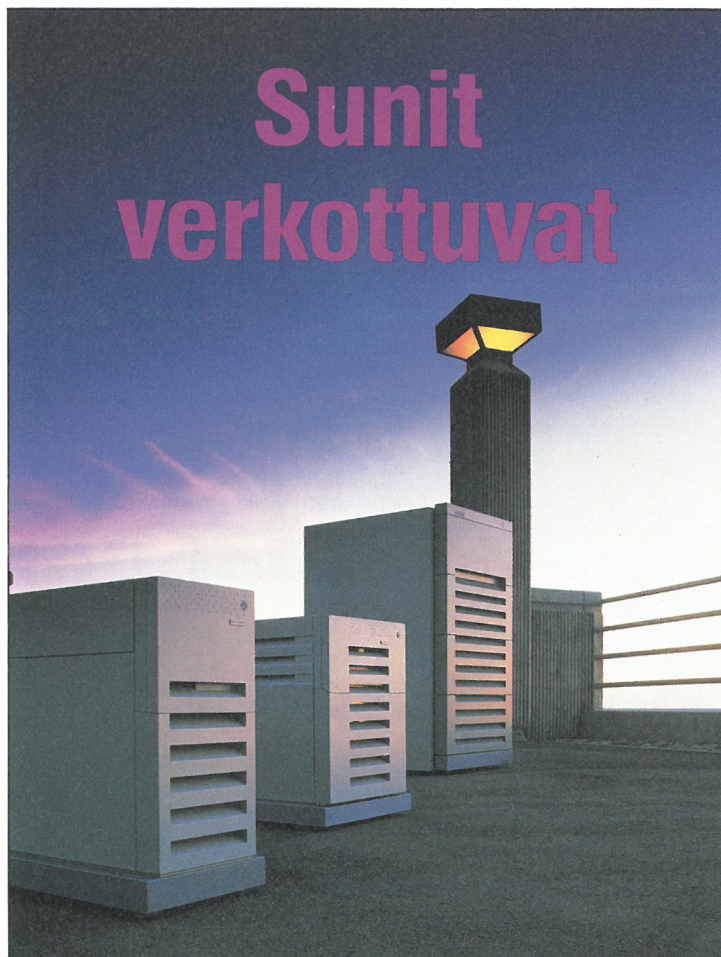
Superkonetta käyttävät erityisesti fyysikot, kemistit, insinöörit, tähtitieteilijät sekä perusetta soveltavaan tutkimukseen. Soveltavasta tutkimuksesta tyypillinen esimerkki on sääennustuksen teko tai sillan lujuuslaskenta. Tähtitieteilijät ja fyysikot tekevät puolestaan tyypillisesti perustutkimusta, testaavat esimerkiksi numeerisesti maailmankaikkeuden syntymälle. Cray onkin helpottanut käyttäjiensä tietojenkäsittelytilannetta huomattavasti, sillä aiemmin pienikin lujuuslaskelma saattoi tukehduttaa esimerkiksi jonkin TKK:n koneen pitkäksi aikaa.

"Luonnontieteiden tutkijoilla on vuosikauden kokemus tieteellisestä laskennasta, ja esimerkiksi perustutkimus aina vain isompi malleineen täyttäisi vaikka kuinka monta superkonetta. Haluamme kuitenkin kannustaa myös muita aloja kokeilemaan superkoneen käyttöä", Serimaa toteaa. Eritoten poikkitieteellistä tutkimusta toivotaan syntyvän. Superkonetta kokeilevia aloja ovat esimerkiksi kielitiede ja taloustieteet, myös taideteollisuutta se on kiinnostanut.

Crayn aika on varattu käyttöön sataprosenttisesti: runsas puolet menee korkeakoulujen tutkijoille, neljännes Ilmatieteen laitokselle ja 15 % VTT:lle. Loput viisi prosenttia myydään teollisuudelle. Aktiivisia konetta käyttäviä projekteja on jo yli 50. ■



Sata megabittiä sekunnissa ...



Sunin tuotejulkistukset eivät rajoitu pelkästään työasemiin ja SPARC-arkkitehtuuriin, sillä markkinoille on tullut peräti viisi uutta Sun-verkkotuotetta, jotka laajentavat jo ennestään runsasta Sun-verkkotuotevalikoimaa.

Uudet tuotteet paitsi parantavat Sunin työasemia sisältävien verkkojen ominaisuuksia myös mahdollistavat sen yhdistämisen IBM-maailmaan. Lisäksi markkinoille on kevään mittaan tulossa uusi käyttöliittymäkehitin, jota käytettäessä ei tarvitse kirjoittaa ohjelmavirheen patkakaan.

Verkot haltuun

SunNet Manager on verkonvalvontaohjelmisto, joka on tarkoitettu TCP/IP-tietoliikenneprotokollaa noudattaviin hajautettuihin työasemaverkkoihin. Se valvoo, kuinka verkon resurssit toimivat, hälyttää virhetilanteiden ilmetessä ja varoittaa niiden uhasta, oli kyse sitten tietokoneista, silloista, verkkosovelluksista

tai palveluista. Lisäksi SunNet Managerin avulla on mahdollista liittää verkkoon uusia protokollia ja työvälineitä.

SunNet Licensen avulla voidaan puolestaan ostaa ohjelmistolisenssi esimerkiksi kymmenelle työasemalle, vaikka ohjelmistoa käytettäisiin 20 työasemassa. Jokaisella ohjelmiston käyttäjälähän on käytännössä oltava oma lisenssinsä. SunNet License rajoittaa kuitenkin samanaikaisten käyttäjien määrän maksettujen lisenssien mukaan. Lisäksi se on ainoa markkinoilla oleva tuote, joka noudattaa NCF-määrityksiä.

FDDI = 10*Ethernet

Pelkkä VME-kortti ja sopiva ohjelmisto riittävät kymmenkertais-

tamaan verkon kapasiteetin. Kyseessä on FDDI/DX, yksi ensimmäisiä tuotteita, jonka avulla Sun-käyttäjät pääsevät käsiksi FDDI-verkkostandardin sisältämiin ominaisuuksiin kuten virheenkorjaukseen ja parantuneeseen tietoturvaan.

FDDI-verkon kapasiteetti on kymmenkertainen nykyiseen Ethernetiin verrattuna, sillä FDDI:n nopeus on sata megabittiä sekunnissa Ethernetin kymmeneen verrattuna. FDDI:n kaksoisverkoarkkitehtuurin ansiosta verkko voi myös eristää virhepaikat ja reitittää tiedon kiertämään viallisen kohdan.

Yhteys isoon siniseen

SunLink Channel Gatewaytä tarvitaan silloin, kun Sunin työasemia jostain syystä halutaan liittää IBM:n keskuslaitteistoihin. Sen avulla niin tiedostot kuin viestitkin liikkuvat paikasta toiseen, minkä lisäksi koneet voivat jakaa myös käskyjä ja tehtäviä.

SunLink käyttää sekä vuorovaikutteista että erätiedonsiirtoa eli sen kautta useat IBM-pääteemulaattorit voidaan kytkeä suoraan Sunin järjestelmiin.

Käyttöliittymäkehitin

GUIDE (Graphical User Interface Design Editor) on graafisten käyttöliittymien suunnitteluun soveltuva työvälineohjelma, joka laajentaa Sunin OpenWindows-sovellusympäristöä.

Ohjelma luo käyttöliittymäkoodin automaattisesti, ja kehitys tapahtuu ikonivalikon ja hiiren avulla. Sen avulla voidaan rakentaa ja kokeilla erilaisten OPEN LOOK -käyttöliittymien ulkonäköä kirjoittamatta riviäkään varsinaista ohjelmakoodia. Koska GUIDE on myös integroitu OpenWindowsin työvälineisiin, sillä suunnitellut sovellukset voivat ladata tiedostoja poimimalla. Käyttäjä yksinkertaisesti valitsee ikonin ikkunaan ja sovellus latautuu automaattisesti.

GUIDE on saatavissa vuoden 1990 alkupuolella.

Lisätietoja: tuotepäällikkö Pertti Manner, puh. (90) 5671. ■

Kantataajuus-modeemi

Nokia Datsiirto on tuonut markkinoille uuden 2-johdintekniikkaa käyttävän kantataajuus-modeemin nimeltä BB 160-2W. Sen nopeusalue on 32–160 kilobittiä sekunnissa, ja se hyödyntää kaiunpoistotekniikkaa.

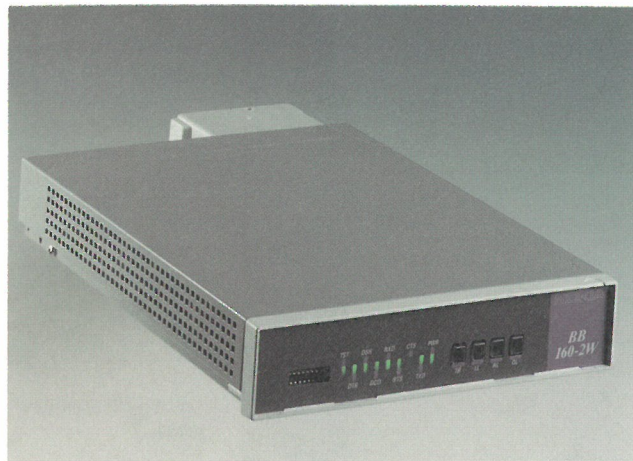
Pienillä nopeuksilla siirtoetäisyydet voivat olla yli kuusi ja suurimmallakin nopeudella yli neljä kilometriä tavallisia puhe-linkaapeleita käytettäessä.

Modeemin avulla voidaan sovitaa hitaita datsiirlaitteita v. 110-suosituksen mukaisesti esi-

merkiksi PCM-kanavaan. Pääte-laiteliitäntä voi olla joko v.24/v.28-, v.11-, v.35-, x.21- tai G.703-suosituksen mukainen. Myös korttimodeemin TTL-tasoinen liitäntä kelpaa.

Modeemia toimitetaan sekä koteloituna että kehikkoon asennettavana. Se sopii käytettäväksi kaikilla kiinteillä, kaksisuuntaisilla, tahdistetuilla, pisteestä pisteeseen yhteyksillä. Testausvaihtoehdot sisältävät muiden muassa sekä paikallis-, kauko- että digitaalisilmukan sekä v.52-suosituksen mukaisen itsetestin.

Lisätietoja: Matti Hyvärinen, puh. (90) 5671.



ES-verkko opettaa

Nokian ES-opetusverkkoon kuuluu opettajan työasema, enintään 20 oppilastyöasemaa ja kolme oheislaitetta.

Harjoitusta vetävä opettaja voi lähettää oman näyttönsä sisällön kaikkien tai yksittäisten oppilastyöasemien ruuduille, joilloin niiden näppäimistöt ja hiiret lukkiutuvat. Grafiikan siirto onnistuu myös toiseen suuntaan, sillä myös oppilastyöaseman näytön kuvan voi siirtää sekä valituille oppilaille että opettajalle. Siirtyvä kuva on täysin VGA-standardin mukainen kaikkine 262144 väreineen.

Opettaja voi ohjata oppilaiden huomiota myös karttakeppikoh-

distimen avulla, joka näkyy kaikkien oppilastyöasemien näyttöillä. Opettaja voi myös lähettää tietoa opetusverkon väylälle, josta oppilas saa sen halutessaan näyttölensä.

Opettaja voi seurata luokan työskentelyä selaamalla oppilastyöasemien näyttöjä joko automaattisesti tai ohjatusti. Yksilöllistä opetusta antaessaan opettaja käyttää oppilastyöasemaa omalta työasemaltaan, jolloin oppilaan näppäimistö ja hiiri lukittuvat. Ottamalla kaikki oppilastyöasemat kerralla haltuun, opettaja voi käynnistää opetussovellukset kaikissa koneissa yhtä aikaa.

Lisätietoja: Eila Hilpi puh. (90) 5671.



Helmi hoitaa hampaat

Helmi on järjestelmä, joka hoitaa hammaslääkärin vastaanoton ruutiinit kuten laskutuksen, potilas-kortiston, ajanvaraukset, varastonvalvonnan, palkanlaskennan jne.

Ohjelmisto soveltuu sekä yksityis-, terveyskeskus- että työterveyshammaslääkäreiden tarpeisiin. Tietoturvan takaamiseksi Helmin kullakin käyttäjällä on oma tunnuksensa ja salasanasensa, joilla hän pääsee käsittelemään omien potilaidensa tietoja.

Vaikka potilaan työstatus jatkuu kehittyäkin, esimerkiksi tarkastusstatus ja parodontologiset tiedot voidaan haluttaessa säilyttää ennallaan, sillä kaikki sta-

tukset ovat erillisiä. Statustietoihin voidaan merkitä tarkasti esimerkiksi maito- ja ylilukuiset hampaat, hemisektiojuuret, juurihoidot, ...

Hoidon kirjaamiseen voidaan käyttää joko lyhenteitä tai SV-kodeja, ja järjestelmä hakee itse loput tiedot hoidon hinnan.

Potilas voidaan kutsua myös uusintatarkastukseen recall-toiminnon avulla. Esimerkiksi tietyn kuukauden aikana kutsuttavien potilaiden luettelo ja tarrat ovat koska tahansa tulostettavissa.

Järjestelmä toimii MS-DOS ja UNIX/XENIX-ympäristöissä, ja se voidaan ottaa käyttöön vaihteittain esimerkiksi laskutuksesta tai ajanvarauksesta aloittaen. Alkuun päästään jo yksittäisen mikron avulla.

Kevään kurssiohjelma

Nokia Datan kurssikeskuksen Nitecin kurssiaikataulut keväälle 1990 ovat ilmestyneet. Omansa voi varata seuraavista numeroista:

Mikrokurssit: Marja Mäenpään, puh. (90) 567 2343.

Muut kurssit: Pia Kaartinen, puh. (90) 567 2545.

Piirikortteja freoneilla

Nokia Datan Kistan tehtailla koekillaan parasta aikaa keraamisten piirilevyjen valmistusta. Niiden valmistaminen vaatii vain puolet siitä toimenpidemäärästä, mitä vastaavien perinteisten korttien tekemiseen tarvitaan eli käytettävien kemikaalien määrä vähenee huomattavasti. Esimerkiksi otsonivaikutuksiaankin tuhoisammiksi kasvihuonekaasuiksi osoittautuneita freoneja ei tarvita enää lainkaan, ja muidenkin päästöjen määrä putoaa neljänneksen entisestään.

Keraamiset piirilevyt eivät ole pelkästään ympäristöystävällisiä, ne ovat myös pieniä, nopeita ja kestäviä. Keramiikka johtaa lämpöä sata kertaa tavanomaista epoksia paremmin, ja siitä valmistettujen korttien on todettu toimivan yhtä hyvin niin -35 asteisessa pakkasessa kuin 125 asteisessa uunissakin. Pienuus johtuu puolestaan keraamiseen piirilevyjen kerrosmaisesta rakenteesta. Kortissa voi olla jopa 60 kerrosta eli piirit kasvavat pysty- eikä vaakasuunnassa.

Neuraaliverkot veivät potin

Viime numeron lukijakyselyyn vastanneista ylivoimaisesti suurin osa valitsi likka Tuomen neuraaliverkkojuttun lehden parhaaksi artikkeliksi. Vain yksi oli sitä mieltä, että ei pitänyt koko jutusta.

Toiseksi tykkätyin oli puolestaan heti tuoteuutisten jälkeen sijoittunut ergonomiajuttu: "Mikro ei ole tuottavin investointi". Tästä numerotahan alan asiaa on turha etsiä, mutta seuraavasta sitten sitäkin enemmän.

Vähiten pidetty oli Tandemin ja Tiimin yhdistämisestä kertova juttu, mutta ei-pidettyin oli kuitenkin odotetusti tavallista pienemmälle kohderyhmälle suunnattu Adakurssi, jonka kohdalla hajonta oli vastaavasti melkoinen.

Ja t-paidat lähtevät tällä kertaa Reijo Honkaselle, Jussi Lakkalle, Ulla Kangasniemelle O. Maliselle ja Leila Honkaselle. ■

Mikko 100 on X/Open

Mikko 100 Unix-koneet ovat osoittaneet täyttävänsä X/Open-standardin vaatimukset läpäisemällä 2800 kokeesta muodostuvan testin. Tästä edes Nokia saa käyttää Mikko 100 -koneissaan X/Open-tuotemerkkiä.

00003 HELSINKI

Sopimus 00380/166

VASTAUSLÄHETYS

PL 780/10

Softari
Nokia Data

Nokia
maksaa
postimaksun

Softari 4/89

Palvelukortti

Softari-kysely

(Anna arvosana (—, 0 tai +) jutusta ja kerro, haluatko aiheesta lisätietoja)

	Sivu	—	0	+	Tahdon tietoa
Riittää tältä päivältä	2	☐	☐	☐	☐
SPARCstation 1 lupaa paljon	3	☐	☐	☐	☐
Valoa projektipääliikön elämään ..	6	☐	☐	☐	☐
Viitosten kaksintaistelu	10	☐	☐	☐	☐
High score	12	☐	☐	☐	☐
Learning Ada — Part 6	14	☐	☐	☐	☐
SQLWindows-kurssi — osa 4	18	☐	☐	☐	☐
Mikä mihinkin ja miksi	24	☐	☐	☐	☐
Työasemasta superiin	30	☐	☐	☐	☐
Sunit verkottuvat	30	☐	☐	☐	☐
Tuoteuutiset	31	☐	☐	☐	☐
Mikkonen	32	☐	☐	☐	☐

Vastanneiden kesken arvotaan viisi MM4-T-paitaa.

Jos voitaa paita, kokoni on: _____

- ☐ Korjaan yllämainitut osoitetiedot.
☐ Poistan yllämainitut osoitetiedot
☐ Lisään allamainitut osoitetiedot
☐ Netin ☐ Softarin rekisteristä/rekisteriin
☐ Haluan lehden kotiosoitteeseeni

yritys _____
 osasto _____
 lähiosoite _____
 postinro _____ postitoimip. _____
 nimi _____



8/0010

MIKKONEN

