

Softari

NOKIAN OHJELMISTOLEHTI 3/87





Kilpajuoksu ajaa eteenpäin

Ohjelmistotyössä on pitkään taisteltu. Hirvitetty hitautta, tuskiteltu työläyttä ja tuottamattomuutta. Tuhannet ovat Suomessakin köyryssä pääteidensä äärellä sitkutelleet ohjelmistopalan kerrallaan kasaan. Pitkäjänteisyys on ollut ohjelmoijalle asetettu vaatimus numero yksi – työn luonteen, mutta yhtä lailla työkalujen alkeellisuuden takia.

Ja kuitenkin ohjelmistotyön tulisi olla tuottavaa, kannattavaa teollisuutta ja ohjelmistojen aiempaa luotettavampia. Paine menetelmien ja apuvälineiden kehittämiseen on kova.

Yksinkertaiset relaatiotietokannat ovat edistysaskel. Relaatiotietokantojen käyttämiseen sopivat välineet, neljännen sukupolven kielet, sovelluskehittimet ja yleiskäyttöiset työkalut helpottavat tavanomaisten kaupallis-hallinnollisten ohjelmistojen tekoa.

On sovellusalueita, joissa ohjelmistot eivät saa pettää koskaan. Yksi apuväline ohjelmistojen laadun parantamiseen on Ada. Toinen käypä ratkaisu ovat vikasietoiset tietokoneet.

Ohjelmistoilta on alettu vaatia myös älykkyyttä. Ihmisen tietämyksen ohjelmointiin koneelle on oma kehittyvä tekniikkansa, tietämystekniikka.

Uusilla, mullistavillakin menetelmillä on puutteensa. Takapakkia on otettu ja otetaan ja tyhjää käydään, mutta sittenkin enemmän mennään eteenpäin. Uusien työkalujen kokeilijat eivät yleensä kadu.

Kun opitaan enemmän, myös vaaditaan enemmän. Sovellukset monipuolistuvat ja mutkistuvat.

Ainako yritetään enemmän kuin osataan?

Softari Nokian ohjelmistolehti

Päätoimittaja:
Hannu Markus puh. (90) 567 2982

Toimittaja:
Terhi Ogbeide puh. (90) 567 2981

Softari-tiimi:
Kari Kaipainen, Hannu Markus,
Terhi Ogbeide, Christer Pyyhtiä,
Pentti J. Savolainen, Ossi Syrjä,
Esko Uotila, Raija Visamo

Taitto:
Taina Sonninen puh. (90) 567 3260

Osoitteenmuutokset:
Jaana Lamberg puh. (90) 567 2980

Toimituksen osoite:
Nokia Informaatiojärjestelmät
Softari-lehti
Hiomotie 5
PL 780
00101 HELSINKI

Painosmäärä: 10 000

Kirjapaino: Vientipaino

Kansikuva: Jarmo Hietaranta

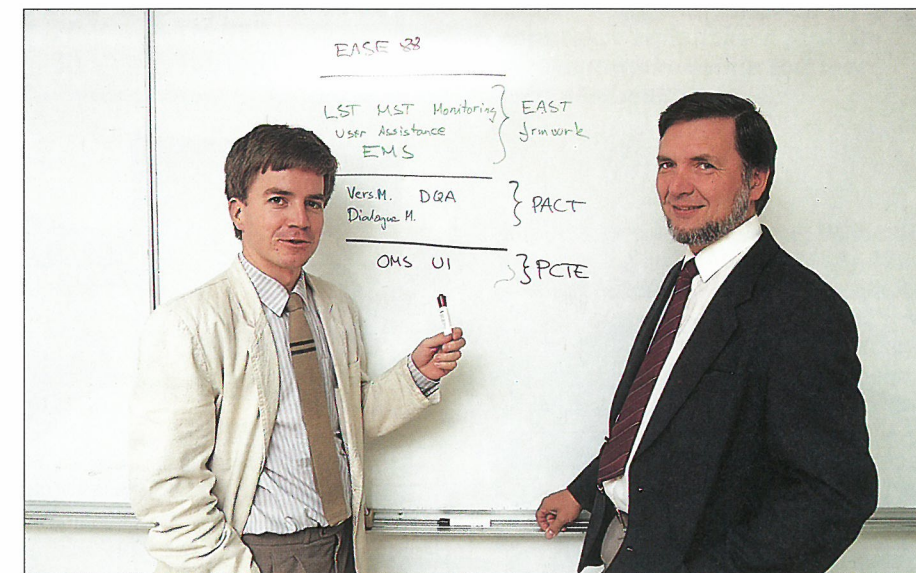
Julkaisija:
Nokia Informaatiojärjestelmät
ISSN 0782-8209

NOKIA

EASTistä yhteensopivat ohjelmointivälineet

Eureka-hankkeisiin kuuluva EAST-projekti (European Advanced Software Technology) on ollut käynnissä vajaan vuoden. Tarkoituksena on kehittää uusi, aiempaa tuottavampi ohjelmistokehitysympäristö.

Nokia Tutkimuskeskus on tekemässä ensimmäisinä EASTin osaprojekteina joukon selvityksiä.



EAST on 6 vuoden ja 600 henkilötyövuoden projekti, johon osallistuu yrityksiä Ranskasta, Tanskasta ja Suomesta.

Suomesta on mukana Nokia Tutkimuskeskus yhteistyökumppaninaan Tampereen teknillisen korkeakoulun tietojenkäsittelytekniikan laitos. Rahoitusta on saatu Tekesiltä.

Järkevä ohjelmistokehitysympäristö

Ohjelmistotyötä on tähän asti haitannut työkalujen irrallisuus. Samoja asioita on jouduttu tekemään yhä uudelleen ohjelmistotyön eri vaiheissa. EAST-tuote koostuu perusrakenteesta ja helppokäyttöisistä, yhteensopivista työkaluista, jotka voidaan yhdistää tuottaviksi ohjelmistokehitysympäristöiksi.

Työkalujen on tarkoitus sopia eri koneisiin ja niiden kanssa on pystyttävä käyttämään eri ohjelmointikieliä.

"Vuoden 1988 loppuun mennessä esitellään tuotteen ensimmäinen toimiva versio", kertoo Suomen EAST-osuuden projektipäällikkö **Petri Pöyhönen** Nokia Tutkimuskeskuksesta.

Pöyhösen mukaan tuotteissa hyödynnetään graafisen käyttöliittymän mahdollisuudet, joten ne myös rakennetaan käytettäväksi 32-bittisten grafiik-

EAST-projektissa kehitettyjä ohjelmistotyökaluja käytetään 32-bittisessä graafisessa työasemassa, joka on vielä tällä hetkellä outo ohjelmistokehityskone. "Se on kuitenkin huomisen MikroMikko", huomauttavat Matti Kilpi (oik.) ja Petri Pöyhönen, jotka vetävät EAST-hanketta Suomessa.

katyöasemien muodostamassa verkossa.

Pohjana käyttöjärjestelmäajennus

"Kaikkea ei ole tarkoitus tehdä itse. Aiemmin kehitettyjä, hyväksi havaittuja menetelmiä ja ohjelmistotyökaluja sulautetaan mukaan, missä mahdollista", Pöyhönen toteaa.

Ohjelmistokehitysympäristön perusrakenteesta on jo valmiina tärkeä palanen. Se on Esprit-projektissa rakennettu PCTE-käyttöjärjestelmäajennus (Portable Common Tool Environment). PCTE sisältää mm. objektitietokannan ja tarjoaa objektien hallinnan ja hajautuksen palvelut. Sen avulla saavutetaan riippumattomuus laitteista.

Kaikkia keinoja käytettävä

"Mitään taikakeinoja ohjelmistotyön tuottavuuden parantamiseksi ei ole

olemassa, eikä EAST:kaan sellainen ole. Ohjelmistojen teko on kallista ja kustannuksia on pakko hillitä. Kaikkia keinoja on käytettävä ja tehokas tietokoneavusteinen kehitysympäristö on yksi niistä", korostaa **Matti Kilpi**, Suomen EAST-projektin vastuullinen johtaja.

Määrittelyä ja selvityksiä

Suomesta EASTiin on osallistunut kymmenkunta henkilöä. He ovat osallistuneet tuotteen määrittelyyn ja tekevät nyt ensimmäisiä selvityksiä tietämystekniikan hyväksikäytöstä ohjelmointiympäristössä. Lisäksi on tutkittu uudelleenkäyttöä eli miten ohjelmia voidaan tuottaa jo olemassa olevista osista ja selvitetty OS/2:n vaikutusta ohjelmistomaailmaan.

Edelleen Suomessa on tutkittu ohjelmien visualisointimahdollisuuksia ja älykäästä uudelleenkäynnöstä, jossa ohjelmistoa muutettaessa ohjelmakoodin käänös voidaan rajoittaa minimiin. □

Ohjelmoijien uudet ulottuvuudet



Ohjelmointityössä siirrytään lapioiden aikakaudesta kaivinkoneiden aikakauteen. Neljännen sukupolven ohjelmistokehitysvälineet, relaatiotietokannat ja sovelluskehittimet, toimivat eri laite- ja ohjelmistoympäristöissä, parantavat ohjelmistotyön tuottavuutta ja helpottavat ylläpitoa.

E.F. Codd kirjoitti vuonna 1970 artikkelin "A Relational Model of Large Shared Data Banks", jossa hän ensi kerran määritteli relaatiomallin teorian ja peruskäsitteet. Tokkopa monikaan tuolloin tajusi, miten merkityksellinen tuo artikkeli oli. Nyt – seitsemäntoista vuotta myöhemmin – toteutetaan Coddin ajatuksia laajasti käytännössä.

Taulukoita ja sarakkeita

Relaatiomallissa tiedot esitetään taulukkomuodossa. Ohjelmoija tai päätteen ääressä istuva käyttäjä ei näe eikä

Relaatiotietokantojen alkutaivalta varjostaneet suorituskykyongelmat ovat väistymässä ja tuotteet kehittyvät juuri nyt rajusti. Eritasoisia relaatiotietokantatuotteita on runsaasti markkinoilla.

käytä minkäänlaisia indeksejä, osoittimia, hajauttimia tai muita "teknillisiä rasitteita". Hän operoi ainoastaan taulukkorakenteella käyttäen taulujen ja sarakkeiden nimiä.

Taulukoiden tehokas käsittely on relaatiomallin kantava ominaispiirre. Käsittely perustuu joukko-oppiin – muis-

tattehan joukko-opin, jota peruskoulun matematiikantunnilla jokin aika sitten kovasti opetettiin. Taulukoihin kohdistettavia joukko-opin operaatioita ovat esimerkiksi liitos, leikkaus, projektiio ja unioni.

Näitä matemaattisen täsmällisiä, mutta käytännössä ehkä hieman luotaantyöntäviä operaatioita ei tarvitse käyttää sellaisenaan. Ne toteuttaa jokin korkean tason ohjelmointikieli. Niistä SQL-relaatiotietokantakieli on ehdottomasti merkityksellisin ja levinnein.

Relaatiotietokannan perustaminen on hyvin nopeaa. Uusien taulukoiden lisääminen ja vanhojen muuttaminen on helppoa.

Relaatiomallin pelkistetty yksinkertaisuus, helpot muutokset ja etenkin kätevät käsittelyoperaatiot tekevät siitä ylivertaisen perinteisiin tietokantajärjestelmiin verrattuna.

Relaatiotietokantojen alkutaivalta varjostaneet suorituskykyongelmat ovat väistymässä ja tuotteet kehittyvät juuri nyt rajusti. Herra Codd on itse määritellyt tarkat ja tiukat kriteerit, jotka tuotteen on täytettävä ennen kuin se saa oikeuden nimitykseen "relaatiomallin".

Sovelluskehittimille ei ole standardia

Relaatiotietokantatuotteet sisältävät yleensä myös sovelluskehittimen. Toisaalta on saatavissa avoimempia sovelluskehittäjiä, joissa on liitanta relaatiotietokantaan, tai jopa useampiin – varsinkin SQL-pohjaisiin – mutta myös muihin tiedostoratkaisuihin.

Sovelluskehittimien tilanne on relaatiotietokantoja sekavampi. Kun relaatiotietokanta-ajattelun peruspalkkina on selkeä teoria, sovelluskehittimille ei löydy vastaavaa taustatukea.

Termi sovelluskehitin tarkoittaa yleensä tuotetta, joka valikko-ohjatusti tuottaa ohjelmia. Kehittäjiä on pulpanut markkinoille laumoittain ja ne eroavat toisistaan suuresti.

Olemme lukeneet menestystarinoita sovelluskehittimistä. Toisaalta on lyöty päätä seinäänkin: moni on luullut kehittäjän ratkaisevan kerralla kaikki ongelmat. Kun näin ei ole käynytkaan, on luovutettu.

Uusia mahdollisuuksia

Erittäin mielenkiintoinen neljännen sukupolven välineisiin liittyvä piirre on siirrettävyys. Jotkut tuotteet toimivat monessa laite- ja varusohjelmistoympäristössä. Sovelluksen voi rakentaa mikrolla ja siirtää sen sitten vaikka jollekin minikoneelle tuotantoon – tai päinvastoin.

Toisaalta mikrojen erinomaisten käyttöliittymäominaisuuksien (ikku-

HENKILÖ

HENKUNN	ETUNIMI	SUKUNIMI	OSNRO	PUHELIN
2404	KARI	VIRTANEN	10	523676
2407	PEKKA	LAHTINEN	20	2987077
1978	JUKKA	JOKINEN	10	8976678
5682	MATTI	RANTANEN	30	
2210	VILLE	SAARINEN	20	289473

OSASTO

OSNRO	OSNIMI
10	MARKKINOINTI
20	TALOUS
30	TUOTANTO

Relaatiokannan perustaminen on hyvin nopeaa. Uusien taulukoiden lisääminen ja vanhojen muuttaminen on helppoa. Muutosten helpous tekee vaikutuksen jäykkiin tietokantaratkaisuihin tottuneeseen ohjelmoijaan.

nointi, ohjausrasia, hyppyvalikot...) hyödyntäminen vaatii aivan omat erikoiskehittimensä, joita odotetaan kiinnostuneina. Joka tapauksessa: on jo aika suhtautua vakavasti mikroiin myös sovelluskehitysmielessä.

Seinätekniikalla alkuun

Välineitä tärkeämpää on, kuitenkin suunnitella sovellus oikein. Kaivinkoneesta ei ole hyötyä, jos oja tuli väärään paikkaan! Suunnittelun paras väline on seinätekniikkaan perustuva käyttäjäkeskeinen systeemityö. Seiniltä on kokonaisuusien selvittyä hyvä lähtöä tekemään malleja sovelluskehittimellä ja rakentamaan relaatiotietokantaa.

Neljännen sukupolven välineet mahdollistavat iteroivan, pala palalta etenemisen. Protoilumahdollisuudet luovat aivan uusia ulottuvuuksia kommunikointiin käyttäjien kanssa.

Työtä siirretään tietokoneelle

Tuottavuuteen vaikuttavat tietysti monet muutkin seikat kuin tehokkaat työkalut. Silti yrityksillä ei ole enää varaa

jättäytyä vanhanaikaisten välineiden varaan.

Ohjelmistotalot ovat viime aikoina yhä yleisemmin alkaneet käyttää sovelluskehittäjiä ohjelmistopakettien tekoon. Onhan tuottavuus ja helppo räätälöitävyys niiden elinehto.

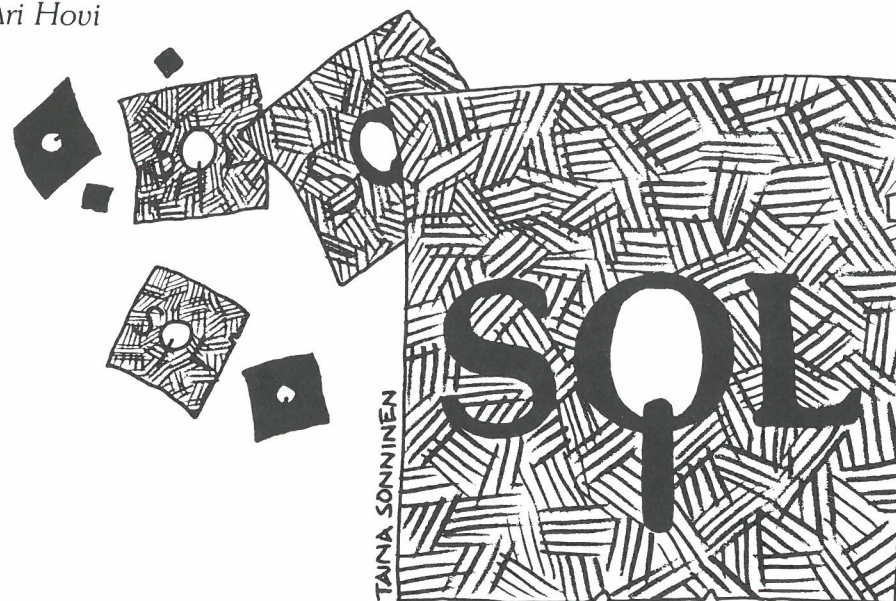
Yhä enemmän tulee päästä siihen, että atk-ammattilaisten työ keskittyy itse sovellusalueen ongelmien ratkaisuun, ei teknisten yksityiskohtien kanssa puuhaamiseen. Neljännen sukupolven välineiden läpitunkevana perusajatuksena on siirtää mahdollisimman paljon ohjelmoijan työstä järjestelmän hoidettavaksi. Sovellukset kehitetään enemmän tietokoneavusteisesti määrittelemällä, jolloin perinteinen ohjelmointi vähenee.

Kokemukset eivät ole olleet yksinomaan myönteisiä. Mutta atk-ammattilaisen on syytä tutustua uusiin välineisiin ja oppia niiden tarjoamista uusista mahdollisuuksista, eikä niinkään aina miettiä niiden rajoituksia.

Uusien välineiden sopiva käyttöön-ottohetki riippuu tietysti monesta tekijästä. Ennenkin tai myöhemmin on kuitenkin aika vaihtaa lapiot kaivinkoneisiin – ja varautua polttoainelaskuihin. □

"Relaatiotietokanta-ajattelu perustuu selkeään teoriaan, mutta sovelluskehittäjien tilanne on epämääräisempi. Standardista ei ole tietoaakaan. Markkinoilla olevat sovelluskehittimet, joita on Suomessa kymmeniä, ovat varsin eritasoisia", toteaa Ari Hovi, joka toimii konsulttina Tietojenkäsittelyneuvonta Oy:ssä.





on levinnyt laajalle

SQL-kielestä on tullut relaatiotietokantakielten standardi. Kymmenittäin SQL-kieltä tukevia tuotteita on saatavissa suuryrityksille, yrityksille ja mikroille.

Kielestä on kuitenkin kehittynyt eri murteita, joten eri toimittajien tuotteet eivät välttämättä ole yhteensopivia.

IBM kehitti laboratoriossaan vuosina 1975 – 1979 relaatiotietokantatuotetta, ns. R-prototyyppiä. Samassa yhteydessä julkaistiin SQL-kieli, "Structured Query Language". Siitä käytetään myös nimitystä "Sequel".

Nimi on harhaanjohtava, sillä SQL kattaa kyselyiden ohella myös tietokannan määrittelyn, käsittelyn ja valvonnan. SQL:ää käytetään nykyisin myös operatiivisissa järjestelmissä.

Useita tuotteita

Oraclen SQL-pohjainen relaatiotietokanta tuli markkinoille vuonna 1979. IBM toi markkinoille ensimmäisen SQL-tuotteensa, SQL/DS:n vuonna 1982. Merkittävämpi tuote, DB2 julkaistiin vuotta myöhemmin. Standardointielimet ovat myös noteeranneet SQL:n. ANSI-standardi ja Unix-ympäristön X/OPEN ovat tuoreita SQL-standardeja.

Julkistukset johtivat siihen, että SQL-kielestä tuli tosiasiallinen standardi relaatiotietokannoille. Nyt lähes

kaikki uudet tietokantajärjestelmät ovat SQL-pohjaisia ja relaatiomallin mukaisia.

Lisäksi monet perinteisten tietokantojen toimittajat ovat joko julkaisseet SQL-tuen tai ilmoittaneet aikovansa tehdä sellaisen. Myös mikroille on saatavissa useita SQL-tuotteita kuten Oracle, Ingres ja Quint/SQL. dBASEenkin on luvassa SQL-tuki.

SQL:n standardointia ja levinneisyyttä on pidettävä merkittävänä. Standardointihan ei tunnetusti ole helppoa.

Monia murteita

Monet toimittajat lupaavat, että heidän SQL-kielensä on yhteensopiva ANSI-standardin tai DB2:n kanssa.

Tosiasia on, että SQL-kielestä on levinnyt monia murteita.

IBM:n SQL oli ennen ANSI-standardin julkistusta esikuvana useimmille muille toimittajille. Luonnollisesti kiusaus laajentaa kieltä omilla pikku parannuksilla oli suuri. Niinpä eri toimittajien SQL-versiot eivät yleensä ole täysin yhteensopivia eivätkä siis siirrettäviä.

Saman toimittajan ratkaisuissa tilanne on toinen. Esimerkiksi Oracle-sovelluksen voi siirtää vaikkapa mikrolta DPS 6:lle, Stratukselle tai Vaxille.

Tietokannan käsittely yksinkertaiseksi

SQL on neljännen sukupolven kieli, joka käsittelee tietokannan tietueita joukkoina, ei tietue kerrallaan. Kielen perusfilosofian mukaan ohjelmoija tai käyttäjä voi keskittyä vain itse ongelman ratkaisemiseen. Osoittimiin, indekseihin ja muihin fyysisen tason asioihin ei tarvitse puuttua. Tietokannan viritys tehdään vasta jälkepäin, vaika

sovelluksen ollessa jo tuotannossa.

Tietokannassa "navigointi" on taakse jäänyttä. Moni on kokenut melkoisen elämyksen havaitessaan miten yksinkertaisia tietyt, ennen niin hankalat tietokantaoperaatiot ovat SQL-kielellä.

Tuottavuutta tietokantoihin

SQL:n avulla on nopeutettu merkittävästi sovelluskehitystä ja helpotettu sovellusten ylläpitoa. Tietokannat voidaan rakentaa joustavammin pala palalta kasvattaen.

Kielen oppii melko nopeasti. Vuorovaikutteisen luonteen ansiosta myös osa loppukäyttäjistä oppii SQL:n. Toimittajat ovatkin rakentaneet SQL:n päälle helppokäyttöisempiä käyttöliittymiä.

Luonnollisesti suuri osa SQL:ään kohdistuvasta ylistyksestä kuuluu taustalla olevalle relaatiomallille sinänsä.

Puutteita ja päällekkäisyyttä

SQL:ää on myös kritisoitu. Etenkin muutamat tärkeät relaatiomalliin kuuluvat piirteet puuttuvat. Kielessä on turhaa päällekkäisyyttä: saman kyselyn voi hoitaa monella eri käskyrakenteella.

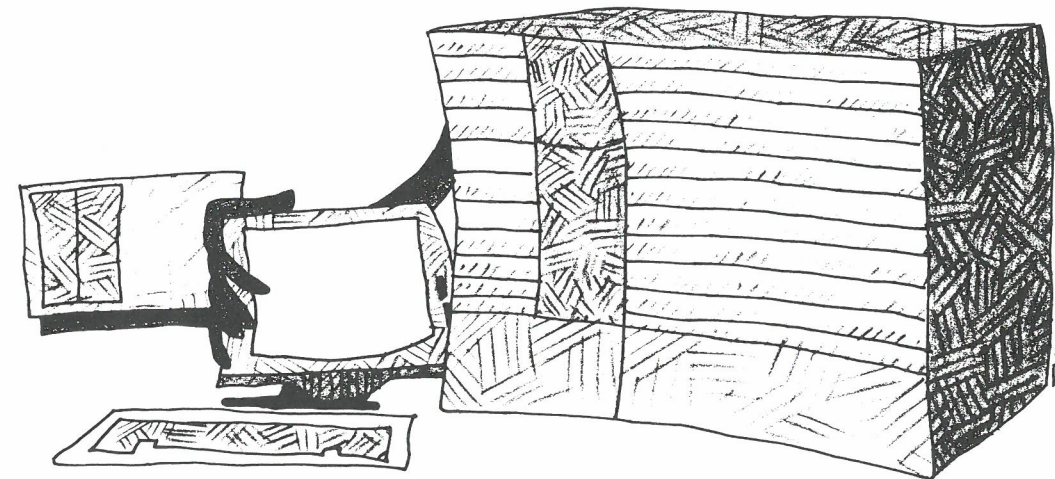
ANSI-standardi on eräänlainen kompromissi eri toimittajien versioiden välillä. Se on jopa eräiltä osin mennyt taaksepäin alkuperäiseen SQL:n verrattuna.

On muistettava, että SQL on "vain" tietokantakieli. Sovellusten tekoa varten tarvitaan lisäksi jokin muu kieli kuten Cobol ns. isäntäkieliliittymän kautta tai sitten sovelluskehitin.

Perinteiseen tietokantaohjelmointiin verrattuna edut ovat kuitenkin kiistatomat. Toivoa vain sopii, että kielen tietyt heikkoudet korjataan tulevisissa standardeissa. □

Oraclella tehdyt ohjelmat ja relaatiotietokannat ovat siirrettävissä sellaisinaan mikroista suuryrityksiin ja päinvastoin. Kun sama sovellusohjelma on käytettävissä useissa eri laitteissa, voidaan ohjelmointikustannuksissa säästää suuret summat.

Oracle-relaatiotietokantaohjelman uudessa versiossa on useita parannuksia. Mm. tiedonhallinta on tehostunut.



Oracle-tietokannat koneesta koneeseen

Keskitettyä tiedonhallintaa helpottaa se, että laitteilla on yhteinen tietokantarakenne. Eräs ratkaisu yhteiseksi tietokantarakenteeksi on Oracle.

Oracle-sovelluksen ja -tietokannan siirrettävyys edellyttää, että Oracle on saatavissa kohdelaitteistolle. Oraclea toimitetaan mm. MikroMikkoon, DPS 6:een ja Stratukseen. Lisäksi kohdelaitteiston Oraclen on oltava versioltaan sama tai uudempi kuin laitteiston, josta sovellus siirretään.

Kun ohjelmiston rakentaja on tehnyt Oraclella hyvän ohjelmiston, pystytään ohjelmoinnin kustannuksia katamaan toimittamalla samaa ohjelmistoa eri laiteympäristöihin.

Kätevästi muutoksia

Oraclen relaatiotietokantarakenne on joustava. Muutosten teko on yksinkertaista ympäristöissä, joissa tietokannan rakenne ja tietokenttien pituus ja tyyppi muuttuvat usein.

Tietokannan muutokset eivät aiheuta muutospaineita ohjelmistoon. Sovellusten käyttöä ei tarvitse keskeyttää muutosten aikana ja muutokset astuvat välittömästi voimaan.

Kaksi versiota

Oraclesta on saatavissa pääasiassa kahta versiota. Vanhempi, versio 4, on työkaluiltaan ja tietoliikennepiirteiltään kehittymättömämpi kuin uudempi versio 5. Sen sijaan tietokannan hallinta SQL-kielellä on lähes samanlaista molemmissa versioissa.

Uudempi versio on tuonut mukanaan parannuksia tiedonhallintajärjestelmään, mm. helpotusta ja tehoa monimutkaisiin kyselyihin ja tiedon masakäsittelyyn.

Versio 5 tukee myös hajautettua tiedonkäsittelyä. Hajautetussa ympäristössä voivat tiedonhallintajärjestelmä ja varsinainen sovellus sijaita eri tietokoneissa, eikä käyttäjien tai sovellusten tarvitse tietää, missä tietokannassa, laitteistossa tai millaisten tietoliikenneyhteyksien päässä tiedot sijaitsevat.

Välineitä atk-ammattilaiselle

Oracle on järeä atk-ammattilaisen väline, jonka opettelu vie aikansa. Sovelluskehitykseen tarvitaan lisäksi muita ohjelmointikieliä.

Oraclen työkaluissa on otettu huo-

mioon sovelluskehityksen tarpeet. Esimerkiksi näytönmaalaus- ja raportointityökalut on yhdistetty tietokannan hallintajärjestelmään. Lisäksi työkaluilla saadaan helposti aikaan käyttäjäystävällinen liittymä järjestelmään.

Uudessa versiossa 5.1 on SQL*Forms, SQL*Plus sekä SQL*Report -työkaluihin tehty uusia piirteitä ja niiden suorituskykyä parannettu.

SQL*Calc on Lotus 1-2-3 -tyyppinen taulukkolaskentaohjelma, jossa on lisäpiirteenä SQL-kieli. Ohjelmalla on mahdollista hakea tietoja tietokannasta taulukkopohjalle.

Lisäksi atk-ammattilaiselle on tehty apuvälineitä helpottamaan tietokannan käsittelyä sovellusten ulkopuolella. Easy*SQL on valikko-ohjattu apuväline, jolla voidaan ohjata ja siirtää tietokannan tietoja ja tehdä niistä kyselyjä, raportteja ja grafiikkatulosteita. □

Lisätietoja Oraclesta saa Heikki Paaterolta tai Antti-Heikki Suoniselta, puh. (90) 5671.

Suomen ensimmäinen tietämystekniikan väitös

Tietämykselle yleinen esittämistapa

Asiantuntijajärjestelmissä on tähän saakka jäljitelty asiantuntijoiden ongelmanratkaisukykyä syöttämällä tietokoneelle pääasiassa "jos-niin"-sääntömuotoisia tietämyksen paloja, joita kone on yhdistellyt päätelmiksi.

Tuoreessa väitöskirjassaan Jukka Bergqvist arvostelee tietämystekniikan käytäntöä, joka keskittyy sääntöesitysmuodon ja oliokeskeisen ohjelmoinnin yhdistämiseen. Näin syntyneissä sekajärjestelmissä ratkaistavan ongelman malli ja tavat esittää tietämystä ovat sekoittuneet keskenään.

Hän on kehittänyt tietämykselle kokonaan uuden, yleisen esittämistavan, tietämystyyppin.

Bergqvistin väitöskirja "A Type-Oriented Approach to Knowledge-Based Systems", "Tyypit tietämysteknisissä järjestelmissä", on Suomen ensimmäinen puhtaasti tietämystekniikkaa käsittelevä väitös.

Tietämystekniikan aseman tähänastisessa tutkimuksessa Bergqvist rinnastaa tilastotieteeseen. Tietämystekniikkaa on monessa työssä käytetty tilastotieteen tavoin apuvälineenä jotakin soveltavaa ongelma-aluetta tutkittaessa.

Tekoälyn Bergqvist määrittelee joukoksi tutkimusalueita: kielen ymmärtäminen ja tuottaminen, kuvantunnistus ja -käsittely, puheentunnistus sekä ihmisen ongelmanratkaisukyvyn jäljittely.

Tekoälyn sovelluksissa käytetään perinteisestä ohjelmoinnista poikkeavaa tietojenkäsittelytekniikkaa – tietämystekniikkaa. Bergqvistin määrittelemänä "tietämystekniikassa pyritään opettamaan tietokoneelle asioita vähä vähältä sen sijaan, että kirjoitettaisiin yksityiskohtaisia, perinpohjaisia ratkaisualgoritmeja".

Tämä tapahtuu syöttämällä koneelle tietämyksen palasia, jotka se kykenee liittämään toisiinsa erilaisilla yhdistelymekanismeilla. Tietämyksen palasia ovat tavallisimmin "jos-niin"-tyyppiset säännöt, mutta myös olio-ohjelmointiympäristöjen oliot voidaan ymmärtää tietämyksen palasiksi.

Mallintaminen ja esittäminen sekaisin

"Ongelman mallintaminen ja tietämyksen esittäminen ovat nykyjärjestelmissä menneet sekaisin. Reaalimaailman käsitteitä kuten asiakkaita, tilauksia ja yrityksen yksiköjä on mallinnettu olioina, joiden attribuutein on kuvattu paitsi itse käsitteiden ominaisuuksia, myös niiden ominaisuuksien ominaisuuksia."

"Esimerkiksi tuloslaskelmataulukon voidaan ajatella olevan yritysyksikön ominaisuus, osa reaalimaailman mallia, kun taas taulukon ulottuvuudet ja ulkoasu ovat taulukon itsensä ominai-

suuksia, tapa esittää tietämystä. Lisäksi tietämyksen "jos-niin"-sääntömuoto on kytkeytynyt melko kömpelösti oliomaailmaan", Bergqvist selostaa.

Yksi esitystapa

Bergqvist esittää tietämystä tyypein. Kaikki esittämistavat voidaan johtaa yleisestä tietämystyyppistä. Bergqvistin teoriassa omantyyppisiään ovat sekä tietämyksen palaset että niiden yhdistelymekanismit. Näin esimerkkejä tyypeistä ovat mm. graafinen matriisi, taulukko, "jos-niin"-sääntö ja aritmeettinen sijoituslause.

Tyyppi koostuu joukosta ominaisuuksia. Lisäksi tyyppiyhdistelmiin liittyy tulkintoja, joiden avulla tyypeille annetaan merkitys. Jos esimerkiksi taulukkotyyppinen tuloslaskelma halutaan esittää graafisesti, lähetetään tälle tulostuspyyntötyyppinen viesti, ja tulkinta, joka piirtää kuvaruudulle kaksulotteisen taulukon, käynnistyy.

"Tietämyksen yhdistelymekanismit, ovat myös tyyppitettyjä tietämyksen palasia. Niinpä uusia yhdistelymekanismeja on mahdollista luoda yhtä helposti kuin niiden käsittelemää tietämystä."

Tuotteisuus vaatii uskallusta

Nokia Tutkimuskeskuksessa on käynnistynyt väitöskirjan pohjalta Lisp-kielisen tyyppikeskeisen ohjelmointiympäristön toteutuminen.

"Suomessa tehdään kansainvälisestikin katsoen tasokasta tietämystekniikan tutkimusta. Sen käytäntöön soveltaminen on vaikeaa ja siihen ei raha ja ehkeivät vielä kyvytkään tahdo riittää. Jos halutaan tuotteita kansainvälisille markkinoille, panostusten tulee olla tuntuva. Meillä Suomessa saisi olla paljon enemmän uskallusta tuotteistaa saatuja tuloksia", huomauttaa Bergqvist, joka työskentelee Nokian tekoälyjärjestelmien vientipäällikkönä. □

Matti Katajamäki

Koko suunnittelu on Skaalassa



Perinteiset CAD-järjestelmät avustavat lähinnä suunnitteluprojektin loppuvaiheen detaljisuunnittelussa. Esisuunnitteluun ei ole ollut tietokoneavusteista työkalua.

Nokia ja Tampella ovat kehittäneet nimenomaan esisuunnitteluun tietämyspohjaisen suunnittelujärjestelmän, Skaalan. Se on yhdistetty yrityksen muihin tietojärjestelmiin relaatiotietokantaa hyväksikäyttäen. Suunnittelun, tuotekehityksen ja myynnin tietojärjestelmät voidaan näin koota yhdeksi insinööri-tietojärjestelmäksi.

Suunnittelutiedon käsittelyn ongelma on ollut tiedon hallinta ja siirto vaiheesta toiseen.

Tiedon hallintaa vaikeuttaa se, että yrityksillä on yleensä käytössä useiden toimittajien laitteistoja ja ohjelmistoja, jolloin tiedon esitys ja tallennus on keskenään yhteensopimatonta.

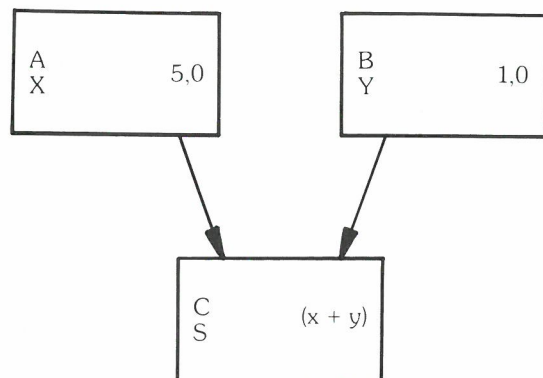
Suunnittelutietoa olisi kyettävä esittämään mahdollisimman yleisessä, suunnittelun näkökulmasta ja sovelluksesta riippumattomassa muodossa sekä massa- että virtuaalimuistissa.

Nokian ja Tampellan Kattilateollisuuden yhteistyönä toteutetussa "Tietämyspohjainen suunnittelujärjestelmä"-projektissa todettiin relaatiotietokanta parhaaksi vaihtoehdoksi massamuistissa tallennettavalle tiedolle ja kehyspohjainen esitys virtuaalimuistissa käsiteltävälle tiedolle.

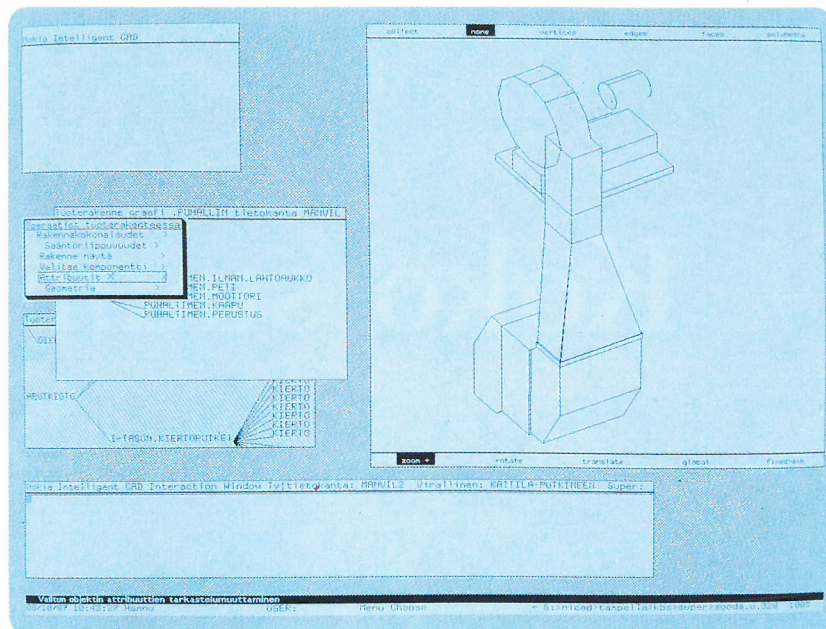
Tietämyspohjainen järjestelmä

Tietämyspohjaisessa suunnittelujärjestelmässä Skaalassa on käytetty työkaluina Symbolics Lisp-työasemaa, KEE (Knowledge Engineering Environment) kehyspohjaista asiantuntijajärjestelmien kehittäjä sekä S-Geometry-volyymimallinninta.

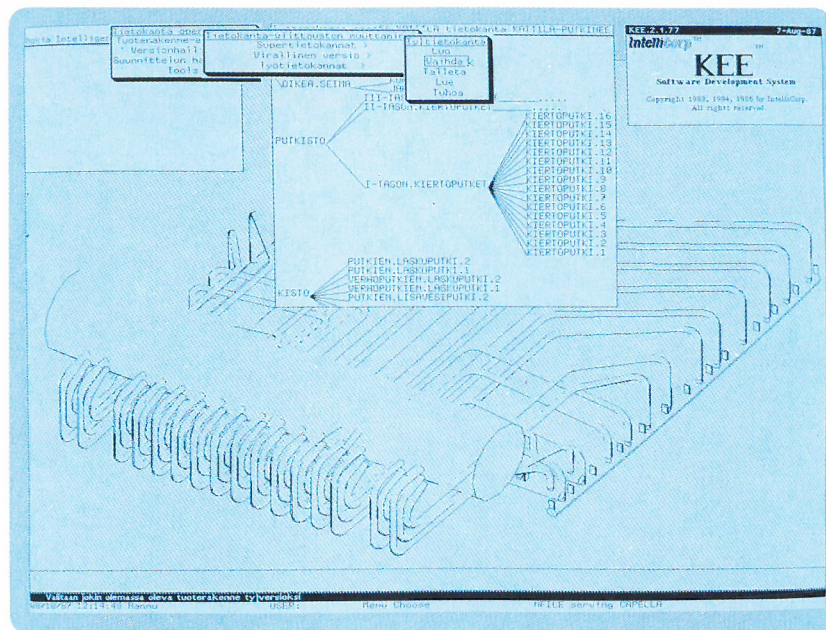
Kehyspohjainen toteutus mahdollistaa suunnittelutietämyksen yhtenäisen esityksen. Tietämyksellä tarkoitetaan suunnittelussa tuotettavan faktatiedon lisäksi niitä sääntöjä, menetelmiä ja standardeja, joiden avulla faktatietoa tuotetaan.



Kuva 1:
Esimerkki paikallisesta riippuvuudesta.



Kuva 2:
Teollisuuspuhaltimen tilavuusmalli. Vasemmalla on puhaltimen tuoterakenne, josta on valittu puhaltimen osan valikko.



Kuva 3:
Soodakattilan yläkiertoputkisto, jossa jokainen putki osaa soveltaa sääntöjä ja reitittää itsensä kytkentöjen ja ympäristön mukaan.

Tietämyksen esityksen perusyksiköitä KEEssä ovat kehykset (objektit). Suunnittelijan kannalta kehys vastaa yhtä osaa, joka voi olla esimerkiksi säiliö, putki tai palkki. Kehykset puolestaan sisältävät attribuutteja, jotka kuvaavat osien ominaisuuksia, kuten massaa, materiaalia, alikomponentteja ja ulottuvuuksia.

Suunnittelu- ja tuotetietämyssääntöt muodostavat osien ominaisuuksien välille riippuvuuksia, joista koostuu laaja riippuvuusverkko. Esimerkiksi kuvan 1 tapauksessa osan C ominaisuus s riippuu osan A ominaisuudesta x ja B ominaisuudesta y laskentäsäännöllä $s = (x+y)$.

Riippuvuusverkon avulla tietämyspohjainen suunnittelujärjestelmä kykenee erittäin monimutkaisiin laskenta-

mitoitus- ja valintatehtäviin. Myös suunnittelun automaattinen virheettömyys perustuu riippuvuusverkkoon.

Järjestelmä pystyy raportoimaan häiriöistä, jotka aiheutuvat suunnittelu-sääntöjen rikkomisesta, raja-arvojen ylityksistä tai geometrisista törmäyksistä.

Käyttöliittymänä tuoterakenne

Järjestelmän toiminnan yhden kulmakiven muodostaa tuoterakennetieto, joka kuvaa suunniteltavan kohteen ja kautumisen puumaiseksi osarakenteeksi. Tuoterakenne on myös oleellinen osa käyttöliittymää. Kuvassa 2 on teollisuuspuhaltimen tilavuusmalli ja tuoterakennepuu. Suunnittelija saa kunkin osan informaation tarkasteltavakseen näytölle osoittamalla ohjausrasialla haluamaansa osaa.

Kuvassa 3 on soodakattilan yläkiertoputkisto. Jokainen putki osaa sääntöjensä ja riippuvuuksiensa perusteella reitittää itsensä kolmiulotteisessa avaruudessa kammion putkiyhteeltä lieriön putkiyhteelle törmäyksittä.

Kaikki kokoonpanoon vaikuttavat ominaisuudet – putkien ja putkiyhenteiden lukumäärä ja paikat kullakin kammioilla, kammioiden sijainnit jne. – ovat muunneltavissa.

Relaatiotietokanta on linkki muihin tietojärjestelmiin

Relaatiotietokanta valittiin tietämyspohjaisen suunnittelujärjestelmän tallennusvälineeksi, koska relaatiotietokannat ovat joustavia, rakenteeltaan yksinkertaisia ja eri valmistajien tietokannat ymmärtävät standardiksi muodostunutta SQL-kyselykieltä.

Relaatiotietokantaa käytetään kaikkien suunnittelun tietojärjestelmien yhteisenä tiedontallennuspaikkana, jonka kautta eri järjestelmät jakavat keskenään projektitietoa.

Koska relaatiotietokannat eivät itse kykene valvomaan niihin tallennetun tiedon järjestyttävyyttä ja yhtäpitävyyttä, on valvonta annettu tietämyskannalle, jonka kautta kaikki tietokannan muutokset tapahtuvat.

Relaatiotietokannan ja tietämyskannan välinen yhteys on luonnollinen myös syntaksiltaan. Jokaista osaluok-

CADistä insinööritietojärjestelmään

Nokia on kehittänyt ja myynyt tietokoneavusteisia suunnittelujärjestelmiä vuodesta 1982. Painopiste on siirtynyt CAD-järjestelmistä yhä laajempiin kokonaisuuksiin.

CAD-järjestelmät tarjoavat graafiset päätteet ja geometriset ohjelmistopakettit avuksi detaljisuunnitteluun. Näin on saatu tehostettua tätä työvoimavaltaisinta suunnitteluvaihetta.

Toisessa vaiheessa CAD-järjestelmät yhdistettiin tietokoneavusteiseen valmistukseen (CAM). Näissä järjestelmissä esimerkiksi koneen työstöradat pystytään määrittelemään päätteellä.

Kolmannessa vaiheessa CAD ja CAM on liitetty yrityksen jo olemassa oleviin järjestelmiin, kuten materiaalihallintoon ja tuotannonohjaukseen kokonaisvaltaiseksi tietokoneavusteiseksi suunnittelujärjestelmäksi, CAE:ksi.

CAE-järjestelmällä pystytään optimoimaan myynti, suunnittelu, tuotekehitys, raaka-ainevarasto, valmistusaikataulut ja valmistevarasto, sillä järjestelmä tietää tarkasti kunkin osa-alueen tilanteen ja pystyy siirtämään tietoa.

Aikaisemmat järjestelmät eivät ole ratkaisseet esisuunnittelun ongelmia. Niihin Nokialla on kehitetty tietämyspohjainen suunnittelujärjestelmä, Skaala.

Tietämyspohjainen suunnittelujärjestelmä perustuu kehyspohjaiseen tiedon esitykseen sekä relaatiotietokantaan. Tietämyspohjainen järjestelmä kytkee yrityksen suunnittelun tietojärjestelmät, kuten teknisen laskennan järjestelmän ja perinteisen CAD-järjestelmän sekä mahdolliset projektihallinnan ja materiaalihallinnon järjestelmät yhdeksi kokonaisuudeksi, insinööritietojärjestelmäksi. Sen avulla tehostetaan suunnittelua ja saavutetaan aika- ja kustannussäästöjä.

kaa vastaa yksi relaatio tai näkemys, jonka sarakkeita ovat ne osan ominaisuudet, joita tietokantaan halutaan tallentaa. Relaatiotaulukoiden rivit muodostuvat eri suunnitteluversioiden osien ominaisuuksista.

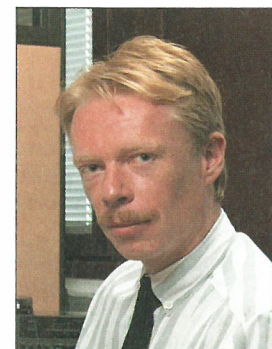
Relaatiotietokannasta saadaan suoraan SQL-raporttikehittimellä projekti- ja versiokohtaista tietoa.

Vaativaan suunnitteluun

Tietämyspohjaisen suunnittelujärjestelmän ominta sovellusaluetta on vaativien mekaanisten tuotteiden ja laitteiden suunnittelu. Sitä on käytetty huhtikuusta lähtien Tampellassa soodakattiloiden suunnitteluun.

Skaalaan voi tutustua 7. – 9. lokakuuta Tampereen Hitec-messuilla.

Dipl.ins. Matti Kattajamäki työskentelee Informaatiojärjestelmien Teollisuus ja suunnittelu-yksikössä kehityspäällikkönä erikoisalanaan tietämyspohjaiset ja CAD-järjestelmät.



Vikasietoisuus ohjelmilla tai laitteistoilla

Vikasietoiset tietokoneet pystyvät käsittelemään tietoa keskeytyksettä, vaikka yksi tai useampi laitteen osa vioittuisikin.

Vikasietoisuus on yleensä saatu aikaan joko ohjelmallisesti tai laitteiston kahdennuksella.

Ensimmäiset vikasietoiset tietokoneet tulivat markkinoille 1970-luvun loppupuolella. Viime vuosina on ilmaantunut useita eri laitevalmistajia.

Johtavat toimittajat ovat Stratus Computer ja Tandem Computer. Tandem sai markkinoille ensimmäiset laitteistonsa 1970-luvun lopussa, Stratus 1980-luvun alussa. Stratus edustaa Suomessa Nokia Informaatiojärjestelmät.

IBM astui vikasietoisien markkinoille solmiessaan sopimuksen Stratuksen kanssa. IBM myy Stratuksen laitteita ja käyttöjärjestelmää omalla System/88-tuotenimellä.

Vikasietoisien laitteiden tarve yrityksissä kasvaa koko ajan, koska toimintoja automatisoidaan yhä enemmän. Ja aika, jonka tietokone seisoo, maksaa.

Jatkuva, oikeata tietoa

Yleensä vikasietoiset laitteistot koostuvat yhdestä tai useammasta tietokoneesta sekä erikoisohjelmistoista.

Toimintaperiaatteena on, että tietojen käsittely ei keskeydy millään lailla, vaikka jokin tärkeä komponentti – suoritin, keskusmuisti, jokin tietoliikenne- ja levyohjaimista tai levy-yksiköistä – hajoaisikin.

Toinen koneiden tärkeä ominaisuus on, että järjestelmässä liikkuvat tiedot ovat koko ajan oikeita.

Tandemiin ohjelmoitava tarkistuspisteet

Tandem perustuu ohjelmalliseen vikasietoisuuteen.

Tandem-tietokonejärjestelmä rakentuu kahdesta tietokonelaitteistosta, jotka on kytketty toisiinsa nopealla väylällä.

Ohjelmat on pilkottu osiin, joista jokaisesta on olemassa varmistusosa. Tietokoneissa ajetaan ristikkäin sovellusohjelmiston varsinaisia osia ja varmistusosia. Mikäli jomman kumman tietokoneen tärkeä komponentti hajoaa tai ohjelmistossa on virhe, sovelluksen suoritus jatkuu ainoastaan toisessa koneessa. Hajonnut kone poiste-

taan toiminnasta.

Tandemissa sovellusohjelmiin on erikseen ohjelmoitava tarkistuspisteet, joiden mukaan varmistusosalle ilmoitetaan, missä vaiheessa sovellusta ollaan menossa.

Stratus erottaa laite- ja ohjelmaviat

Stratuksen vikasietoisuus perustuu pelkästään laitteistoon. Kaikkia toiminnalle tärkeitä komponentteja on kaksi kappaletta yhdessä tietokoneessa. Sen lisäksi jokainen komponentti on itsessään kahdennettu, jolloin toiminta tapahtuu itse asiassa neljässä paikassa yhtä aikaa.

Komponenteissa on vertailulogiikka, jonka avulla komponentin vioittuminen havaitaan välittömästi. Komponentti todetaan vialliseksi, mikäli itsessään kahdennetun komponentin osat antavat eri tuloksen. Komponentin vioittuessa toimintaa jatkaa ehjä pari, joka on toiminut koko ajan tahdistetusti rikkoutuneen parin rinnalla. Tämän ratkaisun ansiosta voidaan laitteisto- ja ohjelmistoviat erottaa toisistaan.

Vikasietoisia tietokoneita on käytetty pääasiassa pankeissa ja rahalaitoksissa suurten tapahtumamäärien ajantasaiseen käsittelyyn. Näissä ympäristössä katkottomuus ja tiedon oikeellisuus ovat ensisijaisia vaatimuksia. Viime vuosina teollisuus on kasvanut markkina-alueena.

Eroja tehossa ja luotettavuudessa

Tietokoneanalyttikko David Moschella markkinatutkimuslaitos IDC:stä (International Data Corporation) näkee Tandemin ja Stratuksen eron seuraavasti:

"Tandemin järjestelmissä laitteistojen käyttö on tehokkaampaa, koska jokaisen laitteiston osa suorittaa eri tehtävää. Mutta Stratukset ovat vikasietoisempia, koska ne perustuvat moninkertaiseen laitteistoon ja edistyneempään tekniikkaan."



Uusimman Stratuksen XA 2000:n tekniset tiedot:

- 32-bittinen keskusyksikkö, kellotaajuus 16 megahertsia
- mikrosuoritin MC 68020,
- aritmetiikkasuoritin MC 68881,
- suoritinmuistia 64 kilotavua
- yhden prosessin osoitettavuus 128 megatavua
- käyttöjärjestelmä VOS 6.0
- laitteisto ja ohjelmisto laajennettavissa
- uusien ja vanhojen koneiden ohjelmistot ja oheislaitteet yhteensopivia
- saatavissa mm. toimistojärjestelmä SOS (Stratus Office Solution)
- ohjelmistokehitykseen mm. Basic, C-kieli, Cobol, Fortran, Pascal, PL/1, Oracle, Nopsa
- Stratuksia on Suomessa mm. Veikkauksella, Helsingin Puhelinyhdistyksellä, Tampereen Puhelinosuuskunnalla, Helsingin aluehälytyskeskuksella, Helsingin Arvopaperipörssillä, Sanomapainolla, Turkistuottajilla ja Nokialla

Ins. Heikki Paatero on Stratuksen tuotepäällikkö. Hän antaa myös lisätietoja Stratuksesta numerossa (90) 5671.

Laitteiston vikasietoisuus yksinkertaistaa ohjelmointia

Ohjelmallinen vikasietoisuus asettaa ohjelmoinnille erityisvaatimuksia ja hankaloittaa ohjelmointia. Mutta silloin kun vikasietoisuus on järjestetty laitteiston kahdennuksella, vikasietoisuutta ei tarvitse ottaa ohjelmoinnissa huomioon. Vikasietoinen järjestelmä voi siten olla jopa perinteistä järjestelmää miellyttävämpi ohjelmointiympäristö.

Ensimmäiset, ohjelmalliseen vikasietoisuuteen perustuvat järjestelmät olivat tavallisia tietojenkäsittelyjärjestelmiä luotettavampia, mutta huomattavasti kalliimpia. Niissä kaksi tietokone-laitteistoa yhdistetään kalliiden, sovel-luskohtaisten ohjelmavipujen avulla. Jos varsinainen järjestelmä vioittuu, sen tehtävät siirtyvät varmistuslaitteis-ton huoleksi.

Ohjelmallinen vikasietoisuus on on-gelmallista, sillä varmistuslaitteistossa on oltava sovellusohjelman varmistus-versio. Varmistusversiota on päivitet-tävä koko ajan, mikä vaatii jatkuvaa tie-tooliikennettä varsinaisen ja varmistus-laitteiston välillä, sekä ns. tarkistuspis-teiden lisäämistä ohjelmakoodiin.

Monimutkaisinta on juuri tarkistus-pisteiden ohjelmointi. Lisäksi vikasie-toinen ohjelmakoodi aiheuttaa laitteis-tolle päällekkäiskuormitusta.

Vikasietoisuutta ei tarvitse huomioida

Stratuksen vikasietoisuus perustuu kahdennettuun laitteistoon, jolloin so-vellusohjelmoijan ei tarvitse huomioi-da vikasietoisuutta omassa työssään.

Stratus-arkkitehtuuri pohjautuu kahteen perusratkaisuun: jokaisen järjestelmäkortin itsetarkistus-logiikkaan ja kahdentamiseen. Nauhaohjainta lukuunottamatta kaikki Stratus-järjestelmän keskei-set osat on kahdennettu.

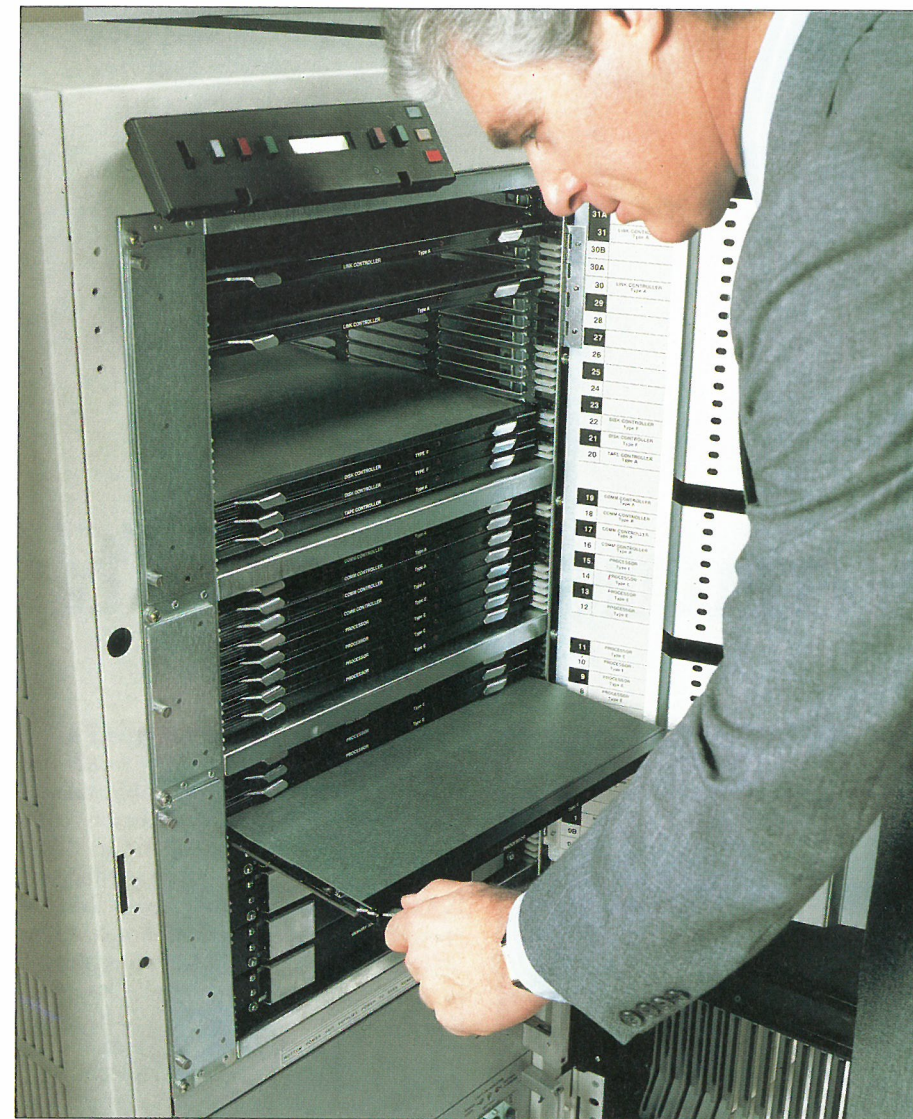
Stratuksen arkkitehtuuri koostuu kahdesta erillisestä, mutta toisiinsa lii-tyvästä ratkaisusta: kaikki keskeiset jär-jestelmäkortit on kahdennettu ja ne si-sältävät kortin toimintakuntaa tarkkai-levan vertailulogiikan.

Tietoa käsitellään kussakin kortissa kahdella rinnakkaisella puoliskolla. Kä-sittelyn tuloksia verrataan 16 miljoonaa kertaa sekunnissa ja jos tulokset poikkeavat toisistaan, kortti katsotaan vialliseksi ja sen toiminta pysähtyy.

Lisäksi kaikkia järjestelmäkortteja on kaksi kappaletta ja ne suorittavat täsmälleen samat toiminnot. Jos toi-nen korteista havaitsee itsensä vialli-seksi, se lopettaa toimintansa. Toinen kortti toimii edelleen, joten järjestel-mässä ei tapahdu käyttäjälle näkyviä muutoksia tai toimintakatoja. Järjes-telmä määrittelee itse kokoonpanonsa uudelleen ilman, että käyttäjän tarvit-see puuttua sen toimintaan.

Stratus ymmärtää yleisimpiä ohjelmointikieliä

Stratus-sovellusten ohjelmointiin voi-daan käyttää suosituimpia korkean ta-son ohjelmointikieliä kuten Basicia,



Cobolia, Fortrania, PL/1:tä, Pascalia ja C-kieltä.

Kaikki kielet käännetään Stratuksel-la samanlaista prosessia noudattaen. Ensiksi kääntäjä käy läpi lähdekoodin ohjelmointikielikohtaisen rakenteen. Tuloksena on yhteinen välikieli (Com-mon Intermediate Code). Sen jälkeen ohjelman osat yhdistellään ja välikie-lestä muodostetaan ohjelmointikieles-tä riippumaton koodi, jonka keskusyk-sikkö kykenee suorittamaan.

Kaikki kääntäjät käyttävät samaa koodin optimoijaa ja kehittäjä, jolloin kääntäjät toimivat erittäin luotettavasti ja uusien kääntäjien lisääminen Stra-tus-järjestelmään on helppoa.

Käännösprosessin merkittävin etu on se, että ohjelmoija voi esimerkiksi kirjoittaa 25 rivin Cobol-ohjelman ym-märtämättä mitään Stratuksen vikasie-toisuustekniikasta ja silti ohjelma on sataprosenttisesti vikasietoinen.

Lisäksi ohjelmointikieliet ovat modu-larisesti yhteensopivia. Esimerkiksi Cobol-ohjelmasta voidaan kutsua mitä tahansa muunkielistä ohjelmaa.

Virtuaaliympäristö nopeuttaa sovelluskehitystä

Stratus-järjestelmän sydän on VOS-käyttöjärjestelmä (Virtual Operating System), joka on helppo sovellusten suunnittelu- ja ohjelmointiympäristö.

Nimensä mukaisesti VOS tukee vir-tuaalimuistia. Lisäksi se on tyypillinen monisuoritin- ja moniajoympäristö.

VOS-käyttöjärjestelmä mahdollistaa työskentelyn erityyppisillä päätteillä. Järjestelmäympäristö voi olla suurikin verkko, eikä ohjelmoijan tarvitse ottaa sitä huomioon. Kaikki nämä ominai-suudet helpottavat hänen työtään.

Tukea neljälle suorittimelle

VOS tukee jopa tehokkaimman Stra-tus XA 2000 -koneen mallin 140 nel-jää loogista suorittinta.

Useiden suorittimien yhtäaikaisten toiminta lisää merkittävästi järjestel-män tehokkuutta, sillä samanaikaisesti voi olla käynnissä mikä tahansa käyttö-järjestelmä-, ohjelmankäytys- ja ajan-tasaisten tapahtumankäsittelytehtävien

yhdistelmä.

Ohjelmoijan näkökulmasta moni-suoritinjärjestelmä toimii kuten yksi-suorittiminen laitteisto, sillä ohjelmoin-nissa ei monisuoritinympäristöä tarvit-se huomioida. Sovellusohjelmat käyt-tävät automaattisesti hyväkseen moni-suoritinjärjestelmän suoman rinnak-kaisajomahdollisuuden.

Useat päätteet kelpaavat

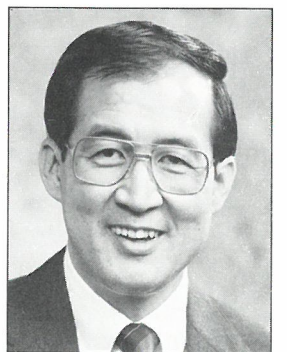
Stratuksen näytön- ja tapahtumankä-sittelyjärjestelmän (Forms and Trans-action Processing Facility, TPF) avulla monen käyttäjän sovellusten suunnit-telu ja ohjelmointi on yksinkertaista.

Ohjelmoijalla on käytössään Forms-nimiset näytönkäsittelyvälineet, joilla hän suunnittelee ja muokkaa näyttöjä ja tallentaa ne. VOS-käyttöjärjestel-män ansiosta näytön käsittely on sa-manlaista useilla päätetyypeillä kuten IBM 3270- ja ASCII-päätteillä sekä mikroilla. Ohjelmoijan ei siis tarvitse sovellusta suunniteltaessa ottaa kantaa siihen, mitä päätettä käyttää. Hän oh-jelmoi sovelluksen päätteeseen liitty-vät toiminnot aivan kuin sovellusta käytettäisiin vain yhdentyyppiseltä päätteeltä.

TPF sisältää erikoisominaisuuksia, jotka tukevat ajantasaista tapahtuman-käsittelyä. Se järjestää useiden päätteiden kanssa kommunikoivan sovelluk-sen yhteydet, valvoo jokaisen käyttä-jän toiminnon etenemistä ja huolehtii useiden tapahtumien rinnakkaisesta suorittamisesta.

Käyttäjälle näkymätön verkko

VOS-käyttöjärjestelmän ansiosta käyt-täjä näkee yhden ja yhdenmukaisen laitteiden ja ohjelmien kokonaisuuden, olipa kyseessä yhden suorittimen Stratus-järjestelmä tai laaja, useista sa-doista paikallisista tai hajautetuista tie-tokoneista koostuva verkko. Niinpä so-vellusten suunnittelu voidaan aloittaa pienellä laitekokooppaanolla, jota voi-daan kasvattaa sovellusta vaihtamatta, kun liiketoiminnan kasvu sitä edellyt-tää. □



Gary Okimoto on Stratus Comput-erin ohjelmistotuot-tannon apulaisjo-h-taja.

Stratus varmistaa HPY:n sanomanvälityksen

**Elisa on Helsingin Puhe-
linyhdistyksen sanoman-
välitysverkko, johon voi-
daan kytkeytyä yksittäi-
sellä päätteellä tai tieto-
konejärjestelmällä – mer-
kistä riippumatta.**

**Se perustuu Nokian toi-
mittamaan S.mail-ohjel-
mistoon ja vikasietoiseen
Stratus-laitteistoon. "Eli-
sa on 20-vuotisen urani
ensimmäinen atk-hanke,
jossa on vältetty atk-toi-
mitusten perisyynit. Ja
ansio kuuluu Stratuksel-
le", sanoo Lauri Hirvonen,
joka on koordinoanut Eli-
sa-projektia Nokian puo-
lella.**



**Nokian Lauri Hirvonen painottaa
sitä, että sanomanvälitysjärjestel-
mää käyttävän asiakkaan on voi-
tava luottaa sanomansa perille-
menoon.**

Elisa eli elektroninen integroitu sano-
manvälityspalvelu otettiin virallisesti
käyttöön viime vuoden maaliskuussa.

HPY:llä Elisasta vastaavaan ryh-
mään kuuluvan käyttöinsinööri **Ilkka
Veuron** mukaan päätepostikäyttäjiä oli
syyskuun puolivälissä noin 700 ja
oman tietokoneen tai tietokoneverkon
kautta liittyneitä järjestelmäasiakkaita
kymmenkunta. Tällä hetkellä käytössä
on Honeywell-, Vax-, IBM- ja luonnol-
lisesti Stratus-järjestelmäliitännät. Li-
säksi teleksiverkosta päästään Elisaa
ja päinvastoin.

Aina käytössä

Ilkka Veuro painottaa sitä, että Puhe-
linyhdistys valitsi ainoastaan ohjelmis-
totoimittajan, jonka vastuulle jätettiin
oikean laiteympäristön valitseminen.

Lauri Hirvosen mukaan Elisa-hank-
keen suunnittelu aloitettiin pohtimalla,
mikä on sanomanvälitysjärjestelmän
tärkein ominaisuus. "Vastaukseksi saa-
tiin luotettavuus. Järjestelmän on olt-
ava sellainen, että käyttäjä voi turvalli-
sin mielin lähettää sanomia, joilla on
hänelle taloudellista merkitystä, eikä
pelkkiä hyvän huomenen toivotuksia",
hän määrittelee.

Tätä taustaa vasten Stratus oli vah-
voilla laiteympäristöä valittaessa.
HPY:n Ilkka Veuro kertoo, että tietoliik-
kennepuolella ei juuri käytetä kahden-
tamattomia järjestelmiä. Tietoliikenne-
palvelu on luonteeltaan sellaista, että
sen on oltava asiakkaan käytössä ym-
päri vuorokauden.

Laitevikoja ei ole ollut

Elisan päivittäisestä toimintakunnosta
vastaavan työnjohtaja **Pertti Heikkisen**
mukaan Stratuksen vikasietoisuutta on
vaikea arvioida, kun todellista vertailu-
kohdetta ei ole.



**HPY:n käyttöinsinööri Ilkka Veu-
ron mukaan Elisaa ei voitu ajatella
rakennettavaksi kahden rinnakkai-
sen tietokoneen varaan. Ainoaksi
mahdollisuudeksi jäi keskeytymät-
tömän toiminnan mahdollistava
vikasietoinen laitteisto. Veuron mu-
kaan HPY kehittää Elisaa edel-
leen. Tulossa on mm. Mikro-Elisa.**

Absoluuttinen totuus on, että järjes-
telmän toiminta ei ole kertaakaan kes-
keytynyt laitevian vuoksi. Aivan ongel-
mitta ei Elisa ole toiminut, mutta Stra-
tuksen niskoille ei paria toimintakat-
koa voida vierittää.

Stratuksen VOS-käyttöjärjestelmän
kanssa ei HPY:llä ole ollut vaikeuksia.
Hyödyllisenä lisäpiirteenä on ollut
VOSin suoma mahdollisuus muuttaa
käyttöjärjestelmäkäskyjä. Näin nekin
käyttäjät, jotka ovat tottuneet työsken-
telemään jonkin muun käyttöjärjestel-
män kanssa, voivat helposti käyttää
VOSia muuttamalla sen käskyt halua-
mukseen.

Stratus antoi atk-synninpäästön

Nokian Lauri Hirvonen pitää Elisaa
hyvin poikkeuksellisenä atk-hankkee-
na. "Kaikkien näkemieni isojen atk-
projektien kolme perisyntiä ovat olleet
aikatauluista lipeäminen, määrittelyistä
tinkiminen ja ohjelmistovirheistä johtu-
va ylläpidon raskaus", hän mustamaa-
laa.

Hirvosen mukaan Elisa toimitettiin
käytännössä jopa hieman etuajassa ja
toiminnoiltaan laajempaan kuin ensi
vaiheen toimitukseen oli sovittu. Myös
ohjelmistovikoja on ollut tavallista vä-
hemmän.

Hirvonen suitsuttaa sumeilematta
Stratukselle ansiota tästä "atk-synnin-
päästöstä". Tavalliseen tietokoneeseen
verrattuna ohjelmisto on saatu nopeas-
ti kehitettyä Stratukseseen, koska ohjel-
moijien ei ole tarvinnut huomioida lai-
tevikojen mahdollisuutta ohjelmis-
saan. Myös Stratuksen VOS-käyttöjär-
jestelmän joustavuus on ollut suureksi
avuksi kehitys- ja testausvaiheissa. □



**Työnjohtaja Pertti Heikkisen mu-
kaan toiminta ei ole keskeytynyt
kertaakaan laitevian vuoksi.**

Uudet Ada-välineet tehostavat testausta

Ohjelmistojen testausta on tarve tehostaa – viehän se yleensä 30 – 50 prosenttia ohjelmistohankkeeseen käytettävää ajasta. Nokialla on kehitetty Ada-ohjelmien testaukseen kaksi työkalua, joilla on ollut kysyntää ulkomaillakin.

Testaukseen on olemassa – joskaan ei ammattikunnassa yleisesti tiedossa – tiettyjä menetelmiä /1/. Niitä tukevia työkaluja ei kuitenkaan ole juuri ollut saatavilla. Jäljitysohjelma, debugger, on ollut pitkään lähes ainoa testausväline.

MPS 10 -tietokoneella on kehitetty Ada-ohjelmien moduulitestaukseen TBGEN ja TCMON -työkalut. Näillä välineillä testaus tehostuu, koska se voidaan automatisoida hyvin pitkälle. Testausympäristöt kehitetään TBGENillä. Testausympäristöt suorittavat testit ja tarkistavat tulokset automaattisesti. Testauksen kattavuus selvitetään TCMONilla.

Mitä testaus on?

Yleensä testauksella tarkoitetaan työvaihetta, joka tehdään ohjelmoinnin jälkeen ennen ohjelmiston toimitusta. Varsinainen testaus on osatehtävä, jossa yritetään löytää mahdollisimman tarkoin testattavan ohjelmiston virheet. TBGEN (test bed generator for Ada packages) ja TCMON (test coverage monitor and program bottleneck finder) -työkalut on tarkoitettu juuri tähän tehtävään.

Kun virheet on havaittu, yritetään selvittää mistä ne johtuvat. Jäljitysohjelma on tähän mitä parhain työkalu.

Kun virheiden syytkin on saatu selville, niiden poistaminen vaatii tilanteesta riippuen uudelleenmäärittelyä, suunnittelua, muokkausta, käännöksiä ja yhdistelyä. Lopuksi aiempi testaus on toistettava, jotta nähdään, onnistuivatko korjaukset ja etteivät ne tuoneet mukanaan uusia virheitä.

Testausta pakkaus kerrallaan

Adalla ohjelmointi on erikseen käännettävien ohjelmayksiköiden ohjelmointia. Yleensä nämä yksiköt ovat pakkauksia (package). Suoritettava Ada-sovellusohjelma yhdistellään pääohjelmasta ja sen käyttämistä ohjelmayksiköistä.

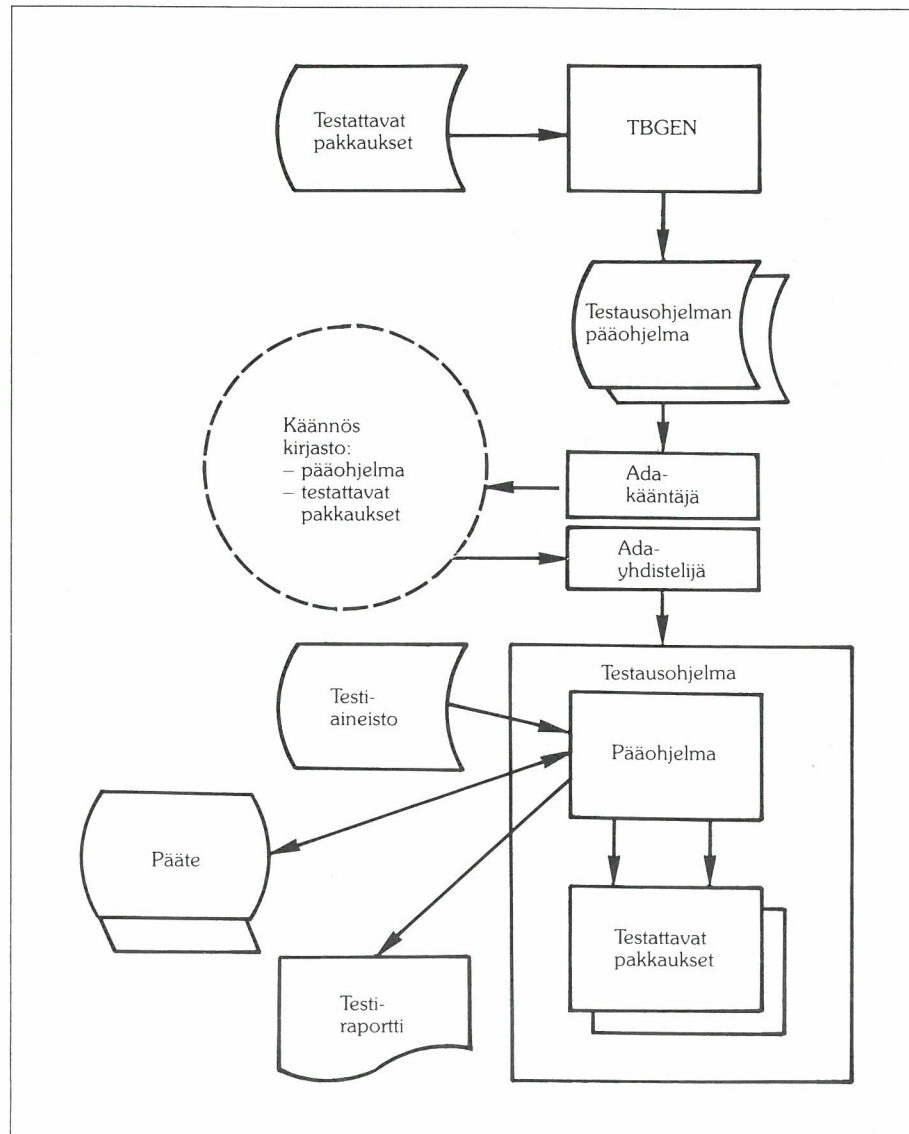
Yksittäiset pakkaukset on järkevintä testata erikseen ennen kuin ne yhdistellään lopulliseen sovellusohjelmaan.

TBGEN-järjestelmä on kehitetty autamaan käyttäjää tässä tehtävässä. TBGENin avulla voidaan kehittää helposti halutuille pakkauksille testausympäristö (testausohjelma, test bed, test

harness), jonka avustamana testaus tehdään. Kuva 1 esittää TBGEN-järjestelmän rakennetta.

TBGEN on sovelluskehittimen kaltaisen työkalu. Sen syöttötietona ovat testattavien pakkausten määrittelyt Ada-kielisessä muodossa. Niiden perusteella TBGEN kehittää joukon Ada-kielisiä ohjelmayksiköitä, jotka muodostavat testausohjelman pääohjelman (test driver). Pääohjelman koko vaihtelee testattavista pakkauksista riippuen. Yleensä se on muutama tuhat Ada-riviä.

Seuraavaksi pääohjelma käännetään ja yhdistellään testattavien pakkausten kanssa. Tuloksena on haluttu testausympäristö.



Kuva 1: TBGEN-järjestelmän rakenne

Testausympäristössä monia toimintoja

Testausohjelman pääohjelma keskustelee pääteen kanssa tietyllä yksinkertaisella Adan tapaisella komentokielellä. Seuraavassa kuvataan testausympäristön keskeisimpiä toimintoja.

Apumuuttujia voidaan varata testausvarten tarpeen mukaan. Ne toimivat samalla tavalla kuin Ada-ohjelmassakin esitellyt muuttujat. Ne esitellään myös käsitteellisesti samalla tavalla eli niille annetaan nimi ja tyyppi. Tyyppi voi olla joku Adan ennaltamääritellyistä tyypeistä tai sellainen käyttäjän tyyppi, jonka esittely on testattavissa pakkausissa.

BED> ALLOC \$INT : INTEGER
BED> ALLOC \$STR : STRING(1..6)
BED> ALLOC \$T1 : MYPACK.T
Apumuuttujalle asetetaan arvo ja se tulostetaan:

```
BED> SET $INT := 100  
BED> SET $STR := "qwerty"  
BED> SET $T1 := ('C', 5.9, (0,1))  
BED> DISPLAY $INT  
$INT = 100
```

Testattavien proseduurien ja funktioiden kutsut ilmaistaan samalla tavalla kuin Ada-kielessä. Todelliset parametrit voivat olla välittömiä vakioita tai edellä mainittuja apumuuttujia. Pääohjelma kutsuu aliohjelmaa annetuilla parametreilla. Kutsun jälkeen pääohjelma tulostaa paluuparametrien saamat arvot. Jos aliohjelman suorituksesta palataan poikkeuksella, pääohjelma tulostaa syntyneen poikkeuksen nimen. Näin käyttäjä näkee välittömästi, miten aliohjelma toimii kyseisillä parametrien arvoilla.

```
BED> MYPACK.PR4('3', $STR, $INT)  
$STR = "333333"  
$INT = 3  
BED> $INT := MYPACK.LEN("How long?")  
$INT = 8  
BED> $INT := MYPACK.LEN("")  
*** exception MYPACK.SIZE_ERROR
```

Testin oikeellisuus voidaan tarkistaa eli todellista ja odotettua tulosta voidaan vertailla. Tämä on tärkeä ominaisuus ja tätä käytetään, kun testitapaukset tulevat tiedostosta. Testausympäristössä on ASSERT-komento, jota käytetään seuraavan esimerkin tapaan:

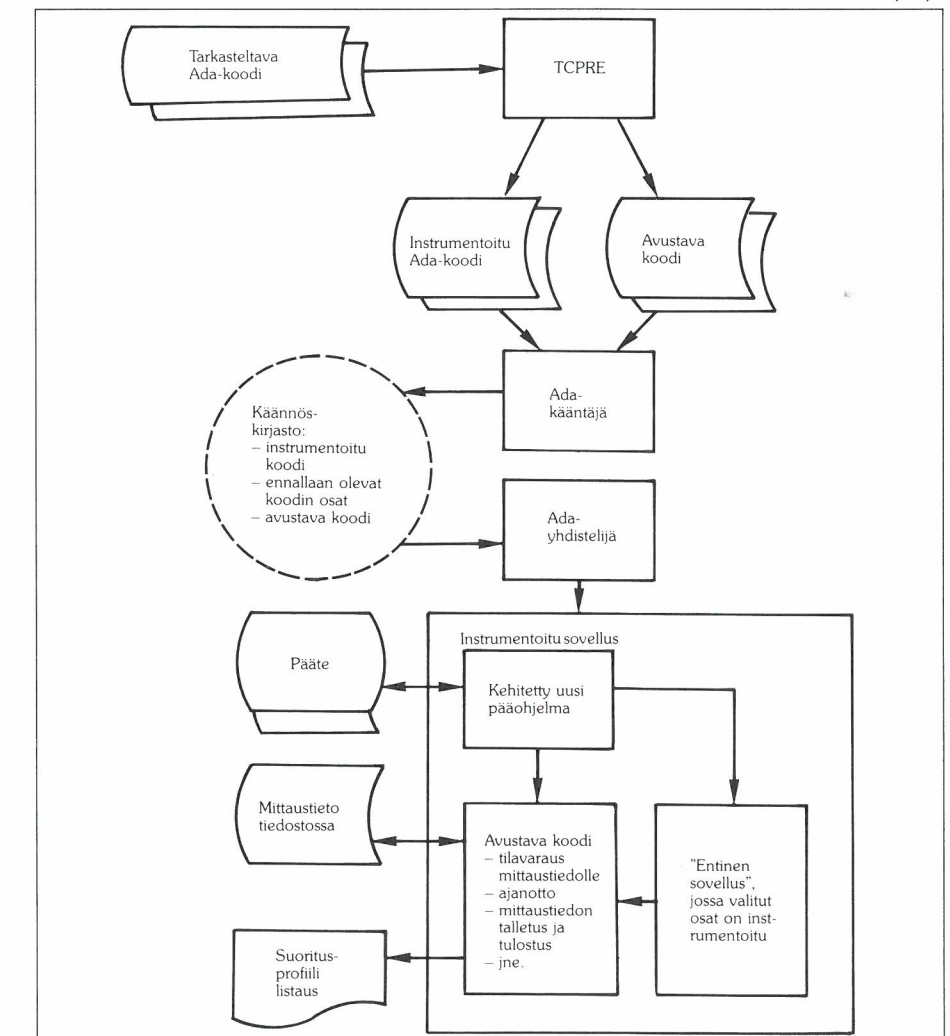
```
$INT := MYPACK.LEN("How long?")  
ASSERT $INT = 9
```

Jos annettu ehto on epätosi, se tarkoittaa, että testi ei toiminut odotetulla tavalla. Näissä tilanteissa testausohjelma on paljastanut mahdollisen virheen. Järjestelmä kirjoittaa ASSERT-poikkeaman esiintymiskohdan tunnistetiedot testausohjelman lokiin myöhemmin tehtävää tarkempaa selvittelyä varten.

Muita toimintoja ovat "on-line HELP", testitapausten lukeminen tiedostosta, testausohjelman lokin hallinta, testin kulun ohjaaminen komentokielen ehtorakenteella, aliohjelmien suoritusajojen mittaaminen jne.

Testisuunnitelman laatu tarkastetaan

Kun testausympäristö on onnistuneesti muodostettu, käännetty ja yhdistelty,



Kuva 2: TCMON-järjestelmän rakenne

Seuraavaksi testisuunnitelman laatu varmistetaan. Sopivia laadunvarmistajia ovat projektipäällikkö ja ne työtöve-rit, jotka tietävät, mitä kehitettävän ohjelmisto-osan tulisi tehdä. Laadunvarmistajien tehtävänä on tarkistaa erityisesti vain kaksi asiaa: ensinnäkin, että testisuunnitelmassa on tarpeeksi testitapauksia, ja toiseksi, että ASSERT-komentoja on riittävästi tarkistamassa testien tuloksia. Tässä laaduntarkastusvaiheessa ei tarvitse kiinnittää huomiota siihen, ovatko testitapaukset tai ASSERT-komennot oikein muotoiltuja. Jos niissä on virheitä, ne kyllä tulevat ilmi myöhemmin testausohjelmaa ajettaessa.

Systemaattinen testaus testausohjelman avulla

Kun testausohjelma on käynnistetty, ohjataan syöttötietojen lukeminen edellä muodostettuun testitapaustiedostoon. Sen jälkeen katsotaan ajosta syntyneitä loki-tiedostoa. Siinä on il-

moitukset niistä ASSERT-komennoista, joiden kohdalla ehto on ollut epä-tosi eli testin tulos on ollut eri kuin odotettu.

Nämä kohdat vaativat selvittelyä. Syy voi olla siinä, että testi oli muotoiltu väärin, että odotettu tulos oli väärä tai testattavassa ohjelmassa on virhe. Kahdessa ensinmainitussa tapauksessa "virhe" korjataan muokkaamalla testisuunnitelma kuntoon.

Kolmannessa tapauksessa virheen syyn selvittelyyn voidaan käyttää testausohjelmaa, jota ajetaan nyt jäljitysohjelman alaisuudessa. Testitapaukset annetaan keskustelumutoisesti tai tiedostosta testausohjelmalle ja jäljitysohjelmalla jäljitetään virheen paikka koodista.

Virheen korjaaminen vaatii suunnittelua, muokkausta, käännöksiä ja testausohjelman uudelleenyhdistelyä. Sit-ten koko testaus toistetaan alusta pitäen. Se käy helposti, kun testausohjelma hoitaa asian käsin koskematta.

Lopulta, tarvittavien testauskierros-ten jälkeen, kun testiajosta syntyvä lo-

ki ei enää sisällä ilmoituksia ASSERT-komennoista, on saavutettu tietty sel-keä välitavoite koodin testauksessa.

Tästä voidaan jatkaa vielä täydelli-sempään testaukseen. Testitapausten vaikutuksia tarkastellaankin ohjelma-koodin kannalta. Varmistetaan, että kaikki koodin osat on suoritettu vähin-tään kerran testien aikana. Jos niin ei ole, testiaineisto on ilmeisen puutteel-lista tai koodissa on tarpeettomia osia.

TCMONilla tutkitaan testauksen kattavuus

TCMON on Ada-työkalu, jolla voidaan selvittää testauksen kattavuus. Sitä voi-daan käyttää yleisemminkin, kun halu-taan vain tutkia (vaikkapa tuotantokäy-tössä olevan) Ada-ohjelman käyttäyty-mistä: kuinka usein kutakin ohjelman osaa suoritetaan ja kuinka paljon kun-kin (sisäisen) aliohjelman suoritusta vie aikaa. Kuvassa 2 (s. 19) on esitetty TCMON-järjestelmän rakenne.

TCMON-järjestelmä perustuu tarkas-teltavan Ada-koodin esikäsittelyyn. Ne Ada-käännösyksiköt, joita halutaan tar-kastella, annetaan lähdekielellisessä muodossa esikäsittelijälle (TCPRE). Aiempaa testausesimerkkiä jatkaak-semme esikäsittelijälle syötetään testat-tavat pakkaukset. Se tuottaa instru-mentoidut versiot näistä pakkauksista sekä hieman ajoaikana tarkastelua avustavaa koodia. Nämä käännetään ja yhdistellään ja tuloksena on instru-mentoitu sovellus.

Testiajot instrumentoidulla testausohjelmalla

Kuten testausohjelma, myös instru-mentoitu testausohjelma on normaali Ada-ohjelma. Kun se käynnistetään, kontrollin ottaa ensin pieni keskustelu-muotoinen pääohjelma. Sen avulla käyttäjä voi kutsua alkuperäistä sovel-lusta, esimerkissämme testausohjel-maa. Toiminnallisesti alkuperäinen oh-jelma ei ole muuttunut. Kun kontrolli on sen instrumentoiduissa osissa (tes-tattavissa pakkauksissa), esikäsittelijän lisäämä vähäinen koodi pitää kirjaa suorituserroista ja -ajoista.

Käyttäjää voi tallettaa kerätyt mittaus-tiedot tiedostoon. Tiedot voidaan tal-lettaa kasautuvasti, jolloin saadaan yh-

distettyä eri testausajokertojen tiedot.

Mittautiedoista voidaan tulostaa suoritustietojen listaus (ks. kuvassa 3 olevaa esimerkkiä). Siinä mittautulok-set on helppolukuisella tavalla yhdis-tetty alkuperäisen lähdekielellisen ohjel-man rivinumeroihin ja ohjausrakentei-siin.

Nämä toiminnot voidaan kutsua myös Ada-koodista, jolloin muokattua sovellusta ajettaessa käyttäjälle näkyvä toiminta on muuttumaton.

Yleisesti testauksen kattavuustietoi-hin liittyvä järjestyminen on seuraava: Jos jo-tain ohjelman osaa ei suoriteta kos-kaan, se on joko tarpeeton tai osoittaa puutteita testiaineistossa. Samanlainen johtopäätös voidaan tehdä sellaisista ehtolausekkeista tai niiden osaehtoi-sista, joiden arvo on ollut pelkästään jo-ko tosi tai epätosi.

Kun testauksessa on päästy siihen, että testausohjelmasta ei tule ilmoituk-sia ASSERT-komennoista ja TCMON osoittaa testauksen olevan kattavaa, voimme olla melko luottavaisia, että kirjoitettu koodi toimii haluamallamme tavalla.

Monia hyötyjä

Kun TBGENiä ja TCMONia käytetään täydessä mitassa Ada-ohjelmien (moduuli)testaukseen, testaustyön tuottavuus, hallittavuus ja tuloksen laa-tu paranevat merkittävästi. Pienem-mällä työmäärällä voidaan tehdä perinpohjaisempi testaus.

Näiden työkalujen avulla testaus on tehokasta, koska testausympäristön rakentaminen on nopeaa. Jos yhtä teho-kas testausympäristö ohjelmoitaisiin käsin, se voisi hyvinkin viedä viikkoja ja olla silti mahdollinen virhelähde.

Testitapaukset voidaan ajaa tiedos-tosta. Järjestelmä voi tarkistaa testitu-lokset. Käyttäjän tarvitsee vain käyn-nistää ajot. Epäonnistuneista testeistä tulee tiivis selvitys testausohjelman lo-kiin.

Testausohjelmalla voidaan tehdä keskustelumutoisia, ennalta suunnit-telemattomia kokeiluja. Testausohjel-ma ja jäljitysohjelma toimivat hyvin yhdessä virheiden selvittelyvaiheessa. Testauksen kattavuus ja koodin pullon-kaulat voidaan selvittää helposti.

Testaus on standardoitua. Eri pak-kauksille muodostettujen testausympä-ristöjen käyttöliittymät ovat samanlai-sia. Kun osaa käyttää yhtä, osaa käyt-

tää toistakin. Siirrettävyytensä ansiosta nämä työkalut ovat samanlaisina käy-tettävissä kaikissa Ada-koneissa. Testi-aineistot ja muut testaukseen liittyvät dokumentit ovat saman muotoisia eri testauskohteissa.

Testauksen edistymisestä voidaan kysyä monia konkreettisia kysymyksiä, joihin voidaan antaa lukumäärävas-taus, esimerkiksi montako testitapaus-ta on testisuunnitelmassa, kuinka monta ASSERT-komentoa on suoritettu ja moniko niistä on epäonnistunut, kuin-ka kattavasti tietty kohta on testattu jne.

Testauksesta dokumentit

Työkalut käyttävät ja tuottavat havain-nollisia, luettavia tekstitiedostoja, joita katselemalla voidaan nähdä missä vai-heessa testaus on. Ne toimivat myös sellaisenaan osana testauksen doku-mentointia. Seuraavat dokumentit syn-tyvät testauksen kuluessa:

- testausympäristön kuvaus, joka on TBGEN-ajosta syntynyt listaus ja kuvaa mille pakkauksille testausohjel-ma on kehitetty.

- testiaineisto, joka on käyttäjän tek-emä testisuunnitelma. Se kuvaa ne testitapaukset, joilla pakkaukset on tes-tattu.

- testiraportti on testausohjelman tuottaman lokin listaus. Käyttäjän va-linnan mukaan se on joko tarkka jälji-tys, joka kuvaa miten testiaineisto käyttäytyi testitilanteessa tai vain pieni yhteenvetoraportti. Jälkimmäisessä ta-pauksessa lähdetään siitä, että testiai-neistossa olevat ASSERT-komennot ovat tarkastaneet testien oikeellisuu-den. Molemmissa tapauksissa tulee ra-portin lopussa olla testausohjelman kirjoittama rivi:

"lkm> ASSERTs evaluated, 0 of them failed"

- kattavuusraportti on TCMON-jär-jestelmän tuottama listaus. Siitä näkyy, kuinka kattavasti koodi on testattu.

Kun nämä dokumentit on saatu val-miiksi ja niiden sisältö on hyväksyttä-vä, ohjelmoija ja hänen vastuullinen esimiehensä todella tietävät miten tes-taus on suoritettu. Hyvin testatun koo-din voi luottavaisin mielin antaa mui-den käyttöön. Tämän tason saavutta-mista voisi pitää myös (moduuli)tes-tauksen päättymishetken määritelmä-nä.

Vientiä myös ulkomaille

TBGENin ja TCMONin toteuttamises-sa on pyritty mahdollisimman hyvään siirrettävyyteen. Tässä on myös onnis-tuttu. Informaatiojärjestelmien tuoteke-hitysyksikössä työkalut on vaivatta voi-tu siirtää MPS 10:stä PC/AT-yhteenso-pivaan MikroMikko 3:een ja VAX:iin.

Työkalujen väkevät ominaisuudet ja sopivuus eri tietokoneisiin on houku-tellut myös Nokian ulkopuolisia Ada-käyttäjiä. Parhaillaan nämä työkalut ovat joko käytössä tai koekäytössä useassa maassa. Äskettäin niistä on myös solmittu ensimmäinen suomalai-sia Ada-ohjelmistojen koskeva jälleen-myyntisopimus. Arvostettu tanskalai-nen Ada-ohjelmistotalo DDC-Internat-ional A/S myy näitä työkaluja maail-manlaajuisesti.

Vientitoimitukset ovat yksi todiste siitä, että MPS 10:n yhteydessä kehite-tyt Ada-ohjelmistot ja -apuvälineet edustavat kansainvälisestikin katsoen huippulaatua. □

Kirjallisuutta:

- /1/ G.J.Myers: *The Art of Software Testing*, Wiley-Interscience, New York, 1979.
- /2/ TBGEN System Version 2.1, *User's Manual, February 1987*, Nokia Information Systems, Softplan.
- /3/ TCMON System Version 2.1, *User's Manual, March 1987*, Nokia Information Systems, Softplan.

```
*****
* TCMON System Version 2.1, (C) Copyright 1987 by Softplan
* Test Coverage Monitor / Program Bottleneck Finder
* Execution profile listing
*****
```

LINE NO.	DIAGR. OF EX. COUNTS	LE-VEL	STRUCTURE DESCRIPTION	Counters		Timers	
				START/ TRUE	END/ FALSE	AVERAGE TIME	TOTAL TIME
Source file: b:stackbod.ada							
2		1	pack STACK		0		0.0000
11		2	proc RESET		2	0.0001	0.0002
12**		2	begin	2	2		
13****>		3	for_loop	6	6		
24		2	proc PUSH		4	0.0002	0.0008
25****		2	begin	5	4 ?		
26****		3	if_then	5	4 ?		
			Condition	5	0 -		
			Subcondition_1	3	2 -		
			Subcondition_2	5	0 -		
29		3	elsif_then	0	0 -		
			Condition	0	0 -		
30		4	raise	0	0 -		
36		2	proc POP		3	0.0001	0.0003
37***		2	begin	3	3		
38		3	if_then	0	0 -		
			Condition	0	3 -		
39		4	raise	0	0 -		
40***		3	if_else	3	3		
46		2	proc SWAP_TOP		0		
47*		2	begin	1	0 ?		

... jne. kutakin instrumentoitua lähdeohjelmatiedostoa kohden

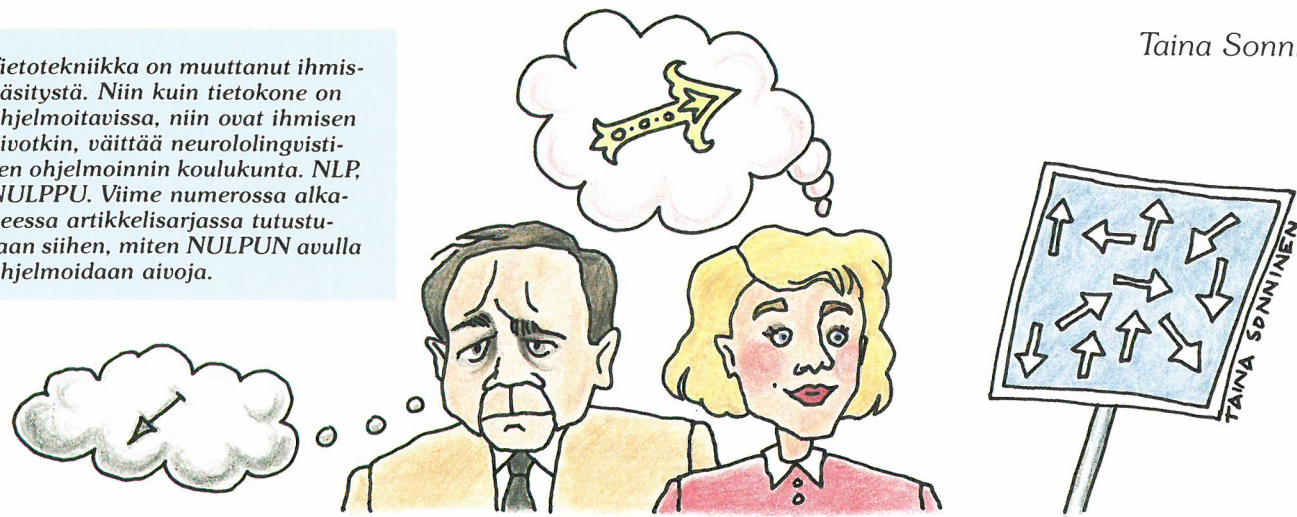
Kuva 3: Näyte suoritustietojen listauksesta

Fil.maist. Olavi
Poutanen toimii
Tampereen Soft-
planissa suunnitte-
lupäällikkönä.



Tietotekniikka on muuttanut ihmiskäsitystä. Niin kuin tietokone on ohjelmitavissa, niin ovat ihmisen aivotkin, väittää neurolingvisti- sen ohjelmoinnin koulukunta. NLP, NULPPU. Viime numerossa alkaneessa artikkelisarjassa tutustutaan siihen, miten NULPUN avulla ohjelmoidaan aivoja.

Taina Sonninen



Unelmat ohjaavat elämää

Tulevaisuudenkuvat ja niiden muokkaaminen eivät ole enää magiaa. Uskonnolliseen profetiointiin ja aivojen apinamaiseen rajallisuuteen ei tarvitse enää uskoa. Ihminen voi ohjelmoida päänsä näkemään tulevaisuuden uudella tavalla. Mielessä syntyviä tulevaisuudenkuvia voi ja kannattaa parannella, sillä niillä on taipumus toteutua. Ihminen ohjaa itseään valiten tiedostamatta ne elämänratkaisut, jotka tuottavat ennalta tunnetun olon.

Me luomme itse todellisuutemme. Aivoissa syntyvät mielikuvat määräävät ihmisen kulloisenkin tunnetilan – ei ulkoinen todellisuus. Tiedostaessaan ajatusmaailmansa peruselementit ihminen voi järjestää ne uudelleen.

Tietoisuutta tehostetaan – kyllä – haaveillen. Sillä unet ja haavekuvat heijastavat syvimpiä halujamme ja motiivejamme. Unien epätodellisuuteen sisältyy symbolisessa muodossa se, mitä ihminen todella haluaa.

Aikaa ei kannata käyttää murehtimalla epäonnistumista, jota ei ole vielä olemassakaan. On siltä energiaa tuhlausta käyttää mielikuvitustaan vain löytääkseen ne tulevaisuuden tapahtu-

mat, jotka voisivat tehdä olon kurjaksi juuri nyt.

Ja vaikka mieluisat mielikuvamatkat tuntuisivatkin toteutuvan ainoastaan omassa unimaailmassa, on muistettava, että nämä alitajuiset haaveet vaikuttavat ratkaisuihimme joka hetki.

Tiedostetut toiveet ja halut ovat paremmin kontrolloitavissa ja siten myös hyödynnettävissä. Mikä sitten tekee tiedostamattomasta tietoisien?

Valveuniharjoitus

Neurolingvistisen ohjelmoinnin ydin on harjoittelussa. Harjoituksen tarkoituksena on tehdä syvimät toiveet tie-toiseksi, jotta voit järjen kanssa pohtia, haluatko todella tulla sellaiseksi mistä haaveilet.

Järjestä olosuhteet niin, ettei sinua häiritä. Ota hyvä asento ja mieti jotain hyvin miellyttävää. Kuvittele itsesi viiden vuoden kuluttua – olet tekemässä juuri sitä mitä hulluimmassa haaveissasi olet aina halunnut tehdä. Älä sensuroi mitään, anna ajatuksillesi suihkuperämoottori. Potki pois ajatuspoliisit, jotka tolkkuvat mielikuvien mahdotto- muutta ja epätodellisuutta.

Missä olet. Katso ympärillesi, mitä teet, kenen kanssa. Kutsu mielikuva lähemmäksi, älä yritä millään tavalla tietoisesti luoda sitä. Tutki yksityiskohtia. Lisää miellyttävän kohteen valoisuutta, suurennat sitä ja lisää sen värikylläisyyttä.

Kuuntele. Mitä ääniä kuulet. Mitä puhutaan. Lisää miellyttävien äänien bassoa, voimakkuutta ja rytmiä.

Tarkkaile miltä sinusta tuntuu eri puolilla kehoa. Jos tunnet kosketuksen, vahvista kokemustasi lisäämällä siihen lämpöä, pehmeyttä, kosteutta.

Missä ja kuka olet kymmenen vuo-

den kuluttua? Mikä on muuttunut, mitä ajattelet viisi vuotta taaksepäin tapahtuneesta? Oletko tyytyväinen?

Jatka kuolinvuoteellesi. Katso elämäsi taaksepäin. Miltä se näyttää?

Sitten kuolema ja hautajaiset. Miltä se kaikki tuntuu? Minkälaisen elämän elit? Mitä elämältä halusit ja sait? Valitsitko oikein? Teitkö elämästäsi sen minkä halusit?

Nauti kaikesta ympärilläsi ja sisälläsi tapahtuvasta. Kertaa parhaat palat. Ja kun maltat, palaa hitaasti nykyhetkeen.

Mielikuvista tulevaisuudenkuvia

Toista harjoitus usein, varsinkin jos tunnelmaan pääseminen tuntuu vaikealta. Toistuvat ja tiedostetut valveunet käyvät yhä yksityiskohtaisemmiksi. Kun sama mielikuva jatkaa esiinmarssiaan kerta toisensa jälkeen, voit ryhtyä miettimään mitä toiveita ja haluja se heijastaa.

Mitä se viestittää, onko siinä mitään toteuttamiskelpoista. Millainen sinun tulisi olla ja millainen tapahtumaketju veisi lähemmäksi tuota haavekuvaa. Hetken mietittyäsi huomaat, että mahdollisuuksien määrä on valtava.

Tutustuessaan syvimpiin haluihinsa ja tulevaisuudenkuviinsa ihminen voi samalla tuottaa itselleen mielihyvää sisäisen morfiininsa, endorfiininsa avulla.

Ihminen on tavoitellut mielihyvää ja tietämystä tulevaisuudestaan ainakin yhtä ahkerasti kuin taivasosuuttaan. Kristallipallon arvoitukset ja maratonaarin ”juoksijan pilvi” voidaan nyt saavuttaa antamalla mielikuvien vilistä.

Uskallatko näpelöidä aivosoftaasi ja ottaa vastuun oman kohtalosi ohjelmasta? □

Thomas Branders

Kelpo kuvia ja kalvoja Freelance Plussalla

Lotuksen tuoteperheeseen kuuluva Freelance Plus -ohjelma on suunniteltu tyydyttämään lähes kaikki toimiston kuvantekotarpeet. Kun käyttäjä korkeahkon oppimiskynnyksen ylitettyään pääsee piirtämisessä vauhtiin, syntyvät kuvat kuin kuvat – esimerkiksi laadulliset kalvot – varsin helposti.

Freelancen kuvat koostuvat yksinkertaisista grafiikkaelementeistä: teksti, viiva, nuoli, suorakulmio, ympyrä, viipale, kaari, merkki ja monikulmio. Elementit ovat peruspalikoita, joiden yhdistelyssä vain oma mielikuvitus on rajoituksena.

Lotukselle tyypilliset valikot

Näyttö jakautuu kolmeen osaan. Pääosan kuvaruudusta muodostaa piirtämisalue. Sen yläpuolella ovat valikot. Oikeassa laidassa näkyy informaatiotaulu, jossa on tietoa piirroksista, oletusarvoista ja sillä hetkellä valittujen elementtien ominaisuuksista.

Piirtämisalue näyttää koko piirroksen tai sen jonkin osan. Kuvaa voidaan suurentaa ja pienentää, jolloin myös katseluikkuna, eli piirroksen kuvaruudulla näkyvä osa vaihtelee.

Freelancen komennot valitaan näytön ylälaidassa olevasta valikosta. Tekniikka on muista Lotus-tuotteista tuttu: painat komennon alkukirjainta tai siirät kohdistimen komennon päälle ja painat kuittausnäppäintä. Ohjausrasiaa eli hiirtä voi käyttää komentojen valitsemiseen ja itse piirtämiseen. Ilman ohjausrasiaakin pärjää hyvin.

Päävalikon komentojen takaa löytyy yhä uusia valikkoja. Enimmillään – tai pahimmillaan – valikkotasoa on neljä. Tottumaton käyttäjä eksyykin helposti valikkojen viidakkoon.



Muokkausta mielen mukaan

Kunkin grafiikkaelementin ominaisuudet ovat vapaasti muunnettavissa. Niinpä jokaiselle viivalle voidaan antaa haluttu leveys, väri ja tyyppi. Kaikenmuotoiset alueet voidaan täyttää erilaisilla kuvioinneilla ja väreillä. Monikulmion yhtä kulmaa voidaan siirtää toiseen paikkaan. Ja niin edelleen.

Käyttäjän mieltymysten mukaiset ominaisuudet voi tallentaa oletusarvoiksi seuraavia kuvia varten.

Muutoksia tehtäessä valitaan kohde ja toimenpide. Juuri kohteen valinnassa Freelance on monipuolisempi kuin muut piirrosohjelmat.

Muutoksen kohteena voi olla yksi tai useampi elementti tai symboli. Voimme yhdellä komennolla valita piirroksista tietyn alueen sisällä tai ulkopuolella olevat elementit. Samaten saamme valituksi vaikkapa kaikki piirroksessa esiintyvät ympyrät ja esimerkiksi täyttää ne mustalla.

Valittuihin elementteihin ilmestyvät pienet mustat neliöt valintaosoittimiksi. Niiden avulla elementtejä voidaan siirtää ja niiden muotoa voidaan muuttaa.

Tekstiä ristiin rastiin

Freelance sisältää suurehkon valikoiman erilaisia kirjasinmalleja, jotka tietyn rajoituksen ovat tulostimesta riippumattomia. Useimmat kirjasinmallit ovat vapaasti skaalattavissa haluttuun kokoon ja kallistettavissa.

Ohjelman kehittyneisyys osoittaa elementtien ja symbolien kallistussuhteen. Näin tekstitkin saadaan juoksemaan missä kulmassa tahansa.

Laaja symbolikirjasto valmiina

Organisaatiokaavioita tai systeemikuvauksia päivittäin tekevä osaa arvostaa Freelancen mahdollisuutta koota usein käytettävät symbolit kirjastoon. Yksi kirjasto sisältää aina korkeintaan kaksikymmentä symbolia.

Piirrettäessä ladataan kirjasto taustalle, josta toimintonäppäimen painalluksella haetaan jokin symboleista omaan piirrokseen. Symbolia voi suurentaa, pienentää tai muuttaa mielin määrin.

Jos symbolia halutaan muuttaa, se

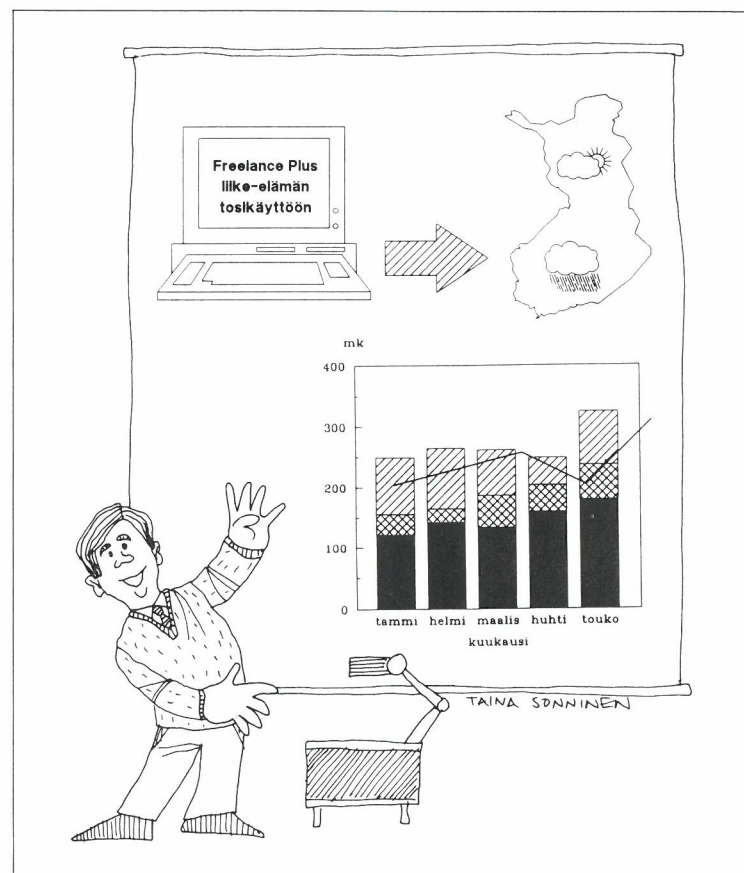
on ensin purettava elementeiksi, jotka sen jälkeen ovat erikseen muunneltavissa ja siirrettävissä. Lopuksi osat kootaan ja lukitaan jälleen symboliksi. Lukitus varmistaa, ettei vahingossa muuteta valmiiksi saatua kuvan osaa.

Freelancen mukana seuraa myös valmiita symbolikirjastoja. Niistä löytyy valmiina useita satoja enemmän tai vähemmän käyttökelpoisia symboleja.

Vähänkin suuremmissa organisaatioissa kannattaa usein käytettyjen symbolien teko keskittää yhdelle henkilölle, joka sitten jakaa kirjastoja tarvitseville. Näin esitysmateriaali saadaan yhtenäisemmän näköiseksi ja säästetään aikaa.

Apupisteillä kohdistaan

Piirrosohjelmalla saadaan aikaan siiste-



Plus antaa enemmän

Freelance Plus on Freelancen terästy versio. Sen tärkein parannus edeltäjänsä verrattuna on pylväs-, viiva- ja piirakkakuvioiden automaattinen luonti joko käyttäjän antamien lukuarvojen tai Lotus-taulukkolaskentiedostojen perusteella.

Uusi versio sallii myös vapaalla kädellä piirtämisen ohjausrasiaa tai piirustustablettia käyttäen.

Freelance Plus toimii hyvin yhdessä muiden Lotus-ohjelmien kanssa. Symphonyn ja 1-2-3:n valmiita kuvia voidaan ehostaa. Jos tehtävänä on tekstikalvoja, voi teksinkäsittelyohjelmasta siirtää tekstin Freelanceen, jossa sitä edelleen muokataan näyttävämpään muotoon.

Alku aina hankala

Freelancen kaikkia ominaisuuksia on mahdotonta sulattaa päivässä tai parissa. Ohjelman rutinoitumiseksi käyttäjäksi pääsee vasta utteran harjoittelun jälkeen.

Käsi kirjassa oleva itseopiskelujakso sopii riittävällä itsekurilla varustetuille henkilöille. Monelle kuitenkin kurssilla käynti lienee paras tapa päästä alkuun.

Laaja laitetuki

Freelancen näyttölaitteena pitäisi olla

Lisätietoja Freelance Plussasta saa Jari Luokkalta, puh. (90) 5671.

Anne Konu



ES-verkko on open apu

MikroMikko ES on oppilaitoksille suunniteltu mikrotietokone. Pieni, tehokas ja laajennettava ES soveltuu opetukseen peruskouluista korkeakouluhin.



MikroMikko ES:n voi hankkia oppilaiden ja opettajien henkilökohtaiseksi työvälineeksi. Laitteiden käyttöä opetuksessa tehostetaan liittämällä ne yhteen Nokia ES-opetusverkoksi. Verkkoon voi liittää MikroMikko ES:n lisäksi MikroMikko 3 ja 3TT -koneita.

Laitteet liitetään opetusverkkoon kiinteästi käsittely-yksikön pohjaan sijoitetulla opetusverkkoyksiköllä, sillä erillisiä verkkoyksiköitä ei ole haluttu oppilaiden työpöydille sijoittaa. Laitteen voi tarvittaessa väliaikaisesti kytkeä pois verkkokäytöstä.

Ope ohjaa verkkoa

Opettaja saa työpöydälleen erillisen verkon ohjausyksikön, jonka avulla hän pystyy vaivattomasti ohjaamaan opetusverkon käyttöä ja näin helpottamaan mikrojen käyttöä opetuksen työvälineenä.

Opetusverkon avulla opettaja voi siirtää oman mikronsa näytön kaikille tai vain osalle oppilaista ja opastaa näin jonkin harjoituksen suoritusta. Myös oppilaan näytön voi siirtää halutulle kohderyhmälle ja antaa hänen opastaa toisia. Tämä onkin usein pedagogisesti parempi vaihtoehto!

Oppilaita voi myös avustaa harjoituksissa asettamalla opetusverkon välille ohjeita tiedoksi. Oppilas saa ohjeet omaan näyttöön painikkeella.

Kokonaiskuvan luokansa työskentelystä opettaja saa selaamalla työasemien näyttöjä joko automaattisesti oppilaasta toiseen tai pysähtymällä aina mielenkiintoiseen kohteeseen pidemmäksi aikaa.

Opettaja voi ohjata oppilastaan "kädestä pitäen" ottamalla oppilaan työaseman haltuunsa ja käyttämällä sitä omalta näppäimistöltään. Opettajan

työaseman näppäimistön voi antaa myös oppilaan haltuun. Oppilas voi näin esittää omalta työasemaltaan ratkaisunsa toisille ikään kuin liitutaulla.

Tulostukseen automaattinen jonotus

Luokassa tarvitaan erilaisia oheislaitteita, joista tavallisimpia ovat kirjoittimet tulostusta ja modeemi linjaliikennettä varten. Nämä saadaan jokaisen oppilaan käyttöön liittämällä ne opetusverkkoon. Oheislaitteita voi olla kolme, ja kaksi samanaikaisesti käytössä. Useamman yrittäessä tulostaa yhtäaikaan verkko hoitaa automaattisesti tulostuksen jonotuksen.

Opettaja voi ohjata oheislaitteiden käyttöä myös omalta ohjausyksiköltään. Hän voi esimerkiksi antaa tietyille oppilaalle modeemin käyttöön toisten käyttäessä kirjoitinta.

Helpompaa tiedostojen siirtoa

Opettajan ajan suurin säästäjä lienee opetusverkossa toteutettu tiedostojen siirto.

Tunnille tullessaan opettaja voi siirtää opiskeltavan sovelluksen tai harjoituksen omalta työasemaltaan kaikkien työasemien levykkeille yhdellä kertaa. Tiedoston siirron voi myös rajata valitulle joukolle. Sovellusten kopiointi jokaisen oppilaan levykkeelle erikseen on näin tarpeetonta.

Myös oppilaat voivat siirtää tiedostoja työasemasta toiseen. Oppilas voi myös tallentaa tai käydä hakemassa tiedostoja opettajan työasemalta, joka toimii näin tiedoston siirron palvelimena.

MikroMikko ES (Education System) on 16-bittinen mikrotietokone, jonka suoritin on 8088-2. Käyttöjärjestelmä on MikroMikko 2:sta ja 3:sta tuttu MS-DOS, versio 3.2 tai uudempi. Työmuistia (RAM) ES:ssä on 640 kilotavua. Työskentely sujuu joko kahdeksan tai 4.77 megahertsin kellotaajuudella. Suoritusnopeutta voi lisätä myös 8087 -aritmetiikkasuorittimella.

Käsittely-yksikössä on lisäksi akkuvarmistettu reaaliaikakello ja kaiutin. Tulostamista, ohjausrasian (hiiren) käyttöä ja linjaliikennettä varten on laitteessa Centronics-tyyppinen rinnakkaisliitäntä ja sarjaliitäntä RS 232 C.

MikroMikko ES:n saa varustettuna yhdellä tai kahdella 720 kilotavun levykeasemalla, sekä yhdellä levykeasemalla ja 20 megatavun kiintolevyllä. Laitteeseen sopivaa 3 1/2 tuuman levykettä on mukava käyttää. Se on pienikokoinen ja täysin kestävä muovikuoren suojassa, mikä tekee siitä nimenomaan oppilaitoskäyttöön sopivan.

Näytöksi ES:ään haluttiin valita paras: tuttu nokialainen, väikkymätön positiivinäyttö, jossa on vakiona myös Hercules-grafiikka. Värinäyttö noudattaa EGA-standardia. ES:n näppäimistö on Nokia 102, joka on AT/E -tyyppinen.

Verkosta verkkoon

ES-järjestelmä pitää ovet avoinna tulevaisuuden ratkaisuille.

Luokassa opetusverkko on opetussellisesti tarkoituksenmukaisin ratkaisu. Jatkossa on ehkä tarpeen liittää luokka koulun sisäiseen lähiverkkoon ja yhdistää eri puolilla koulua sijaitsevat työasemat toisiinsa.

Hallintolaitteet voi seuraavassa vaiheessa yhdistää kunnan koululaitoksen atk-ympäristöön.

Paljon ohjelmia

MikroMikko ES on teollisuusstandardin mukainen. Nokialla ja muilla ohjelmistotoimittajilla on tarjolla runsas valikoima sopivia ohjelmia.

Ohjelmia löytyy tekstinkäsittelyyn, taulukkolaskentaan, tietoliikenteeseen, tiedonhallintaan ja ohjelmistokehitykseen. Myös tietokoneavusteiseen opetukseen sopivia ohjelmia on saatavana runsaasti.

MikroMikko ES:ään sopivassa Microsoftin Windows-ohjelma-ympäristössä on mahdollisuus tutustua erityyppisiin työvälineisiin tekstinkäsittelystä ja piirrosohjelmista kortisto- ja tietoliikenneohjelmiin saakka. Mukaan on sisällytetty muun muassa tekstin ja kuvien yhdistelyohjelma, laskin, kello, kalenteri, muistilehtiö ja peliohjelma.

Ohjausrasia nopeuttaa ja helpottaa ohjelmien käyttöä. Niitä voi pitää esillä yhtäaikaan omissa ikkunoissaan ja siirtyä sujuvasti ohjelmasta toiseen. Kiintolevyllä varustettu MikroMikko ES lisää Windows-ohjelmien nopeutta. □

Peruskoulunopettaja Anne Konu työskentelee Nokia Informaatiojärjestelmissä opetusjärjestelmien asiantuntijana.

Terhi Ogbeide



ERGONOMIA SUUNNITUS

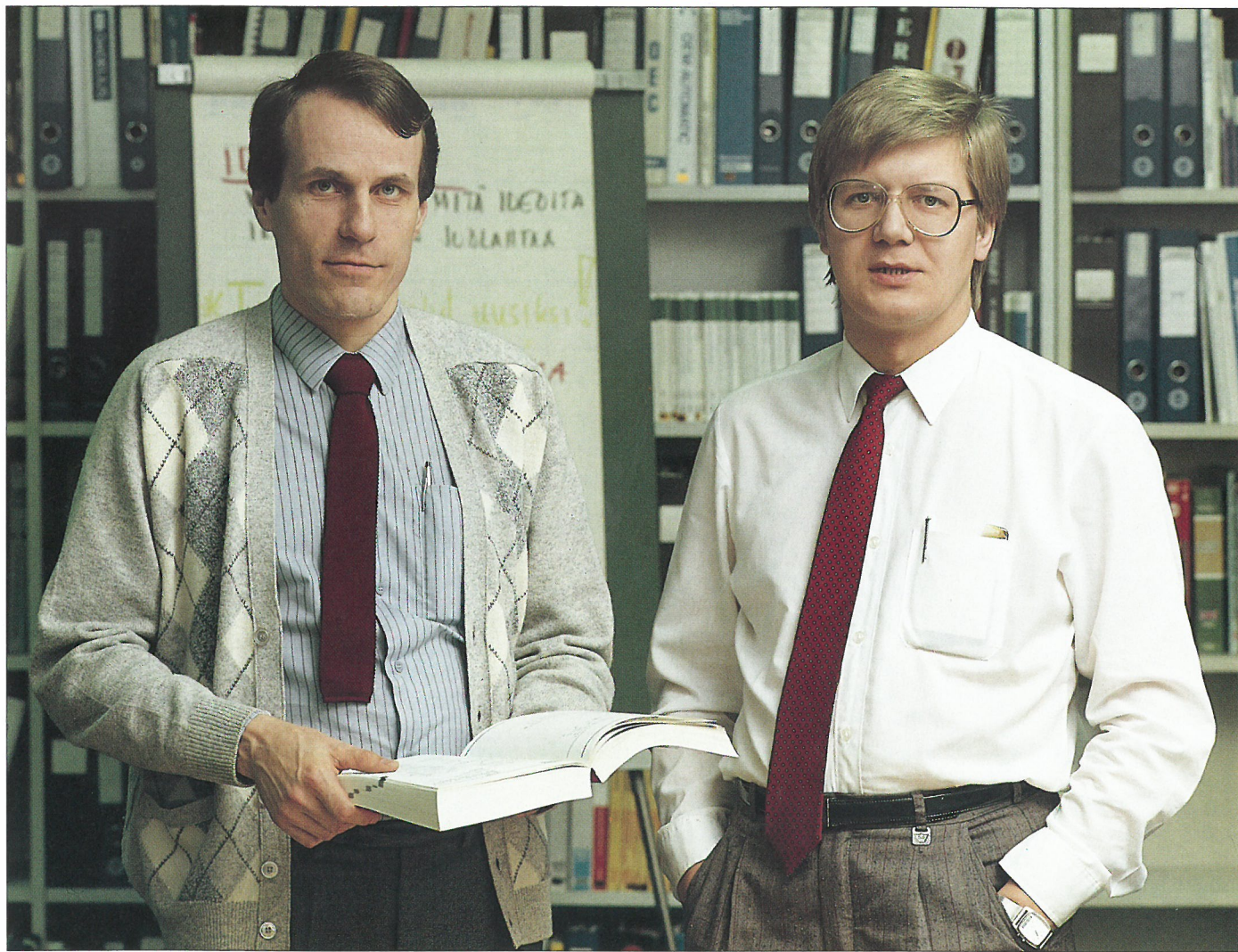
Laite- ja ohjelmistovalmistajat ovat tähän asti tienneet kysymättäkin, mikä asiakkaalle on parasta. Käyttäjän mielipidettä kuunnellaan vasta nyt, kun kilpailusyyt siihen pakottavat.

Nokialla on opittu jo kauan sitten, että tietokonevalmistaja pärjää vain asiakasta kuuntelemalla. Nokian mikrojen ergonomiset piirteet, kuten MikroMikko 2:een jo vuonna 1982 tehty väikkymätön positiivinäyttö, ovat tulosta tiiviistä vuoropuhelusta aktiivisen asiakkaan kanssa.

Se on kannattanut. Ergonomia on ollut Nokian valtti. Nyt kun ergonomiset piirteet yleistyvät maailmalla laitteiden perusominaisuuksiksi, Nokialla suunnataan ohjelmistoergonomiaan.

Ergonomiapanostukseen on kannustanut ennen kaikkea se, että Ruotsissa, yhdellä Nokian päämarkkina-alueista, ergonomia painaa ostopäätöksessä paljon. Ruotsalaiset asiakkaat ovat myös halunneet vaikuttaa lopputulokseen. Aktiivisin on ollut Ruotsin valtiollinen televiranomainen Televerket.

Televerket on yksi Ruotsin suurimpia tietokoneiden käyttäjiä. "Liiketoimintaamme eli kaikkea teletoimintaamme ohjataan tietokoneella. Tuhannet ihmiset työskentelevät päätteellä koko työpäivänsä. Siksi kannattaa sijoittaa työkalujen mukavuuteen ja ihmisten viihtyvyyteen", sanoo **Lennart Möller**, yksi Televerketin ergonomiasiantuntijoista. "Viihtyvyys motivoi, parantaa tuottavuutta ja vähentää poissaoloja", Möller perustelee. ►►



Näyttö kuin paperi

Möller pitää näyttöpäätteen hyvänä lähtökohtana vuosisatojen kuluessa hioutunutta kirjapainotaitoa: mustia selkeitä merkkejä valkoisella paperilla. "Miksi siitä yhtäkkiä luovuttaisiin", hän kysyy.

Televerketin toiveiden ja tutkimuksista saatujen vihjeiden perusteella MikroMikko 2:n näytöstä kehitettiin mahdollisimman paperinomainen.

"Televerket testasi erilaisia ehdotelmia eri valaistusolosuhteissa, jotta näytön taustaväri oikea valkoinen sävy, vaalean taustan ja tummien merkkien vastakohtaisuus ja jälkiloisto saatiin optimoitua. Lopullinen tulos oli sekoius kolmesta eri fosforista", toteavat laitteistokehityksestä vastaavat **Timo Kahelin** ja **Raimo Puntala** Nokia Informaatiojärjestelmien tuotekehitysyksiköstä. Kuvaputkivalmistaja Toshiba sai sekoituksesta kuvaputkiinsa perusfosforin.

Näytön kooksi valittiin 15 tuumaa. Vakaaksi ja välikymättömäksi se saatiin nostamalla kuvantoistotaajuus 71 hertsiiin.

Kentät kuriin

Möller, Kahelin ja Puntala laskevat tärkeäksi ergonomian osatekijäksi myös

"Meillä on pelkästään hyviä kokemuksia pohjoismaisesta ergonomia-yhteistyöstä. Monet ergonomiset piirteet kuten positiivinäyttö ja muotoilu, jotka Nokia toteutti Televerketin toivomuksesta mikroihinsa jo vuonna 1982, ovat olleet Nokian kilpailuvaltti. Vasta nyt ne alkavat yleistyä muualla mikrojen perusominaisuuksiksi", huomauttavat Timo Kahelin (vas.) ja Raimo Puntala, jotka vastaavat laittekehityksestä Informaatiojärjestelmien tuotekehityksessä.

turvallisuuden. Kun käyttäjät huolestuvat siitä, että näyttöpäätteiden säteily vaarantaisi heidän terveytensä, oli reagoitava.

"Kysymyshän on kahdesta erityyppisestä kentästä, magneetti- ja sähkökentästä, joita on kaikissa sähkölaitteissa. Niiden vaarallisuudesta kiistellään. Koska emme ole halunneet ottaa mitään riskejä, olemme poistaneet kentät", Raimo Puntala toteaa.

Pääteessä, kuten televisiossakin, kuva muodostetaan ampumalla elektronikyillä elektroneja kuvaruudun fosforille. Magneettikenttä syntyy kahdesta tykin suuaukon lähellä sijaitsevasta käämistä, jotka ohjaavat elektronisuihkua. Ohjauksessa nopeista magneettikentän muutoksista muodostuu hajakenttä.

Hajakentälle on kehitetty vastakkaisuuntainen magneettikenttä sijoittamalla kuvaputken ylä- ja alapuolelle kaksi lisäkäämiä. Kun elektronisuihkua ohjaava virta kulkee lisäkäämien kaut-

ta, ne kumoavat käyttäjään päin tulevan kentän. Magneettikentän voimakkuuden vaihtelu on näin saatu heikennetty kymmenekseen eli 20 milliteslaan sekunnissa 30 cm:n etäisyydellä näyttöpäätteestä.

Positiivisesti varautuneen kuvaruudun lasipinnan ja negatiivisesti varautuneen käyttäjän väliin muodostuva sähkökenttä on poistettu maadoittamalla kuvaruudun pinta.

Ergonomialle normit

Televerketille ergonomiasta on tullut niin tärkeä, että yrityksessä on yhteistyössä Ruotsin työsuojeluhallituksen, yliopistojen ja ammattiliittojen kanssa kehitelty ohjeisto päätteiden ergonomisuuden mittaamiseksi. Mitattaviksi suureiksi on valittu mm. kuvan värit, merkkien selväpiirteisyys ja staattinen sähkö. "Korostamme kuvan laatua, koska sillä on keskeisin asema tiedon välittämisessä koneesta ihmiselle",

Möller sanoo.

"Suosituksillamme on ollut selvä normittava vaikutus. Valmistajat ovat ottaneet vaatimuksemme huomioon erittäin kiitettävästi", Möller toteaa.

"Kun näyttöpäätteiden laadulle on saatu aikaan kansalliset vaatimukset, miksei niitä saataisi aikaan kansainvälisestikin", Möller miettii. Ruotsalaiset aloittivat muutama vuosi sitten keskustelut kansainvälisten standardien saamiseksi kuvan laadulle. Syksyksi saataneen valmiiksi ehdotus, joka alistetaan standardointijärjestö ISO:n jäsenäänestykseen.

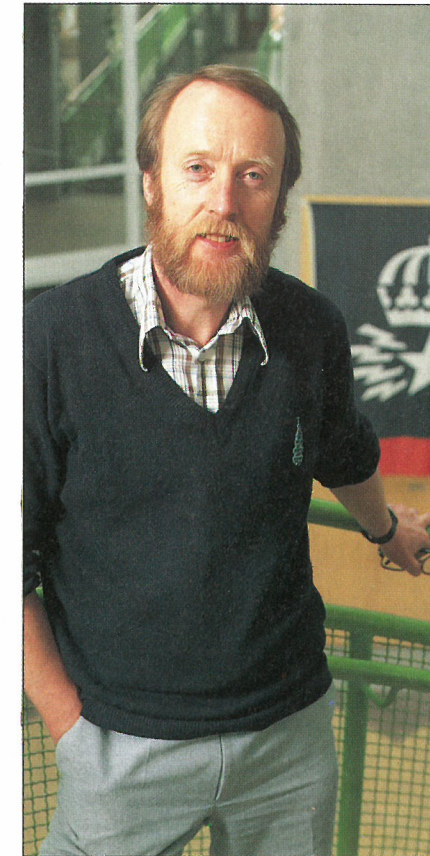
Möller muistuttaa kuitenkin, että ergonomiakeskustelu on keskittynyt liiaksi näyttöpäätteeseen. Tietysti tärkeää on päätteen muotoilu. Koneen on oltava miellyttävä ja helppo käyttää eikä se saa aiheuttaa fyysisiä vaivoja. Mutta miellyttävyyteen vaikuttavat niin monet muutkin, sekä ympäristö- että asennetekijät. Möller korostaa mm. organisatorisia seikkoja kuten organisaation joustavuutta, työn järjestelyä, koulutusta ja erityisesti ohjelmistojen laatua.

Ohjelmistotkin ergonomisiksi

Ergonomisuutta pitää vaatia myös ohjelmistoilta.

"Laskelmien mukaan 95 prosenttia tietotekniikan käyttökustannuksista on ihmisen ja koneen välisistä vuorovaikutustilanteista aiheutuvia henkilökus-

"Ergonomian päämäärä on työn tuottavuuden parantaminen. Skandinavisessa ergonomia-ajattelussa lähdetään siitä, että työ on tehtävä mahdollisimman mielekkääksi ja turvalliseksi, jotta se tuottaa. Työstä pyritään poistamaan hyvää oloa häiritsevät tekijät", toteaa menetelmäpäällikkö Veikko Kehä Nokia Informaatiojärjestelmien tuotekehityksestä.



"Ergonomia jää usein pelkäksi käyttäjäystävälliseksi puheeksi, kun sen täytyisi olla tietotekniikasakin itsestäänselvä lähtökohta ja joukko kouriintuntuvia toimenpiteitä", sanoo Televerketin ergonomia-asiantuntija Lennart Möller. Möller määrittelee ergonomian ihmisen ja koneen yhteispelin optimoinniksi.

Se on myös opittavissa nopeasti ilman käsikirjoja ja koulutusta. Käyttäjä tietää, mitä koneessa kulloinkin tapahtuu. Viestit ovat selkeitä ja tarvittaessa eritasoisia. Mahdollisesta virhetilanteesta ohjelma opastaa pois ja mukautuu vielä käyttäjän muuttuviin tarpeisiin.

Asenne- ja johtamisongelma

Ergonomiset piirteet suunnittelun lähtökohtana eivät kuitenkaan ole itsensänselvyys. Kehän mukaan suurin työ on yrittää muuttaa asenteita, jotta ohjelmiston tekijä oppisi asettumaan käyttäjän asemaan.

Kehä pitää ergonomiia myös johtamisongelmana. Jos aikataulu on tiukka ja resurssit niukat, lopputuloksena on ei-ergonominen ohjelmisto.

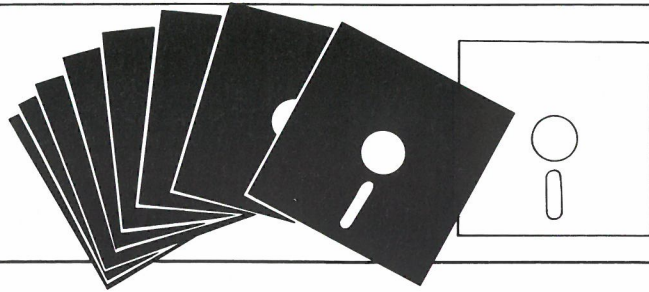
Kun ergonomia otetaan alusta asti suunnittelussa huomioon, suunnittelu-aika pitenee ja kustannukset lisääntyvät jonkin verran. Tämä ei ole kuitenkaan koko totuus. Sen mitä ergonomia maksaa, se tuo takaisin moninkertaisesti. "Asiakkaiden valitukset vähenivät olennaisesti", Veikko Kehä toteaa.

"Kun ylläpitokustannukset ovat yleensä reilut puolet ohjelmistokustannuksista, voidaan ergonomisilla ohjelmistoilla saavuttaa merkittäviä kustannussäästöjä. Jos yksinkertaisemmalla ohjelmistolla säästetään esimerkiksi yhden henkilön työaikaa päivittäin vaikkapa vain yksikin minuutti, siitä kasautuu vuosien kuluessa suuret summat", hän muistuttaa.

Ohjekortisto ja tarkistuslista

Ohjelmistoergonomian toteuttamisessa Nokialla käytetään rinnakkain ohjekortistoa ja tarkistuslistaa. Ohjelmiston tekijöillä on työvälineenä tiivis tietopakkaus siitä, mitä kannattaa ottaa huomioon alkaen esityskielen valinnasta suojausmenettelyihin. Ohjeistossa korostuu standardien noudattaminen ja pyrkimys yksinkertaisuuteen, selkeyteen, yhdenmukaisuuteen, kuvaavuuteen, joustavuuteen ja opastavuuteen.

Ohjelmistojen laatu tarkastetaan ryhmätyönä tarkistuslistan avulla sekä suunnittelu- että valmistumisvaiheessa. □



MikroMikko TopTen

Tamänkertainen MikroMikkojen ohjelmistolista perustuu elokuun 1987 myyntiluvuista laskettuihin suhteellisiin indekseihin.

Listalta voidaan heti nähdä tekstinkäsittely- ja taulukkolaskentaohjelmien myynnin jatkuvan voimakkaana. Niin ikään WS7300 ja WS220 -pääte-emulaattorit ovat käyneet hyvin kaupaksi.

MikroMikko TopTen

1. (5) WordPerfect	72
2. (1) WS7300-emulaattori	62
3. (2) Windows	39
4. (6) WordStar	38
5. (3) Multiplan	32
6. (4) WS220-emulaattori	25
7. (-) Edix	12
8. (8) Lotus 1-2-3	11
9. (-) Symphony	9
10. (-) Wordix	8

Suluissa sijoitus edellisessä Top-Tenissä (Softari 2/87).

WordPerfect tavuttaa nyt suomeksi

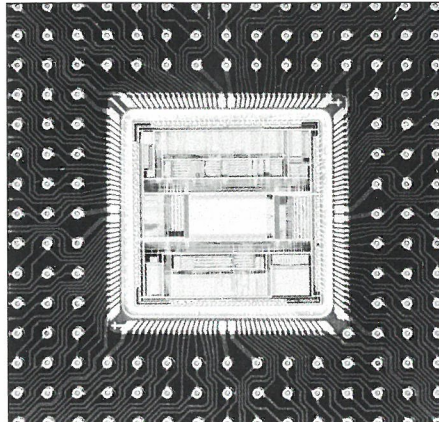
WordPerfect-tekstinkäsittelyohjelmasta on ilmestynyt uusi versio 4.2.

WordPerfect 4.2 toimii MikroMikko 2:ssa, 3:ssa, 3TT:ssä, Nokian PC-Net- ja MS-Net-verkoissa ja on yhteensopiva edellisen version 4.1 kanssa.

Uusina piirteinä ohjelmistossa on mm. automaattinen suomenkielinen tavutus, erillinen opetusohjelma, uudistettu sanakirja, uudet sarkaintyytit sekä mahdollisuus lisätä asiakirjaan reunahuomautuksia, jotka eivät näy lopullisessa tulosteessa.

Tekniset lisätiedot Ari Viitanen ja myyntitiedot Hannu Impola, Nokia, puh. (90) 5671. □

Uuden sukupolven Lisp-siru Symbolicsilta



Tekoälyn leviämisen esteenä on pitkälti ollut jakeluongelma. Miten siirtää tehokkailla Lisp-koneilla tehty järjestelmät edullisesti tuotantokäyttöön ja miten liittää tekoälysovellukset jo toimivaan atk-ympäristöön. Näihin ongelmiin tarjoaa käyttökelpoisen ratkaisun uusi Ivory-Lisp-siru.

Ivory-siru on ensimmäinen kaupalliseen käyttöön tarkoitettu Lisp-siru. Sen on kehittänyt amerikkalainen Symbolics Inc, joka on johtava Lisp-tietokoneiden valmistaja maailmassa.

Lisp osaksi muita koneita

Ivory-siru on helposti liitettävissä muihin koneisiin Lispiä tai yleensä symbolista käsittelyä suorittavaksi komponentiksi. Näin siru mahdollistaa varsinaisilla Lisp-koneilla kehitettyjen tekoälysovelluksien tuotantokäytön muisakin koneissa.

Tulevaisuudessa yhä useammat tie-

tokoneet ovatkin monisuoritinjärjestelmiä, joissa Lisp-suoritin toimii rinnan esim. MS-DOS -suorittimen sekä erilaisten erikoissuorittimien kanssa. Symbolics myy Ivory-sirua muille laitevalmistajille, jotka voivat käyttää sitä lisäsuorittimena koneissaan.

Siru myös Symbolics-koneisiin

Symbolics aikoo myös itse käyttää Ivory-sirua ensi vuoden aikana julkistettavassa uudessa tuotesarjassaan. Varsinainen Lisp-tietokone voidaan rakentaa sirun ympärille lisäämällä keskusmuisti ja tarvittavat liitäntäsirut. Kaikki nämä mahtuvat yhdelle piirikortille.

Sirun avulla saadaan Symbolics-koneisiin kolmikertaisesti lisää tehoa nykyisiin malleihin verrattuna. Myöhemmin tehon lisäys kasvaa jopa kymmenkertaiseksi. Lisp-siru mahdollistaa entistä laajempien ja monimutkaisempien tekoälysovellusten toteuttamisen.

Uutta arkkitehtuuria

Ivory-siru edustaa ns. kolmannen sukupolven Lisp-tietokonearkkitehtuuria. Se käsittelee 40 bitin sanoja, joissa 8 bittiä on varattu tyyppitarkistukseen ja 32 bittiä (4 tavua) datalle tai osoitteelle.

Datakentän pituus (32 bittiä) on yhteensopiva muiden yleisesti käytettyjen tietokoneiden kanssa, mikä yksinkertaistaa sirun käyttöä lisäsuorittimena.

Osoitekentän kasvu 32 bittiin mahdollistaa entistä suuremman virtuaalimuistin käytön.

Lisätietoja: Tero Haavisto, Nokia, puh. (90) 455 71. □

K-miehestä uusi versio

K-mies/2-tiedonhallintaohjelmiston uuden version 2.01:n toimitukset ovat alkaneet.

Se toimii MikroMikko 2:ssa, 3:ssa, 3TT:ssä. MS-Net-verkko-ympäristöön on saatavissa K-mies/2:n verkkoversio.

Ohjelmaan on lisätty uusia toimintoja, jotka tekevät siitä aiempaa joustavamman ja tehokkaamman. Ohjelmassa voidaan mm. käyttää hyväksi koko vapaana oleva muisti ja siihen voidaan

liittää Lattice C-kielellä ohjelmoituja lisätoimintoja.

K-mies/2 versio 2.01:n komentokieli on yhteensopiva edellisen version 2.00:n komentokielen kanssa.

Lisäksi MikroMikko 3:n ja 3TT:n K-mies/2 versio 2.01 tukee EGA-grafiikan käyttöä.

Tekniset lisätiedot Martti Melen ja myyntitiedot Hannu Impola, Nokia, puh. (90) 5671. □

Xi Plus suomeksi

Xi Plus -asiantuntijajärjestelmäkehittimestä on julkaistu suomenkielinen versio. Xi Plussan käyttö helpottuu entisestään, kun tietämys voidaan esittää selvällä suomen kielellä.

Englantilaisen Expertechin valmistama sääntöpohjainen kehitin on tullut Suomessakin hyvin suosituksi. Sillä on tehty mm. tiettävästi Suomen suurin mikropohjainen asiantuntijajärjestelmä, Osuuspankkien Keskuspankin ja Nokian tutkimuskeskuksen yhdessä kehittämä maatilalainojen asiantuntijajärjestelmä. Järjestelmä sisältää maatilalainojen myöntämiseen liittyvän monimutkaisen säännösten lähes kokonaan.

Uusia piirteitä

Suomenkielisyyden ansiosta asiantuntijajärjestelmän avulla voidaan ilmaista sellaisin lausein ja käsittein, että käyttäjä voi itse aktiivisesti kommentoida asiantuntijajärjestelmän sisältämän tietämyksen oikeellisuutta ja jopa tehdä itse muutoksia sääntöihin.

Vaikka tietämyksen suunnittelijan rooli on keskeinen sääntöjen kirjaamisessa järjestelmään, yhteisen kielen löytäminen nopeuttaa kehitystä huomattavasti ja parantaa asiantuntemuksen laatua.

Asiantuntijajärjestelmä voidaan nyt myös yhdistää muihin sovelluksiin. Xi Plussasta pystytään käynnistämään muita mikrossa toimivia sovelluksia kuten taulukkolaskenta- ja tietokantasovelluksia. Monissa sovelluksissa tällainen integraatio on erittäin tärkeää.

Edelleen uusi piirre on ns. lomake-syöttö. Sovellus ja käyttäjä voivat keskustella värikkäiden kuvaruutujen avulla ja useita tietoja voidaan kysellä ennen varsinaisen päättelyn aloittamista. Xi Plus pystyy yhdistelemään joustavasti lomakkeita myös päättelyn etenemisen aikana.

Lisätietoja: Jari Talvinen, Nokia, puh. (90) 437 61. □

MikroMikkojen ohjelmistoja erikoishintaan

Nokia tarjoaa erän merkittävien päämiestensä, kuten Microsoftin, WordPerfectin ja MicroPron tuotteita edulliseen hintaan. Ota pikaisesti yhteys MikroMikko-myyjääsi tai Nokian mikromyy-mälään, puh. (90) 567 4004. Tarjous on voimassa vain määräajan. □



Kirjastoille Riimi-informaatiojärjestelmä

Nokia on ryhtynyt myymään kirjastojen kanssa kehittämäänsä kirjaston tietojärjestelmää - Riimiä. Järjestelmä soveltuu yleisille kirjastoille. Se on käytössä Kuopion ja Lahden kaupunginkirjastoissa.

Kirjastojärjestelmään voi liittää pääkirjaston osastoineen, sivukirjastot, laitoskirjastojen toimipisteet ja kirjastoautot. Järjestelmä perustuu toimipisteiden työasemaverkkoihin, jotka käyttävät keskustietokoneen tietokantaa. Toi-

mipitejärjestelmät turvaavat asiakaspalvelun sujuvuuden keskuslaitteiston mahdollisten häiriöiden aikana.

Riimi koostuu lainausjärjestelmästä, kirjastoaineiston hankinta- ja luettelointijärjestelmästä, näitä tukevista nopeista ja monipuolisista hauista, tilastointi- ja raportointijärjestelmästä sekä yhteyksistä muihin kirjastojärjestelmiin ja valtakunnallisiin tietokantoihin.

Lisätietoja: Harri Lindfors tai Matti Tertsunen, Nokia, puh. (90) 5671. □

Uusia emulaattoreita myyntiin

Nokia on ottanut tuotevalikoimaansa kolme uutta pääte-emulaattoria.

Nokia AST-5251/11-emulaattori, versio 3.01, mahdollistaa MikroMikko 3:n ja MikroMikko 3TT:n liittäminen päätteeksi IBM:n System 34, 36 tai 38 -laitteistoihin. Yhdistämiseen käytetään twinax-kaapelia. Nokia AST-5221/11 sisältää sekä näytin- että kirjoitinemuloinnin.

Nokia AST-5251/12, versio 4.0, mahdollistaa etäisäyhteyden MikroMikko 3:sta tai 3TT:stä modeemin kautta IBM:n System 34:ään, 36:een tai 38:aan. Emulaattori sisältää näytin- ja kirjoitinemuloinnin.

IRMA 3278/3279, versio 1.49, mahdollistaa MikroMikko 3:n ja 3TT:n toimimisen IBM 3278 tai IBM 3279-2B -pääteinä. MikroMikossa voidaan käyttää muita MS-DOS-ohjelmia rinnan emulaattorin kanssa.

Teknisiä lisätietoja saa Jouko Salolta ja myyntitietoja Erhard Volkilta, Nokia, puh. (90) 5671. □

Nopeaan tiedon-siirtoon Diginet 64K-verkkopäätteellä

Nokia Informaatiojärjestelmät on tuonut markkinoille Diginet 64K -verkkopäätteen, joka toimii digitaalisen puhelinverkon modeemina. Koska Diginet 64K sisältää toimintoja, joita ei ole tavallisessa modeemissa, kutsutaan laitetta verkkopäätteeksi.

Verkkopäätettä voidaan käyttää valintaisessa digitaalisessa puhelinverkoissa tahdistettuun (synkroniseen) tiedon-siirtoon nopeuksilla 64000-1200 bit/s ja tahdistamattomaan (asynkroniseen) siirtoon nopeuksilla 19200-1200 bit/s. Verkkopäätteeseen liitetään tiedonsiirron päätelaite ja puhelin kuten modeemiin. Puhelin on käytössä tavallisen puhelinliittymän tapaan.

Alueelliset puhelinlaitokset ja PTL-Tele ovat aloittaneet Diginet-palvelun, johon palvelun tilaaja liitetään Diginet 64K -verkkopäätteellä.

Lisätietoja: Lippo Rantanen, Nokia, puh. (90) 567 3522. □

Nokia
Softari-lehti
PL 780
00003 HELSINKI

Vastauslahetus
HKI 10 Lupa 2509

Nokia
maksaa
postimaksun

Softari 3/87

Palvelukortti

☐ Haluan esitteitä seuraavista
Nokian palveluista ja tuotteista.

☐ Pyydän, että minuun otetaan
yhteyttä.

puhelin: _____
Olen kiinnostunut _____

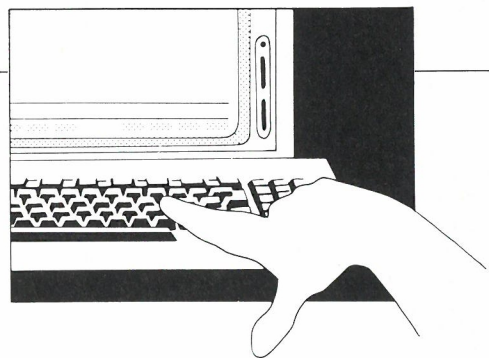
- ☐ Korjaan yllämainitut osoitetiedot.
☐ Poistan yllämainitut osoitetiedot.
☐ Lisään allamainitut osoitetiedot.
☐ Softarin ☐ NETin rekisteristä/rekisteriin.

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

JOUKKOJULKAISU



8/00010



Ooppiumia

Minun puolisoni uskoo tekniikkaan. Ensin meille tuli sähkölumilapio. Sitten lumentekokone, että lunta saisi luoda kesät talvet. Luominen on tekniikan ihmisellekin elintärkeää.

Luoja nähköön, että tekniikka tahtoo olla uskonto ja tietotekniikka supersellinen. Omine luomiskertomuksineen kaikkineen, ja tietoyhteiskuntaivaivaneen. Ja niin kuin kunnon uskonnon kuuluu, haluaa tietotekniikkakin naista niin kauan kuin se pysyy vaiti eikä naputa – paitsi näyttöpäätteellä. Nainen luotiin tietoyhteiskunnan huoltojoukkoihin. Kun huolto ei pelaa, ei tee mitään hyvällä softalla, ei hyvällä hardiksella.

Salaa siipaltani olen selannut lehtiä, lukenut jonkun kirjankin. Kertoilivat niissä uskonlahkolaiset toinen toisilleen perin juurin luomistyönsä tuloksia.

On keksitty tekoälyä ja robottia – kaukaista tulevaisuutta eivät ole silmälappuvideotkaan, jotka pakataan aivokuoremme viereen. Uusia elämyksiä, visioita. Uutta ooppiumia.

Jossain maailmassa saa vuokrata robotin tekemään työt, jotta mies itse pääsee luomaan uraansa golfkentälle. Kaikenlaisia elimiä ja toimintoja voidaan korvata. Kuukävelyn jälkeen kävelee pian lääkäri potilaansa suonon sisään.

Sen minkä tietotekniikka on kodillemme antanut on isänpojat, joiden kasvoja valaisee tietokonepelin laserpommitukset.

Sainpa kotiin minäkin oman pikku teknisen vempeleeni. Ehkä hiukkasan minulle vaikean. Sain imurin. Tosi hyperimutehoisen. Siinä minä seisoin vieressä ja odotin, että alkaisi siivota, että alkaisi noukkia sukankurttuja ja stumppeja. Ei se imenyt. Ei inahtanutkaan.

Itse minun oli siihen tartuttava. Raahattava perässä, tungettava sängyn alle, kiemurrettava hyllyn väliin. Uusi kone toisensa jälkeen tuli ja meni. Äiteen avuksi. Nyt minä niitä hoidan ja tilaan huoltomiestä, putkimiestä ja koneenkorjaajaa. Että joku edes kävisi. Kun koneet eivät käy. Ja aviomies käy kuin kone. Missä on minun vaipanvaihtoautomaattini, juokseva potta ja kävelevä räätitelineeni. Kuka keksisi tottelevan yötutin ja täyttyvän tuttipullon? Keksi ennemmin imuri kuuhun kuin kotikäyttöön. Sillä mitä siitä tulisikaan jos vaimo saisi aikaa ajatella?

*Softariajan
vaimo*