Juha Arffman
Anne Lahti
Kajaanin ammattikorkeakoulu
Hallinnon ja kaupan ala
Tietojenkäsittelyn koulutusohjelma

OPINNÄYTETYÖ

9.10.2000

Vuokatin Hiihtokoulun Varaustenhallinta- ja raportointiohjelma HiiVa

Kajaanin ammattikorkeakoulu

OPINNÄYTETYÖ
TIIVISTELMÄ

Ala	Koulutusohjelma
Kaupan ja hallinnon ala	Tietojenkäsittelyn koulutusohjelma

Tekijä(t)
Juha Arffman ja Anne Lahti

Työn nimi
Vuokatin Hiihtokoulun Varaustenhallinta- ja raportointiohjelma HiiVa

Vaihtoehtoiset ammattiopinnot	Ohjaaja(t)
	Ritva Pihlajaniemi
Aika	Sivumäärä
Syksy 2000	44+12

Tiivistelmä

Opinnäytetyömme käsittelee tietokannan käsittelyyn ja hallintaan luotuja toimintoja Visual Basicissa, Access-tietokantaa, varaustenhallintajärjestelmää ja seurantaraporttien luomista sekä ohjelmistoprojektin toteuttamista. Varauksien tekeminen ja niiden järjestelmällinen hallinta ovat tärkeimpiä Vuokatin Hiihtokoulun tietojenkäsittelyn tehtävistä. Raportoinnin avulla hoidetaan tietyt päivittäiset ja viikottaiset tilitykset, palkanmaksu sekä kausittainen toiminnan seuranta.

Visual Basic on selkeän graafisen käyttöliittymänsä sekä useiden ohjelmointia edesauttavien toimintojensa ansiosta helppo ohjelmointiympäristö. Tietokantojen käsittelyyn Visual Basic -ympäristössä on melkein alusta lähtien kuulunut olennaisena osana objektimallit, joiden avulla tauluja ja niihin tallennettuja tietoja käsitellään. Visual Basic versioon 6.0 on kehitetty lisää uusia hyödyllisiä toimintoja tietokantoihin, Web-käyttöön ja raportointiin liittyen. Data Environment on työkalu, jonka avulla tietokantaliittymän luominen ja keskittynyt hallinta on yksinkertaista ja luotettavaa. Ensimmäistä kertaa Visual Basiciin sisältyy myös raportointiympäristö Data Report, jolla voidaan luoda ammattimaisia ja monihierarkkisia tulosteita.

Hiihtokoulutoiminta Suomessa on lisääntynyt paljon. Useimmat hiihtokoulut toimivatkin yrityspohjalta ja niissä työskentelee päätoimisia opettajia. Harrastuksena laskettelun ja lumilautailun suosio kasvaa edelleen ja asiakkaina on yhä enemmän ulkomaalaisia. Tehokkaan ja luotettavan toiminnan turvaamiseksi hiihtokoulut tarvitsevat atk:lla hoidetun varaustenhallinnan ja tuloksen seurannan.

Toteutimme Vuokatin Hiihtokoulun varaustenhallinta- ja raportointiohjelman (HiiVa) ohjelmistoprojektina ohjelmoimalla Accessin tietokantaa Visual Basicin versiolla 6.0. Teimme graafisella käyttöliittymällä varustetun stand-alone -ohjelman, jolla korvataan hiihtokoulun opetustuntien varausten seurannassa käytetty varauskirja ja lisäksi palkanmaksussa ja opetustuntien seurannassa käytetyt kuitit sekä kassakone. Tietokantaan tallennetut tiedot myös mahdollistavat hiihtokoulun tarvitsemien erilaisten yhteenvetoraporttien luomisen.

Luottamuksellisuus	
Hakusanat	Visual Basic, Data Environment, Data Report, varaustenhallinta
Säilytyspaikka	Kajaanin ammattikorkeakoulun kirjasto

Kajaanin ammattikorkeakoulu

ABSTRACT OF THE FINAL YEAR PAPER

Faculty	Degree programme
Faculty of Business and Administration	Data Processing

Author(s)
Juha Arffman and Anne Lahti

Title
The Reservation Management System and Reporting Program for Vuokatti Ski School (HiiVa)

Alternative professional studies	Instructor(s)
	Ritva Pihlajaniemi
Date	Total number of pages
Autumn 2000	44+12

Abstract

Our thesis is concerned with the Visual Basic operations for the manipulation and management of databases, Access database and carrying out the software project. It also accesses the reservation management system and control report generation. Making reservations and management of them are the most important tasks of data processing at Vuokatti Ski School. The particular daily and weekly accounts, salary payment and business observations of the skiing seasons are handled with the aid of reports.

Visual Basic is a quite comfortable programming environment because of its clear and well-defined graphical user interface. It also includes many actions, methods and objects, which contribute to programming. Almost from the beginning there have been object models for manipulating the databases. With these models it is possible to handle tables and data. The version 6.0 has been developed to include more new useful actions concerning databases, Web use and reporting. Data Environment is a tool which make it easier and safer to create and manage database connections. For the first time Visual Basic contains the reporting environment Data Report for the generation of the professional and multi-hierarchical printouts.

The ski school business in Finland is growing more and more. Most schools practise like business enterprises and have teachers on a full-time basis. Downhill skiing and snowboarding are becoming more popular as hobbies and there are numbers of foreigners. With the help of computer systems, schools can ensure their efficient and secure business.

We designed and implemented The Reservation Management System and Reporting Program (HiiVa) as a software project. The program was carried out by the database of Microsoft Access 97 and the programming was done by Visual Basic Enterprise 6.0. This stand-alone program has a graphical user interface and its main target is to replace the reservation book and notes which are used in the salary payment and follow-up of the lessons. From the information of the database it is also possible to get different summarized data in reported form.

Confidentiality status

Keywords	Visual Basic, Data Environment, Data Report, reservation management
Deposited at	Kajaani Polytechnic Library

SISÄLTÖ

TIIVISTELMÄ

ABSTRACT

1 JOHDANTO 1

2 VISUAL BASIC JA TIETOKANNAT 3

2.1 Objektimallit 3

2.1.1 Data Access Objects 3

2.1.2 Remote Data Objects 4

2.1.3 ActiveX Data Objects 4

2.1.4 Data Environment 5

2.2 Projektin päivittäminen 5

3 DATA ENVIRONMENT- JA DATA REPORT –TYÖKALUT 8

3.1 Yleistä Data Environmentista 8

3.1.1 Connection 9

3.1.2 Command 10

3.1.3 Sidotun ohjaimen liittäminen 10

3.1.4 Data View –ikkuna 10

3.1.5 Tietokantaprojektin luominen 11

3.2 Yleistä Data Reportista 14

3.2.1 Raportin luominen 15

3.2.2 Raportin laajentaminen 17

3.2.3 ToolBoxin ohjaimet 19

4 VUOKATIN HIIHTOKOULUN HISTORIAA JA TOIMINTAA 21

5 HIIVA–PROJEKTI 23

5.1 Organisaatio 23

5.2 Alkutilanne 24

5.3 Tavoite 24

5.4 Suunnittelu 25

5.5 Ohjelman toteutus 26

5.5.1 Tietokanta 26

5.5.2 Käyttöliittymä 28

5.6 Testaus 34

5.7 Asennus 34

5.8 Käyttöönotto 34

6 POHDINTAA 35

6.1 Visual Basic työvälineenä 35

6.2 Projektista ja omasta työstä 36

6.3 Tulevaisuus 37

KÄSITTEET 39

LÄHTEET 42

LIITTEET 44

1 JOHDANTO

Tänä informaatioteknologian aikakautena atk:n hyödyntäminen alalla kuin alalla on tullut yhä tärkeämmäksi, melkein välttämättömäksi. Tämän asian huomasi myös Juha Arffman työskennellessään Vuokatin Hiihtokoululla hiihdonopettajana. Vielä viime laskettelukautena 1999-2000 hiihtokoulun kaikki tuntivaraukset ja dokumentit tehtiin manuaalisesti.

Manuaalinen tietojenkäsittely on hidasta ja hiihtokoulun toiminnan ollessa vilkkaimmillaan se vei liikaa työaikaa hiihdonopettajilta. Siitä Juha sai idean toteuttaa hiihtokoululle ohjelma varausten tekemiseen ja tarvittavien dokumenttien kuten laskujen ja raporttien laatimiseen. Keskusteltuaan asiasta hiihtokoulun puheenjohtajan Heidi Kelhon kanssa ja tämän pitäessä ideaa loistavana, Juha päätti aloittaa tiedustelut asian eteenpäin viemiseksi. Hän kertoi hyvälle ystävälleen Anne Lahdelle mahdollisesta tulevasta projektista päättötyönä ja näin Anne lähti mukaan projektiin.

Aloitimme ohjelman suunnittelun tammikuussa. Tarvittavien toimintojen kartoittamisen ja tietokannalta vaadittavien ominaisuuksien selvittelyn jälkeen alkoi toteuttamiseen sopivien työkalujen etsiminen. Tutustuimme erilaisiin potentiaalisiin työkaluihin internetin avulla ja haastattelemalla opettajia. Päätimme kuitenkin toteuttaa ohjelman käyttäen ohjelmointiympäristönä jo ennestään tuttua Visual Basicia ja tietokantana Microsoft Accessin kantaa.

Aluksi kyseisen ohjelman toteuttaminen parityönä vaikutti suppealta aiheelta opinnäytteeksi, mutta huomasimme sen laajentuvan ohjelman valmistuessa lisääntyneiden toiminnallisuustarpeiden vuoksi. Ohjelman tekemisen yhteydessä

opiskelimme Data Environment- ja Data Report -toimintojen käyttöä. Ne ovat Visual Basic version 6.0 uusia tietokannan hallintaan, käsittelyyn ja raportointiin luotuja objekteja. Niitä käytetään sovellettuina Vuokatin Hiihtokoulun varaustenhallinta- ja raportointiohjelmassa. Opinnäytetyömme valmistuessa opimme käytännössä, mitä todellisen ohjelman toteuttaminen ja tietokannan käsittely koodaamalla vaatii.

2 VISUAL BASIC JA TIETOKANNAT

Tietokantaohjelmointi Microsoftin Visual Basicilla ei ole aina ollut niin helppoa ja loogista, mitä se on tänään. Tämä johtuu aikaisemmin käytetyistä monimutkaisista teknologioista, jotka ovat kehittyneet muun teknologian kehityksen mukana vastaamaan nykypäivän ja tulevaisuuden vaatimuksiin.

2.1 Objektimallit

Visual Basicin tietokantojen käsittelyssä on aina ollut olennaisena osana objektimallit, joiden avulla tauluja ja niihin tallennettuja tietoja on voitu käsitellä lisäämällä, muokkaamalla ja poistamalla. Nämä kaikki olemassa olevat Microsoftin tietokantaohjelmoinnin paradigmat ovat edelleen täysin tuettuina Visual Basicin uusimmassa versiossa 6.0.

2.1.1 Data Access Objects

Visual Basicin ensimmäinen objektimalli Data Access Objects (DAO) esiteltiin VB:n versiossa 3, jossa ensimmäisen kerran pystyttiin luomaan tietokantayhteyksiä. DAO-mallia on päivitetty jokaiseen Visual Basic -versioon. DAO on suoraan hierarkkinen ja näin ollen se ei erota yhteyksiä ja Recordset -objekteja, vaan vaatii avoinna olevan yhteyden ennen Recordset-objektin muodostamista. Isäobjektin instanssi kuten Database-objekti täytyy luoda ennen kuin lapsiobjekteja voidaan tehdä. DAO:lla ODBC-tietolähteen (Open DataBase Connectivity) käyttäminen vaatii kaikkien pyyntöjen välittämisen Jet-moottorin

kautta. Jet-tietokantamoottori on teknologia, jota käytetään Microsoft Access –tauluissa ja muissa tiedon lähteissä sijaitsevan tiedon saamiseksi. Jet-moottorin käyttö hidastaa toimintoja selvästi. Kuitenkin DAO on edelleen suosittu ja tehokas työkalu yhden järjestelmän sovelluksiin ja verkkoihin pienistä keskisuurille työryhmille. (Halvorson 1998, Rahmel 1999)

2.1.2 Remote Data Objects

Remote Data Objects (RDO) oli ensimmäisen kerran mukana VB 4:ssä ja se tarjosi yksinkertaisen liittymän tietokantoihin ODBC -ohjainten kautta. RDO teki mahdolliseksi nopean liittymän matalan tason ohjelmointiliittymiin, jotka olivat aikaisemmin vain C++:n tapaisten järjestelmätason ohjelmointikielten käytössä. RDO tarjoaa paljon DAO:n tapaan ohjelmoidun liittymän, mutta se ohittaa Jet-moottorin tason. RDO:ia käyttävät monet tietokantakehittäjät, jotka työskentelevät Microsoft SQL Serverin, Oraclen ja muiden isojen relaatiotietokantojen kanssa. (Halvorson 1998, Rahmel 1999)

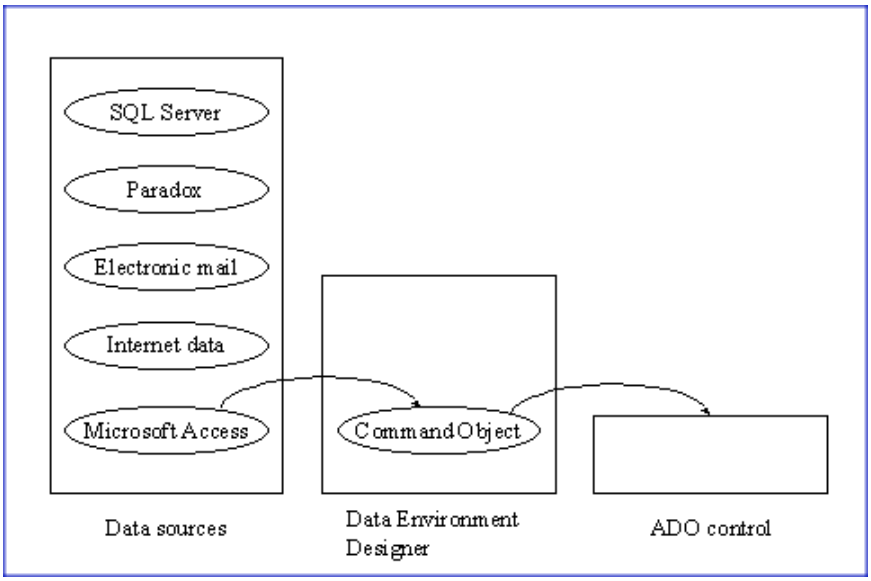
2.1.3 ActiveX Data Objects

ActiveX Data Objects –malli (ADO) suunniteltiin kahden edellä mainitun mallin jatkajaksi. ADO oli ensimmäisen kerran mukana Microsoft Web -palvelimessa nimeltään Internet Information Server (IIS). Vaikka DAO-mallia käytettiin ensisijaisesti Visual Basicissa tietokantojen käsittelyyn, Microsoft on asettanut ADO:n perusmalliksi riippumatta käytettävästä teknologiasta. ADO:a voidaan käyttää IIS, VBA, VB, VBScript, Visual C++, Visual J++ ja monissa muissa ohjelmointiympäristöissä. Kaikissa edellä mainituissa ympäristöissä käytetään yhtä objektimallia ja näin säästytään lukuisien eri teknologioiden asentamiselta. ADO:n ominaisuuksissa merkittävä parannus DAO:iin verrattuna on sen kyky yhdistää tietokantayhteyksiä, jotka selvästi parantavat järjestelmien tehokkuutta. ADO on myös paljon yksinkertaisempi vanhempiin malleihin verrattuna. Se sisältää vain kolme pääobjektityyppiä, kun muissa malleissa objektikokoelmat ovat paljon laajempia. (Halvorson 1998, Rahmel 1999)

2.1.4 Data Environment

Data Environment ei ole ollut missään aikaisemmassa Visual Basic -versiossa ennen 6.0 versiota. Versiossa 5.0 oli lisätoimintona UserConnection -niminen avuste yhteyden luomiseen. Sitä ei voinut sitoa dataperusteisiin ohjaimiin ja siksi sen käyttö ei ollut kovinkaan laajaa. Sen sijaan Data Environmentin luomisen jälkeen kaikki sen sisältämät objektit ovat käytössä sovelluksen jokaisessa osassa. (Halvorson 1998, Rahmel 1999)

Data Environmentiin sisältyvällä Data Environment Designerilla voidaan itse luoda uusilla tavoilla objektikomentoja. Seuraava kuvio selventää tiedonlähteiden välisiä suhteita.



Kuvio 1. *Tiedonlähteiden väliset suhteet, ADO-komento-objekteja ja ohjelma, joka käyttää ADO-resursseja. (Halvorson 1998, 594)*

2.2 Projektin päivittäminen

Todennäköisesti jokainen ohjelmoija joskus kohtaa Visual Basicin vanhemmalla versiolla tehdyn projektin. Tällöin voi tulla eteen tilanne, jossa projekti sovitetaan uusimmalle Visual Basicin versiolle.

Päivittämisessä perusmuunnokset ovat aika selviä ja koska itse ohjelmointikieli on täysin yhteensopivaa taaksepäin, sen kanssa ei luultavasti kohtaa kovin

monia rakennevirheitä. Ongelmia saattaa esiintyä silloin, kun päivitys sisältää muunnoksen 16-bittisestä 32-bittiseen käyttöjärjestelmään. VB kuitenkin muodostaa automaattisesti loki-tiedoston päivityksen aikana tulleista mahdollisista ongelmista. Tiedostossa kuvataan jokainen virhe erikseen, mikä helpottaa niiden korjaamista. Loki-tiedosto nimetään automaattisesti ongelmia aiheuttaneen tiedoston alkuperäisen nimen mukaan käyttäen log-lopputunnistetta. (Rahmel 1999, 297 - 299)

Projektin päivittäminen on normaalisti luotu automaattiseksi toiminnoksi. VB:n versio 6 tunnistaa VB 4:llä (32-bittisellä versiolla) tai VB 5:llä tehdyt projektit. Päivityksen alkuvaiheessa ohjelma ainoastaan tiedustelee halukkuutta päivityksen suorittamiseen. (Rahmel 1999, 297 - 299)

Visual Basicin suosio kasvoi paljon 16-bittisten Windows-järjestelmien aikana ja tämän vuoksi erityisesti VB 3:lla tehtiin paljon sovelluksia. Tästä johtuen päivitettäessä 16-bittistä versiota 32-bittiseksi, on versio 4 avainsana sen tekemiseen. Visual Basic 4 julkaistiin samoihin aikoihin kuin Windows 95, joka ensimmäisenä teki 32-bittisistä versioista standardin ja siksi sen rakenne on tehty soveltuvaksi toimimaan yhdistäjänä näiden kahden maailman välillä. Esimerkiksi VB 3:lla tehty projekti kannattaa ensin päivittää toimivaksi VB 4 -projektiksi ja sen jälkeen VB 6 -projektiksi. (Rahmel 1999, 297 - 299)

Ennen OLE (Object Linking and Embedding) -kontrolleja, jotka tunnetaan myös nimellä OCX (OLE Custom Controls) tai ActiveX, oli olemassa VBX (Visual Basic eXtension) -kontrolleja. Ne ovat 16-bittisiä komponentteja, joten niitä voidaan käyttää vain VB 3:ssa ja VB 4:ssä. Päivitettäessä 16-bittinen järjestelmä 32-bittiseen järjestelmään, tarvitaan luonnollisesti VBX -komponenttien kanssa yhteensopivia OCX -komponentteja. Kaikille Microsoftin aikaisemmin luomille kontrolleille on saatavilla vastaavat OCX -kontrollit. (Rahmel 1999, 299)

Päivitettäessä projektia saattaa ongelmaksi muodostua epätietoisuus käytettävistä kirjastotiedostoista. Markkinoilla on olemassa suomalainen englanninkielinen Project Analyzer -ohjelma, joka on ensisijaisesti tarkoitettu VB-koodin optimointiin, ohjelmointityylin tarkasteluun ja projektin toiminnallisuuteen liittyvien ongelmien etsimiseen sekä korjaamiseen. Ohjelman

ajamisen jälkeen saadaan yksityiskohtaisia tietoja sekä korjausehdotuksia koodin ja rakenteen parantamiseksi. Lisäksi ohjelma selvittää myös projektin käyttämät DLL- (Dynamic Link Library) ja OCX-tiedostot. Tätä ominaisuutta voidaan käyttää hyväksi myös päivitystilanteessa, ainakin kirjastotiedostojen osalta. Project Analyzer -ohjelma toimii 32-bittisissä Windows-järjestelmissä ja analysoi VB 3, 4, 5 ja 6 -koodeja. (http://aivosto.com/project/suomi.html)

2 DATA ENVIRONMENT- JA DATA REPORT -TYÖKALUT

Data Environment ja Data Report ovat kaksi Visual Basic 6:n hyödyllisimmistä uusista työkaluista Visual Basic 6:ssa. Tämän osion tarkoituksena on esitellä niiden ominaisuuksia sekä niiden luomiseen ja käyttämiseen liittyviä toimintoja tarkemmin.

3.1 Yleistä Data Environmentista

Aikaisemmin liittymät tietolähteeseen muodostettiin erillisillä ConnectionString- ja RecordSource-ominaisuuksien asetuksilla. Data Environment yksinkertaistaa huomattavasti liittymän luomista ja tarjoaa selkeän ympäristön tietolähdeliitäntöihin. Keskittämällä tietokantayhteydet Data Environmentiin voidaan kaikki projektin sisältämä informaatio pitää yhdessä. Se lisätään projektiin samalla tavalla kuin esimerkiksi lomake. Data Environmentissa määritellään tietokantayhteys minkä jälkeen RecordSet-objektit ovat käytettävissä ohjelman mistä tahansa osasta ja niitä voidaan sitoa myös muihin ohjaimiin. Data Environment on tiedosto, joka voidaan jakaa ja uudelleen käyttää myöhemmin muissa projekteissa. Yhteen Data Environmentiin on mahdollista liittää useita tietokantayhteyksiä ja yhteen projektiin voidaan lisätä useita Data Environment -ympäristöjä. (Rahmel 1999, 79)

Data Environment -tiedosto voi sisältää linkkejä ja yhteyksiä, Connection- ja Command -objekteja, ADO-tapahtumien koodeja ja Command-objektin ryhmittelyjä ja yhdistelyjä. (Rahmel 1999, 79)

Data Environment -ympäristön ominaisuuksiin liittyvät tiedot on esitelty Taulukossa 1.

Ominaisuus	**Tapahtuma**
Attributes	Tallentaa Connection-objektille välitettäviä ylimääräisiä asetusmerkkijonon ominaisuuksia
CommandTimeout	Sisältää sen ajan sekunteina, jonka palvelin odottaa käskyä palauttaakseen vastauksen
ConnectionSource	Tietolähdeyhteyden polun kuvaava merkkijono
ConnectionTimeout	Sisältää sen ajan sekunteina, jonka palvelin odottaa yhteyden avautumista
CursorLocation	Määrittelee yhteyden käyttämän kursorin sijainnin

Taulukko 1. *Data Environment -ympäristön ominaisuuksia* (Rahmel 1999, 78)

Data Environment -ikkunassa voidaan helposti

- lisätä uusia yhteyksiä ja käskyjä
- lisätä tallennettu proseduuri
- lisätä tai muokata ympäristön tapahtumakoodia
- asettaa ympäristön asetuksia. (Rahmel 1999, 78)

3.1.1 Connection

Data Environment -ympäristön keskeinen ominaisuus on Connection –objekti, joka sisältää todellisen yhteyden tietokantaan. Siinä määritellään kaikki parametrit kuten polkumääritys tai sijainti, joita käytetään tietokantayhteyden luomisessa. Tarvittaessa objektiin voidaan määritellä käyttäjätunnus ja salasana.

3.1.2 Command

Command-objekteja käytetään määrittämään tietokannasta haettavia tietoja kyselyä suoritettaessa. Objekti voi perustua tietokannan rakenteeseen kuten tauluun, näkymään tai SQL-kielellä tehtyyn kyselyyn. Lisäksi sitä voidaan käyttää taulujen välisten yhteyksien määrittelyyn. Command-objektin käyttäminen edellyttää sen liittämistä Connection-objektiin. (Rahmel 1999, 89)

3.1.3 Sidotun ohjaimen liittäminen

Erilaisten sidottujen ohjainten määrä on hyvin suuri. Niiden avulla voidaan toteuttaa helposti erilaisia lomakkeita. Sidottavan ohjaimen kaksi pääominaisuutta ovat DataSource ja DataMember. DataSource-ominaisuus osoittaa käytettävän Data Environment -ympäristön. DataMember-ominaisuutta käytetään valittaessa haluttu Command-objekti. (Rahmel 1999, 91)

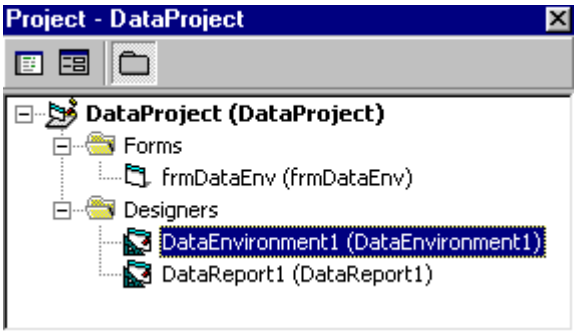
3.1.4 Data View -ikkuna

Data Environmentin lisäksi Visual Basicissa on toinen vastaava tietokantojen käsittelyyn liittyvä menetelmä. Se on nimeltään Data View -ikkuna, joka on globaali eli se saadaan näkyviin missä tahansa projektissa. Sitä voidaan kutsua todelliseksi liittymäksi yhteyksien hallinnasta puhuttaessa. Data View -ikkuna voi sisältää kaksi erillistä osaa, Data Environmentin ja Data Links -osan. Ikkunan avulla voidaan hyödyntää vedä-ja-pudota -toimintoa sijoitettaessa objekteja Data Environment -ympäristöön. Lisäksi se antaa mahdollisuuden luoda ja muokata tauluja, ohjelmoida tallennettuja proseduureja ja triggereitä sekä rakentaa tietokantakaavioita. Data View -ikkunasta päästään käsiksi Visual Database -työkaluihin, joihin kuuluu kaksi hyödyllistä apuohjelmaa Database Designer ja Query Designer. Database Designer näyttää visuaalisesti tietokannan taulut kenttineen sekä niiden väliset yhteydet, jolloin rakennetta voidaan muuttaa. Query Designerilla voidaan rakentaa ja suorittaa SQL-kyselyjä. (Rahmel 1999, 92 - 93)

DataLink-objektit ovat samanlaisia Connection-objektien kanssa sillä erotuksella, että ne ovat saatavilla missä tahansa Visual Basic -ympäristössä. Data Linkin luomisen jälkeen sitä voidaan käyttää sen osoittaman tietolähteen rakenteen tutkimiseen. Tämä johtuu siitä, että Data View -ikkuna näyttää kaikki käynnissä olevan VB 6 -ohjelman tietokantayhteydet. (Rahmel 1999, 92 - 93)

3.1.5 Tietokantaprojektin luominen

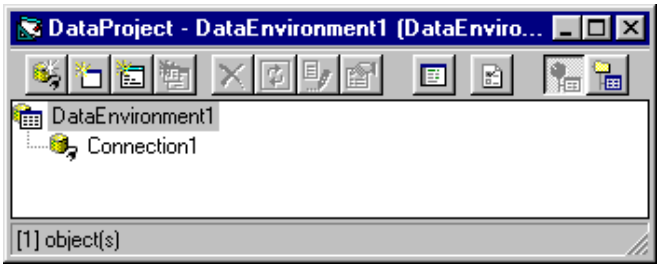
Tietokantaprojektin aloittamisen helpottamiseksi, varsinkin aloittelijoille, on Visual Basicissa malliprojekti, joka sisältää automaattisesti kaikki sovelluksen pääosat: Data Environment -ympäristö, yksi tyhjä lomake, kaikki käytettävissä olevat Data-sidonnaiset ohjaimet ja yksi raporttitiedosto. Malliprojekti on nimeltään Data Project ja se valitaan aloitusikkunasta uuden projektin aloittamisen yhteydessä (Kuva 1).



Kuva 1. *Data Project -malliprojekti*

Data Environment voidaan lisätä myös myöhemmin jo olemassa olevaan projektiin. Tämä voidaan tehdä normaalissa käyttöliittymäikkunassa tai Data View -ikkunassa.

Data Projectin luomisen jälkeen avataan ikkunaan Data Environment -kansio. Sen alapuolella nähdään Connection1 -kansio, joka kuvaa nimensä mukaisesti yhteyttä ja on lisätty siihen oletuksena (Kuva 2).

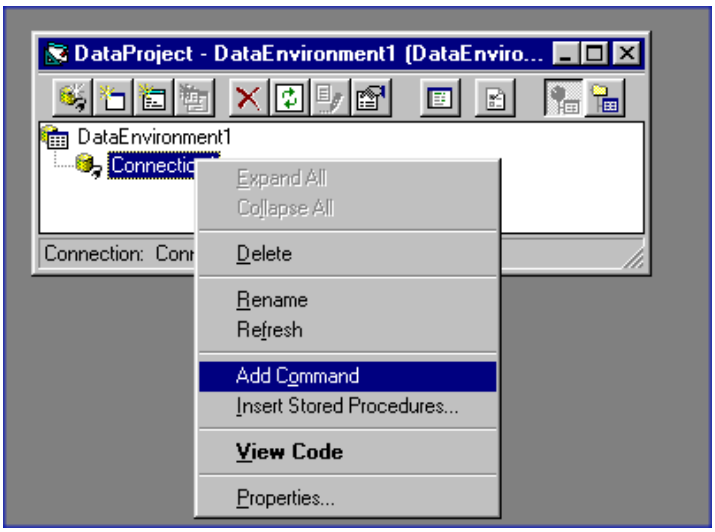


Kuva 2. *DataEnvironmentin oletuksena tuleva Connection1*

Connection1-objektille määritellään ominaisuuksiksi oikea tietokantatyyppi ja tietokannan sijainti. Valinta käynnistää velhon, jonka avulla ConnectionString -ominaisuuden arvot voidaan asettaa. Komento tuo näkyviin Data Link -ominaisuusikkunan, jossa objekti asetetaan kommunikoimaan minkä tahansa ActiveX-objektin sisältävän tietolähteen kanssa.

Ohjaimen tyyppi valitaan käytettävän tietokannan osoittamiseen. Microsoft Accessin tietokantamoottorin (Jet-moottori) valitseminen mahdollistaa yhteyden MDB (Microsoft Data Base) -muotoiseen tiedostoon. Tällöin valitaan Provider -välilehdeltä esimerkiksi Jet 3.51 OLE DB Provider. Yhteyksiä voidaan luoda useita kappaleita yhteen Data Environmentiin. Tietokannan tauluun päästään käsiksi Command-objektin avulla, jota voidaan sen määrittämisen jälkeen käyttää minkä tahansa sidotun ohjaimen lähteenä.

Command-objektista muodostuu lapsi Connection-objektille (Kuva 3). Ominaisuuksia muokataan osoittamaan pääyhteys oikeaan tietokantaan ja tauluun. Objektin ominaisuuksia määriteltäessä se nimetään ja asetetaan tyyppi. Tyyppivaihtoehtoina ovat kehittynyt tietokantarakenne, yksinkertainen viittaus tauluun tai määrätyn tiedon palauttava SQL-kysely. Tässä yhteydessä on mahdollista myös määrittää välitettäviä parametrejä, asettaa relaatioita, ryhmitellä tietoja ja määrittää taululle sen lukitustyyppi. Esimerkiksi taulun, jonka tietoja halutaan myös muokata pelkän lukemisen sijasta, lukitustyypiksi valitaan Optimistic.



Kuva 3. *Command-objektin lisääminen*

Valittaessa tyypiksi Table eli taulu ovat kaikki valitun taulun tietueet normaalisti käytettävissä. Kaikki taulun kentät saadaan näkyviin Projekti-ikkunassa tietokannan tauluun kytketyn Command-objektin avulla. Kenttien asetuksia voidaan tutkia ja niitä voidaan käyttää vedä-ja-pudota-toiminnolla lomakkeen tekemisessä. Data Environmentin käyttö, toisin kuin Data-ohjaimen, on riippumaton lomakkeesta. Tarvittavat kentät voidaan näyttää millä tahansa lomakkeella. Vetämällä kentät lomakkeelle ne saavat automaattisesti nimen ohjainten nimeämissopimuksen mukaan. Lisäksi toiminto tuo otsikot kentille valmiiksi.

Tietokannasta voidaan hakea haluttu, tietyt ehdot täyttävä, tietuejoukko esimerkiksi raporttiin. Valittaessa Commandin tyypiksi SQL-kysely, asetetaan ominaisuudet siten, että se suorittaa kirjoitetun SQL-käskyn. Kysely syötetään kirjoittamalla lauseke SQL Statement -ominaisuuden arvoksi tai tehdään kysely SQL Builderin avulla. Commandin luomisen jälkeen Projekti-ikkunassa nähdään kyseiset haetut kentät.

Data Environmentin luomisen jälkeen projektiin voidaan normaalisti lisätä ohjaimia. Sidottuja ohjaimia voidaan sitoa suoraan Data Environmentiin niiden DataSource-ominaisuuden avulla niin kuin edellä esiteltiin. Aikaisemmin käytössä ollut menetelmä Data-ohjain teki navigointipainikkeet automaattisesti. Nyt painiketoiminnot ohjelmoidaan itse RecordSet-objektin avulla.

Data Environmentia voidaan käyttää myös kehitettäessä Data-ohjaimien avulla toteutettua projektia.

3.2 Yleistä Data Reportista

Useimmissa tietokantasovelluksissa on tärkeää pystyä luomaan tulosteita. Aikaisemmin raporttien tekemiseen Visual Basicissa oli käytettävissä Crystal Report -ohjelmiston erikoisversio. Visual Basic 6.0 on ensimmäinen versio, johon on liitetty täydellinen raportointiympäristö Data Report -toiminto. Sen avulla voidaan luoda laajoja eriteltyjä raportteja tietokantayhteysmahdollisuuksia hyödyntäen ja se voidaan sitoa suoraan Data Environment -ympäristön objekteihin. (Rahmel 1999, 164)

Data Report tulostaa kyselyjen tuloksien perusteella raportin hierarkkisesti ammattimaiseen muotoon. Data Report -objekti tallennetaan lomakkeen tapaan tiedostoon. Projektissa tiedostot tunnistetaan dsr-lopputunnisteesta. Raportille voidaan asettaa tavalliseen tapaan yleisiä ominaisuuksia, kuten reunusten tyylit ja leveydet. Objektin teho ilmenee seuraavissa piirteissä:

- Graafinen raportin rakentaminen
- Automaattinen esikatselu ennen varsinaista tulostusta
- Liityntä VB-koodiin raportin tekovaiheessa
- Asynkroninen operointi taustaprosessin kanssa
- HTML-raportin luominen. (Rahmel 1999, 164)

Oletusarvoisesti raportti näytetään ensin esikatseluikkunassa, jossa sijaitsee raporttipainike tulostuksen lähettämiseksi kirjoittimelle. Tarvittaessa raportti voidaan tulostaa suoraan kirjoittimelle, tekstitiedostoon tai muotoiltuun HTML-tiedostoon. Tekstitiedostoon kirjoitettaessa on mahdollista valita joko ASCII- tai Unicode-merkistö. Tallennettaessa raportti HTML-muotoon, se voidaan helposti lähettää www-sivuksi tai sähköpostina, ladata käsimikroon tutkimista varten tai tallentaa se myöhempää käyttöä varten tai historiatietueeksi. Tiedoston nimeä annettaessa tarkentimeksi tulee ".htm" automaattisesti, jos sitä ei erikseen määritellä joksikin muuksi. Raportin kirjoittamista eri muotoihin

ohjataan ExportReport -ominaisuudella, jonka avulla annetaan raportille tyyppivakio (rptKeyHTML, rptKeyUnicodeHTML_UTBF, rptKeyText tai rptKeyUnicodeText) sen muodon generoimiseksi. (Rahmel 1999, 166 - 167)

2.2.1 Raportin luominen

Raportti rakennetaan joukosta kaistoja tai kappaleita. Kaistoja on neljää päätyyppiä: raportin ylä-/alatunniste, sivun ylä-/alatunniste, ryhmän ylä-/alatunniste ja yksityiskohta-kaista.

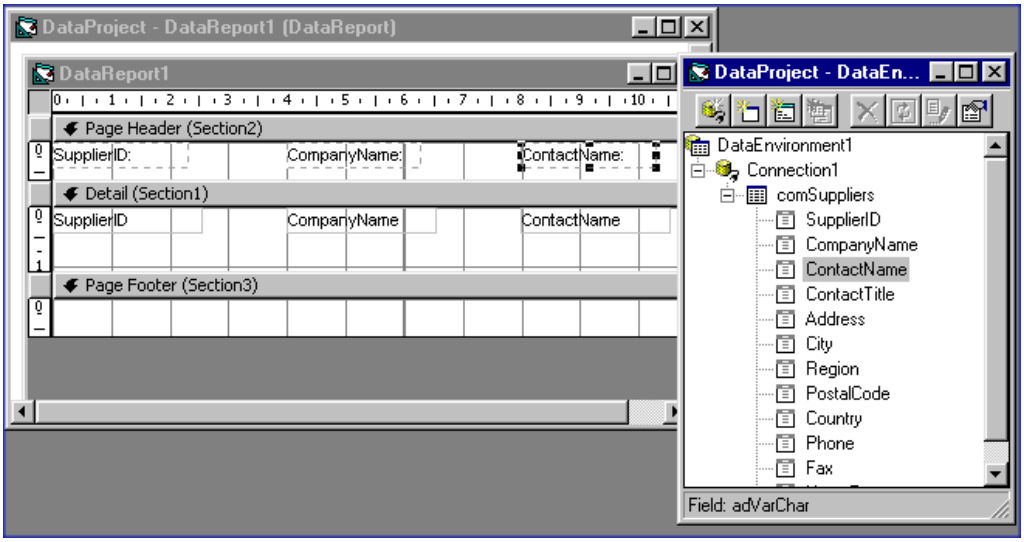
Raportin kaikki tiedot saadaan käyttöön Data Environment -ympäristöön luotujen yhteyksien kautta. Raportin luominen on yksinkertainen toiminto, mutta kannattanee ensin aloittaa yksinkertaisesta raportista. Sitä voidaan helposti laajentaa monimutkaisempien tietojen esittämiseen.

Seuraavassa esimerkissä on jälleen käytetty edellä esiteltyä Data Project -malliprojektia pohjana ja esitetyt tiedot ovat Microsoft Visual Studio 6.0 -paketin mukana tulleesta Northwind-esimerkkitietokannasta.

Jokainen raportti voidaan asettaa osoittamaan ensisijaiseen tietolähteeseen, esimerkiksi Commandiin. Tässä raportissa se kytketään Data Environment1 -ympäristön comSuppliers-objektiin, joka on luotu Commandiksi Suppliers-tauluun.

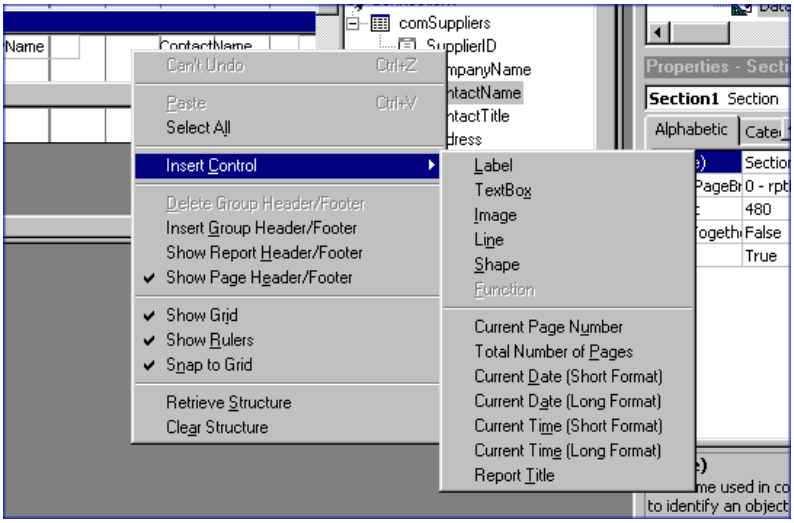
Yhteyksien luomisen jälkeen avataan Data Report1 -raporttipohja Projekti -ikkunan Designers-kansiosta. Annetaan raportin nimeksi rptSuppliers, jotta sen tunnistaminen ja kutsuminen koodissa on helpompaa. Valitaan raportin tietolähteeksi DataSource-ominaisuutta käyttäen DataEnvironment1 ja DataMember-ominaisuudeksi comSuppliers. Jotta voidaan olla varmoja raporttipohjan puhtaudesta, suoritetaan Retrieve Structure -toiminto, joka tyhjentää pohjan.

Raportin rakentaminen on helppoa vedä-ja-pudota-toiminnon ansiosta. Vedetään comSuppliers-commandin kenttiä raportin Details-kaistalle ja siirretään Label-kentät PageHeader-kaistalle otsikoiksi (Kuva 4).



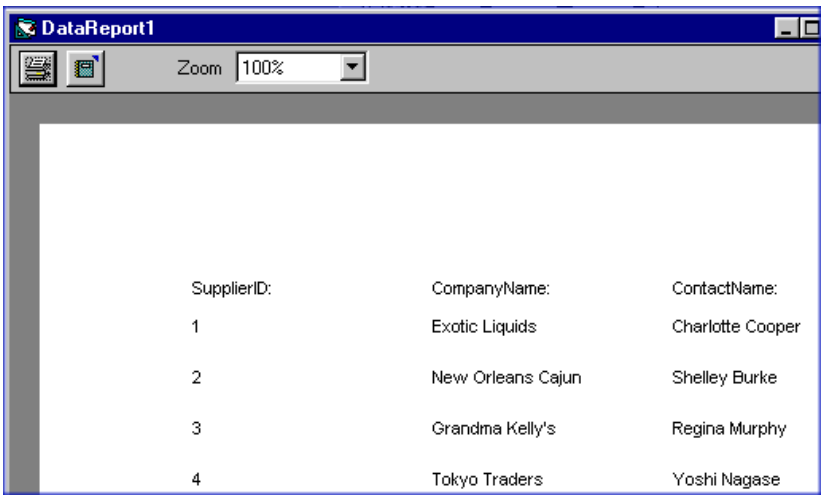
Kuva 4. *Yksinkertaisen raportin luominen*

Raporttiin voidaan lisätä useita erilaisia ohjaimia Insert Control -toiminnolla. Kuvassa 5 näkyvän valikon alaosan ohjaimet ovat automaattisia systeemin tietoja. Esimerkiksi sivunumeroiden raporttiin lisääminen tapahtuu Current Page Number -toiminnolla.



Kuva 5. *Controllin lisääminen raporttiin*

Raportin käynnistämistä varten luodaan lomakkeelle nappula, johon lisätään tarvittava koodi ’rptSuppliers.show’.

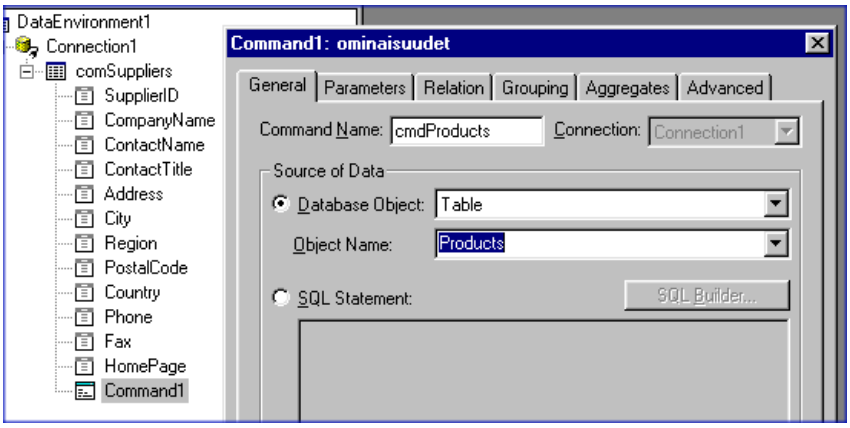


Kuva 6. *Esimerkkiraportti esikatselussa*

Sijoittamalla raporttiin muutamia asiaankuuluvia ylä- ja alatunnisteita, saadaan ammattimainen raportti asioiden esittämiseen. Yleensä raporteissa esitetään monimutkaisempia tietoja ryhmiteltynä, jotta niissä olisi enemmän tarpeellisia tietoja selkeästi lajiteltuina. Data Report sisältää mahdollisuuden myös sellaisen tekemiseen.

2.2.2 Raportin laajentaminen

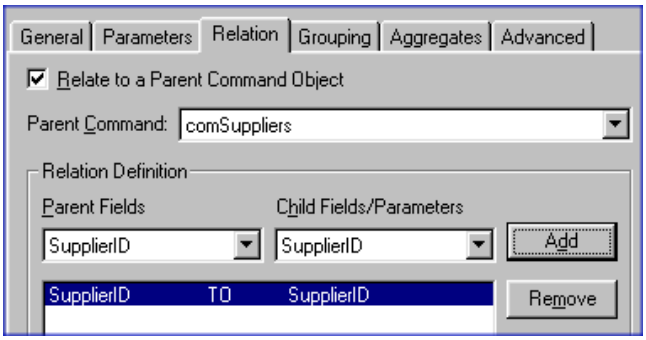
Lisäämällä comSuppliers-commandin alle toisen lapsiobjektin ja liittämällä taulut yhteen, saadaan laajempi hierarkkinen raportti yksityiskohtaisemman tason tiedon esittämiseksi.



Kuva 7. *Command1:n lisääminen lapsiobjektiksi*

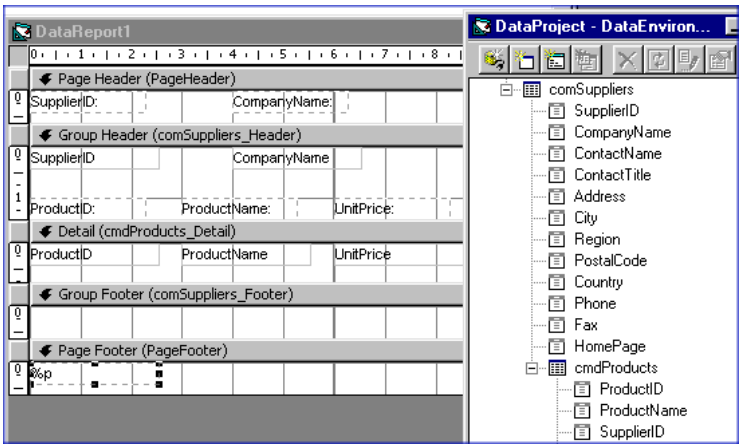
Lapsiobjektin lisääminen comSuppliers-objektille tehdään Add Child Command -komennolla, jolloin se lisätään objektin yhdeksi yksiköksi taulun kenttien lisäksi. Määritellään ominaisuudet, luodaan yhteys Products-tauluun ja annetaan nimeksi ’cmdProducts’ edellä esitellyllä tavalla (Kuva 7).

Ominaisuudet-ikkunan Relation-välilehdellä määritellään tauluille yhteys. Visual Basicin automaattitoiminto on valinnut Relate to a Parent Object -valinnan ja ehdottaa comSuppliers-taulua automaattisesti Parent Commandiksi eli isäntä -tauluksi. Ohjelma on hakenut SupplierID:n taulujen yhdistämiskentäksi.



Kuva 8. *Kahden taulun linkittäminen toisiinsa kentän avulla*

Ennen raportin uudelleenrakentamista kaikki aiemmin lisätyt kentät poistetaan kokonaan. Visual Basic muodostaa uuden tietorakenteen ja muokkaa raporttipohjaa sitä vastaavaksi lisäämällä siihen uudet kaistat Group Header ja Footer. Group Header -kaistaa käytetään comSuppliers-objektin (isäntä) kanssa näyttämään otsikkotiedot ja Detail-kaistaa comProducts-objektin (lapsi) kanssa näyttämään yksityiskohtaiset tiedot (Kuva 9). Tarvittavat kentät vedetään raporttipohjalle normaalisti.



Kuva 9. *Raportin luominen kahdesta taulusta*

Uudessa raportissa on nyt tietoa kahdelta eri tasolta. Ryhmittelytasoja voidaan luoda enemmänkin kuin kaksi lisäämällä Command-objekteja hierarkkisesti. Data Report ei kuitenkaan voi esittää kuin yhden lapsiobjektin tietoja kerrallaan, vaikka niitä olisi luotu useampia.

Kuva 10. *Esimerkkiraportti esikatselussa*

Valmiin raportin tulostaminen paperille on tavanomaisin käytäntö. Edellä mainitulla tavalla Visual Basic tarjoaa mahdollisuuden kirjoittaa raportti tekstitiedostoon tai HTLM-muotoiseen tiedostoon. Käyttämällä ExportReport -metodia tiedosto voidaan tallentaa levylle. Luodaan lomakkeelle painike, johon lisätään seuraava koodi:

RptSuppliers.ExportReport rptKeyHTML, ”(tallennuspolku)”, True, rptRangeAllPages

Näin raportti tallentuu annettuun hakemistoon rptSuppliers.htm -nimellä. Antamalla hakemistoksi palvelimen www-kansio saadaan tulos välittömästi tutkittavaksi web-selaimen avulla.

2.2.3 ToolBoxin ohjaimet

Data Report ei pysty käyttämään tavallisia ActiveX-komponentteja. Raportin ulkonäön muokkaamiseen ja tunnisteiden asettamiseen Data Reportin mukana asentuu muokattuja ActiveX-ohjaimia. Ohjaimia on kuusi ja ne löytyvät

Toolbox-ikkunasta Data Report -välilehdeltä. Ohjaimet ovat tuttuja vastaavista lomakkeiden OCX-ohjaimista, jotka ovat TextBox, Label, Image (voi sisältää kuvan, mutta sitä ei voida sitoa tietolähteeseen), Line, Shape ja Function (tekstikenttä jolla voidaan laskea arvo raportin luontihetkellä). Näitä kaikkia ohjaimia voidaan käyttää sijoittamalla niitä määrätyille kaistoille lopulliseen raporttiin. Näin saadaan näyttävä ja selkeä raportti halutun tiedon esittämiseen.

4 VUOKATIN HIIHTOKOULUN HISTORIAA JA TOIMINTAA

Vuokatin hiihtokoulun toiminta on perustunut koko sen olemassa olon ajan tiiviiseen yhteistyöhön Vuokatin hiihtokeskuksen kanssa ja hiihdonopetusta on luonnollisesti annettu yhtä kauan kuin siellä on laskettu mäkeä alas. Vuokatin Hiihtokoulu perustettiin vuonna 1987 itsenäiseksi, rekisteröimättömäksi yhdistykseksi. Hiihtokoulun perustamisen jälkeen hiihdonopetustoiminta jatkui kuitenkin edelleen hiihtokeskuksen alaisuudessa ja hiihdonopettajat olivat sen palkkalistoilla.

Vuonna 1992 pidettiin perustava kokous Vuokatin Hiihtokoulu ry:n perustamiseksi. Vaikka toimet yhdistyksen rekisteröimiseksi eivät toteutuneet, hiihtokoulun toiminta erotettiin virallisesti keskuksen toiminnasta. Yhdistykseen kuului sillä hetkellä 16 jäsentä. Samana vuonna Hiihtokoulu sai omat toimitilansa ja toimintaan liittyvä rahaliikenne kuten tuntien myynti ja suksinarikka siirtyi kokonaan hiihtokoululle. 1990-luvun puolivälissä koulu päätti kokeilla myös makkaranmyyntiä ja erilaisten after-ski -ohjelmien järjestämistä. Niistä kuitenkin luovuttiin muutaman vuoden kokeilun jälkeen kannattamattomuuden vuoksi ja päätettiin keskittyä itse opettamiseen sekä sen jatkuvaan kehittämiseen.

Hiihtokoulutoiminta Suomessa on viime vuosina muuttunut ammattimaisemmaksi. Yhä useammat hiihtokoulut toimivat yrityspohjalta ja niissä työskentelee päätoimisia opettajia. Vuokatin hiihtokoululla työskentelee talvikauden aikana kahdesta kolmeen päätoimista opettajaa ja muita opettajia päivystysperiaatteella, mikä tarkoittaa noin 15 työpäivää kauden aikana.

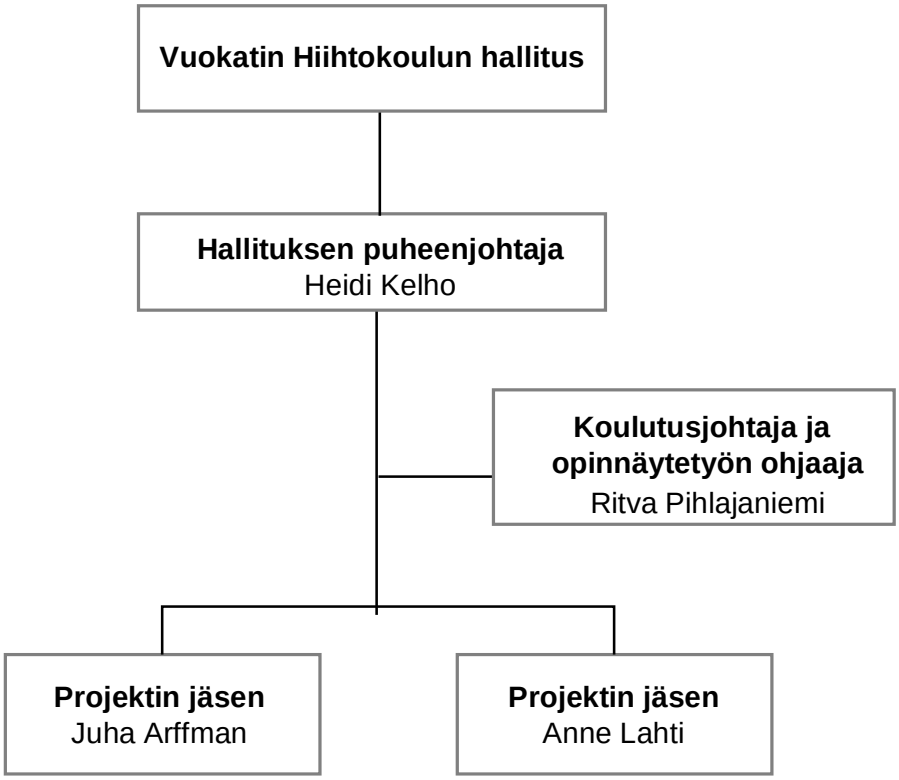
Nykyään hiihtokoulu toimii yhdistyksenä, jossa viisi henkinen hallitus valvoo ja ohjaa koulun toimintaa. Yhdistyksen jäseneksi voi päästä vain vähintään Alppihiihdonopettajakurssi I:sen suorittanut henkilö, joka taitotestien kautta otetaan ensin vuoden koeajalle. Mikäli hallitus katsoo kokelaan sopivaksi, se kutsuu henkilön hiihtokoulun varsinaiseksi jäseneksi. Tällä hetkellä jäseniä on noin 30.

Hiihtokoulun tulevaisuus näyttää hyvältä. Erityisesti laskettelun ja lumilautailun suosio kasvaa edelleen vuosi vuodelta ja yhä enenevissä määrin asiakkaina on myös ulkomaalaisia. Tästä johtuen potentiaalisten asiakkaiden määrä tullee lisääntymään. Viimeisen viiden vuoden aikana liiketoiminnan kasvu on ollut todella nopeaa. Päättyneellä hiihtokaudella 1999 - 2000 liikevaihto nousi 300 000 markkaan eli 40 prosenttia edellisestä kaudesta.

5 HIIVA-PROJEKTI

Toteutimme HiiVa-projektin Vuokatin Hiihtokoulun puheenjohtaja Heidi Kelhon toimeksiannosta (Liite 1). Toimeksiantoon kuului luettelo toivotuista ohjelma toiminnoista (Liite 2), jolla rajattiin projektin tuloksena syntyvän ohjelman laajuus. Projektin nimi HiiVa on johdettu ohjelman nimestä Vuokatin Hiihtokoulun varaustenhallinta- ja raportointiohjelma.

5.1 Organisaatio



Kuva 11. *HiiVa-projektin organisaatiokaavio*

5.2 Alkutilanne

Ennen HiiVa-projektin aloittamista Vuokatin Hiihtokoulussa kaikki varaustoimintaan liittyvät tehtävät tehtiin manuaalisesti ja tietokoneen käyttö keskittyi lähinnä erilaisten tekstien ja taulukoiden käyttöön. Talletetut tiedot olivat omina yksittäisinä asiakirjoinaan. Palkanmaksun perusteena käytettävät tuntilistat täytettiin käsin valmiille pohjille. Näiltä listoilta laskettiin myös kausikohtaiset yhteenvetotiedot. Raporttien laskennassa käytettiin apuvälineenä taulukkolaskentaohjelmaa.

5.3 Tavoite

Projektin tavoitteena oli luoda graafisella käyttöliittymällä varustettu ohjelma, jolla korvattaisiin hiihtokoulussa opetustuntien varausten seurannassa käytetty varauskirja sekä lisäksi palkanmaksussa ja opetustuntien seurannassa käytetyt opetuskuitit sekä kassakone. Tietokantaan tallennetut tiedot mahdollistaisivat myös erilaisten yhteenvetoraporttien tulostamisen. Tarkoituksena oli saada ohjelma, joka nopeuttaisi päivittäisiä ja kausitasolla tapahtuvia rutiinitehtäviä sekä tehdä hiihtokoulun varauksien tekemisen ja raportoinnin kannalta olennaiset toiminnot luotettavimmaksi.

Projekti rajattiin käsittämään vain itsenäinen eli stand-alone -ohjelma, koska ohjelman verkkokäytön mahdollisuus ei vielä ole ajankohtaista. Projektiin kuului käyttöliittymän-, ohjelman- ja tietokannan suunnittelu ja toteutus sekä käyttöohjeiden laadinta ja ohjelman käyttöönoton järjestäminen. Hiihtokoulun tarpeiden ja projektiin suunnitellun ajankäytön vuoksi kirjanpitotoimintojen ja reskontran teko eivät kuuluneet sen sisältöön.

5.4 Suunnittelu

Suunnittelun ensimmäisessä vaiheessa kartoitimme yleispiirteittäin tietokannalta vaadittavat ominaisuudet ja siihen talletettavat tiedot sekä ohjelmaan tulevat toiminnot toimeksiannon mukaiseksi. Toteutimme ohjelman vapaamuotoisesti siten, että ainoana vaadittuna kriteerinä oli tarvittavien raportointitarpeiden toteutuminen. Tämä johtui siitä, että Juha Arffmannilla oli vuosien kokemus hiihtokoulun toiminnasta ja päivittäisistä rutiineista. Hänellä oli kaikki tarvittava tuntemus ohjelmaan tallennettavien tietojen ja vaadittavien toimintojen osalta. Niiden pohjalta suunniteltiin sopiva tietokanta tauluineen ja ohjelman toimintojen kuvaukset (Liitteet 3,4 ja 5).

Seuraava vaihe työ aloittamiseksi oli löytää sopivat työkalut ohjelman toteuttamiseen. Visual Basic oli ohjelmointityökaluna molemmille tutuin, koska sitä oli opiskeltu koulussa eniten. Toinen esillä ollut vaihtoehto oli Delphi. Visual Basicin oliosuuntautuneisuus sekä uusimman version 6.0 myötä tulleet entistä paremmat ominaisuudet tietokantojen hyväksikäyttöön sekä mahdollisuus upotetun SQL:n käyttöön ratkaisivat asian Visual Basicin hyväksi.

Käytettävää tietokantaa mietittäessä tutkimme ensin eri vaihtoehtoja (MS Access, Paradox, dBase), niiden käyttämisen hyviä ja huonoja puolia sovellettuna Visual Basic -ohjelmointiympäristöön. Microsoft Access 7.0:n tietokanta oli kuitenkin vahvin ehdokas, koska kumpikin hallitsi parhaiten sen käytön. Tiesimme myös, että Visual Basic tukee hyvin MS Accessin tietokantaa. Asiaa vielä hiihtokoulun edustajalta tiedusteltaessa ilmeni, että hiihtokoulu ei ollut valmis investoimaan dynaamiseen, arvokkaaseen tietokantapalvelimeen. Asiasta keskusteltaessa ensimmäisessä alustavassa palaverissa ohjaavan opettajan Ritva Pihlajaniemen kanssa esille nousi seikka Accessin kykenemättömyydestä toimimaan reaaliaikaisesti useassa työasemassa. Hiihtokoulun arvioiden mukaan ohjelman yhtäaikainen käyttö usealta koneelta (Client-Server -ohjelma) ei kuitenkaan vielä olisi ajankohtaista, joten tämäkään seikka ei puoltanut jonkin toisen tietokannan valitsemista.

5.5 Ohjelman toteutus

HiiVa toteutettiin suunnitelman mukaisesti käyttäen MS Accessin tietokantaa Visual Basic 6.0 Enterprise -sovelluksen toimiessa ohjelmointiympäristönä. Tietokantayhteytenä HiiVa käyttää luotua tietokantaa (HiiVa.mdb) ODBC -rajapinnan (Open DataBase Connectivity) kautta. ODBC:n käyttöön syynä oli se, että tavoitteena oli saada HiiVasta mahdollisimman itsenäinen ohjelma. Tämä tarkoittaa sitä, että HiiVaa voidaan käyttää myös ilman MS Access -sovellusta eli se on asennettuna käytettävään työasemaan.

Yhteys luotuun ODBC-yhteyteen on määritelty Visual Basicin omassa tietokantaympäristössä Data Environmentissa, joka on esitelty julkaisun alkuosiossa. Tietojen muokkauksessa ja tallennuksessa käytetään pääasiassa upotettua SQL:ia ja tietokannan lukemisessa Data Environmentia sen hyödyllisine ominaisuuksineen.

5.5.1 Tietokanta

HiiVa-tietokanta muodostuu 14 taulusta (Liite 6), joista tärkein on Varaukset -taulu. Sen ympärille rakentuu koko varausjärjestelmä ja se onkin raportointitarpeita ajatellen tärkein taulu.

Varaukset-tauluun tallentuu kaikki oleelliset tiedot varauksista. Varatun tunnin tietoina tauluun tallennetaan varauksen numero (avainkenttä), päivämäärä, aika, tunnin pitävä opettaja, tunnin myynyt opettaja, oppilaan nimi, oppilasmäärä, tunnin tyyppitiedot ja hinta markkoina ja euroina sekä maksuun liittyvät tiedot. Huomio-kenttään voidaan kirjoittaa vapaamuotoista tekstiä esimerkiksi oppilaasta (lapsi, aikuinen, aloittelija). Varausjärjestelmän kannalta tärkeitä tallennettavia tietoja ovat kuittaaminen maksetuksi ja pidetyksi, koska näitä tietoja käytetään seurantaraporttien tulostusehdoissa. Avainkenttänä on Vnro ja indeksoituja kenttiä ovat ONro, VPvm, ValkuAika, TMNro ja TLNro. Tauluun on varattu mahdollista myöhempää käyttöä varten kentät oppilaan tasosta ja iästä sekä muutama varakenttä.

Ensimmäisessä taulumallissa kyseessä olevaan Varaus-tauluun tallennettiin opettajan nimi, tuntimuoto ja tuntilaji tekstimuodossa. Tietokannan normalisoinnin yhteydessä kantaan tehtiin kuitenkin hieman muutoksia. Silloin päädyttiin tallentamaan edellä mainitut tekstimuodossa olevat tiedot numerotietoina ja luotiin kolme uutta taulua, jotka linkitettiin Varaukset-tauluun.

TuntiMuoto-tauluun tallentuvat tiedot tuntimuodon järjestysnumerosta, nimestä (yksityis- tai ryhmätunti, kurssi), hinnasta markkoina ja euroina, arvonlisäveroprosentista, oletusarvo oppilasmäärästä ja opettajan saaman palkan osuus pidetyn tunnin hinnasta. Alv-prosentti tallennetaan käytettävien verokantojen mahdollisten muutoksien varalta. Avainkenttänä on TLNro ja TMNimi on indeksoitu.

TuntiLaji-tauluun tallennetaan tuntilajin järjestysnumero ja nimi (laskettelu, lumilauta, telemark). Avainkenttänä on TLNro ja TLNimi-kenttä on indeksoitu. MaksuTapa-tauluun liittyvät tiedot ovat käytettävän maksutavan järjestysnumero (avainkenttä) ja nimi. Maksutapa-taulu on myös linkitetty Varaukset-tauluun, johon tallentuu maksutavan numero.

Tuntimuodot, tuntilajit ja maksutavat ovat omissa tauluissaan tietokannan optimoinnin ja ylläpidollisten seikkojen vuoksi. Niitä tarvitaan myös Kassatilitys-raporttiin.

Opettajat-tauluun tallentuvat opettajien perustiedot. Näitä tietoja ovat henkilötunnus, osoite-, puhelinnumero- ja sähköpostitiedot sekä veroprosentti ja maksuyhteystiedot. Avainkenttänä muihin tauluihin toimii ONro-kenttä. Onimi-kenttä on indeksoitu.

OpettajienKurssit-tauluun tallennetaan opettajien käymät kurssit ja niiden suoritusvuodet. OKNro-kenttä toimii avainkenttänä Opettajat-tauluun. KrsNro -kenttä on indeksoitu. Kurssit-tauluun tallennetaan erilaiset Ahory:n (Alppihiihdon opettajat ry) tai muun vastaavan tahon järjestelmät kurssit. Kurssin järjestysnumero on avaimena OpettajienKurssit-tauluun. Paivystykset-tauluun tallennetaan opettajille merkityt päivystysvuorot. Tauluun tallennetaan opettajan numero, päivystyspäivämäärä ja tieto siitä onko kyseessä oleva päivystys

kuitattu pidetyksi. Avainkenttänä on laskurikenttä ja PPvm- ja PONro-kentät ovat indeksoituja.

KoulutusKassa-tauluun tallennetaan opettajakohtaisesti kymmenen markkaa jokaisesta opettajan pitämästä tunnista. Koulutuskassa on voimassa kolme vuotta ja se on käytettävä erilaisiin koulutusmenoihin päättymispäivään mennessä. Taulu on linkitetty KKONro-avainkentällä Opettajat-tauluun.

Kayttaja-taulussa ovat tallessa opettajien käyttäjätunnukset ja salasanat. Näitä tarvitaan käyttäjän tunnistamiseksi, jotta hän pääsee tarkastelemaan ja tarvittaessa muuttamaan koulutuskassaan, perusylläpitotoimintoihin ja käyttäjäoikeuksiin liittyviä tietoja. Avainkenttänä on KaytNro-kenttä ja Onro -kenttä on indeksoitu. Perustiedot-taulussa on tallennettuna perustietoja hiihtokoulusta. Niitä käytetään yhteystietoina eri raportteja tulostettaessa.

KassaTilitys-tauluun tallennetaan päiväkohtaiset kassatilitykset ja PankkiTilitys -tauluun tallennetaan tehdyt pankkitilitykset. Näistä molemmista saadaan tarvittaessa myös tulostettua seurantaraportit. Avainkenttänä KassaTilitys -taulussa on KTnro-kenttä ja KTPaiva-kenttä on indeksoitu. PankkiTiltys-taulussa PankKTNro-kenttä on avainkenttä ja PankTPvm-kenttä on indeksoitu. Mmyynti -tauluun tallennetaan muista toiminnoista kuin varauksista kertyneet tulot. Tällaisia voi olla esimerkiksi suksinarikkamaksut. Muusta myynnistä kirjataan tauluun päivämäärä, palvelun/tuotteen nimi ja summa markkoina ja euroina. Avainkenttänä on MMyyntiNro.

5.5.2 Käyttöliittymä

HiiVa-ohjelman käyttöliittymä koostuu lomakkeista ja raporteista. Lomakkeet on luotu Visual Basicin Form-komponentilla ja niihin on sijoitettu tarvittavia lisäkomponentteja tietojen esittämiseen. Raportit on luotu käyttämällä Data Report -toimintoa ja muokkaamalla niitä koodaamalla.

Aloituslomakkeessa (Kuva 12) on kuvattu talvinen maisema Vuokatin hiihtokoulun logolla varustettuna, ohjelman virallinen nimi ja version numero.

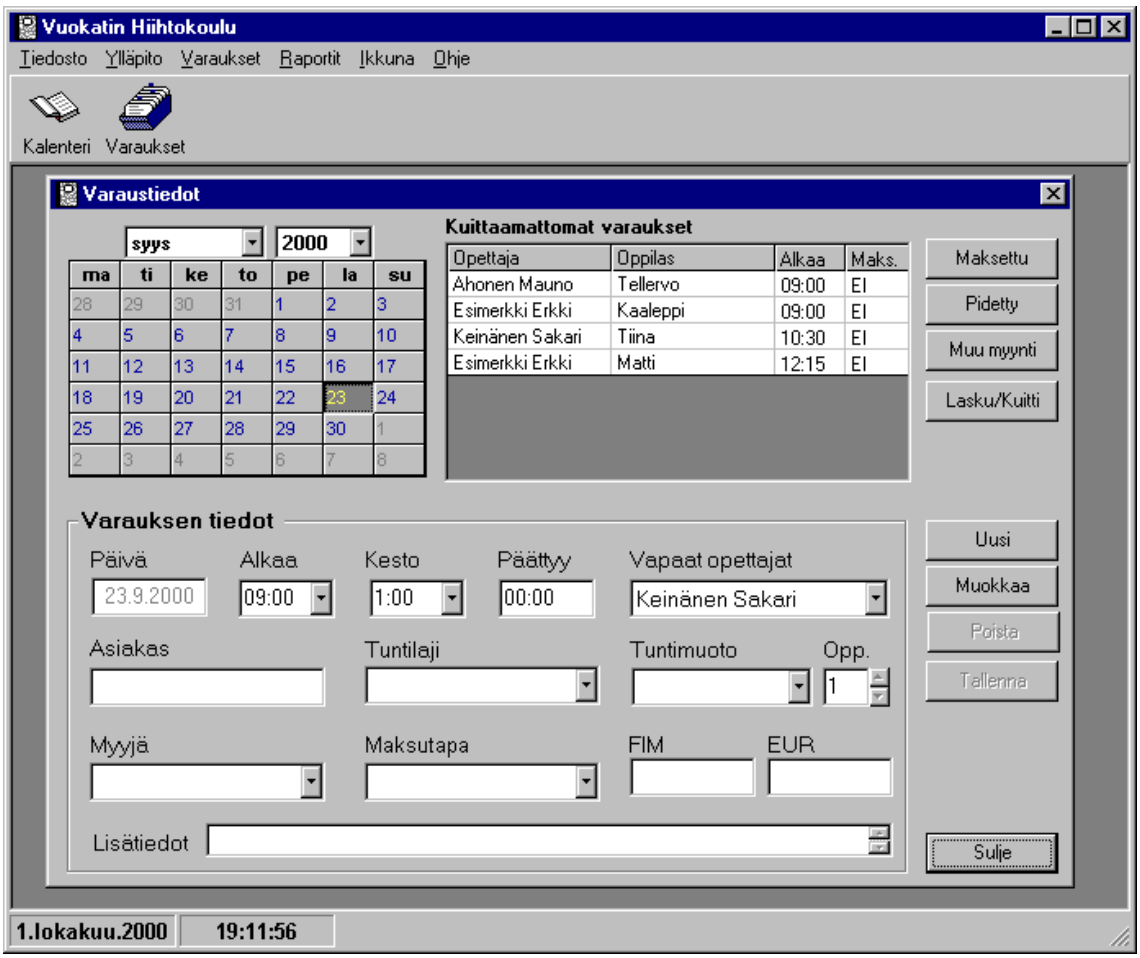
Kuva 12. *Ohjelman aloitusikkuna*

Käyttöliittymässä pohjalla on aina ohjelman MDI-pääikkuna, joka on aina näkyvissä vuorollaan aukeavien eri ikkunoiden (lapsi-ikkunoiden) alla. HiiVa -ohjelman tärkeimmät ikkunat on Varaustiedot- ja Varauskalenteri-ikkunat.

Varauksen tekeminen

Varaustiedot-ikkuna avataan Varaukset-valikosta. Varausten käsittelyssä voidaan tallentaa uusia ja muokata pidetyksi kuittaamattomia varauksia. Kuittaamattomat varaukset -luetteloon ovat listattuna kalenterista valitun päivän varaukset. Yksittäisen varauksen tietoja voidaan tarkastella lomakkeella valitsemalla luettelosta kyseinen haluttu varaus.

Varauksella voi olla kolme erilaista tilaa: varattu, maksettu tai pidetty. Varaustietoja on mahdollista muokata tai se voidaan kokonaan poistaa silloin, kun varauksen tilana on varattu tai maksettu. Varauksen pidetyksi kuittaamisen jälkeen se häviää luettelosta, eikä näin ollen sen tietoja voida enää muuttaa Varaustiedot-ikkunasta, vaan mahdolliset muutokset on tehtävä Perusylläpito -ikkunassa.



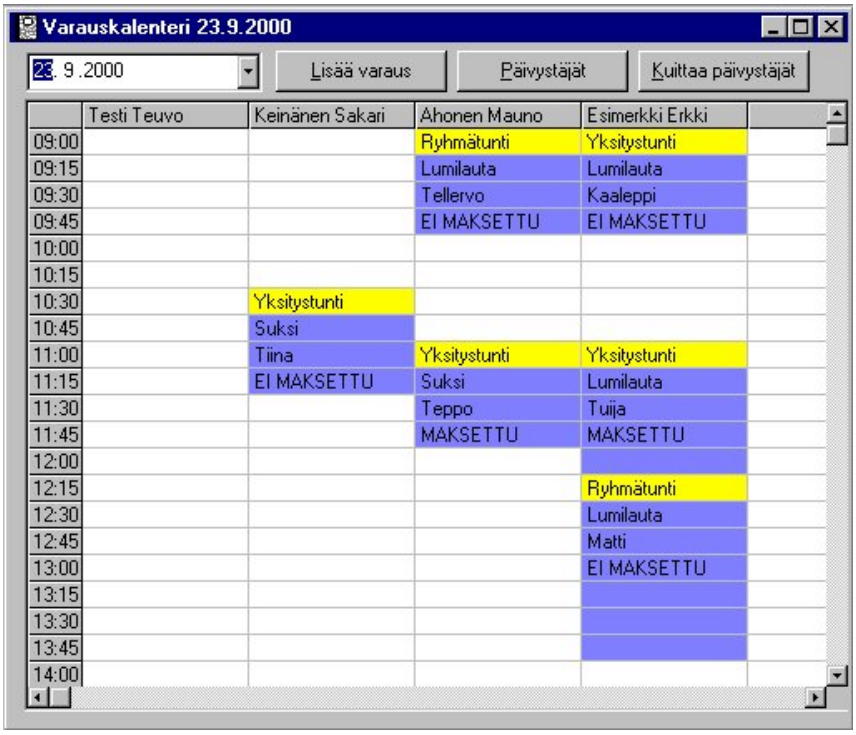
***Kuva 13.**. Varaustiedot-ikkuna avattuna MDI-pääikkunaan*

Uuden varauksen tallentaminen aloitetaan joko varauskalenterista tai Varaustiedot-ikkunasta Uusi-painikkeella. Lisätietoja lukuun ottamatta kaikki muut varauksen eritellyt tiedot ovat pakollisia. Vapaat opettajat -pudotuslistaan tulevat automaattisesti varausajan perusteella vapaana päivystysvuorossa olevat opettajat. Luettelosta valitun varauksen tietoja voidaan myös muuttaa tai se voidaan poistaa. Varauksen muokkaus tapahtuu valitsemalla Muokkaa-painike ja muuttamalla halutut tiedot Muokkaa varaustietoa -ikkunassa, jonka jälkeen muutokset tallennetaan normaalisti. Varauksen poistaminen tapahtuu Poista -painikkeella.

Asiakkaan maksettua varauksen se kuitataan maksetuksi. Tämän toiminnon suorittaminen on ehtona sille, että varaus voidaan kuitata pidetyksi. Pidetyksi kuittaamisen yhteydessä tunnin pitäneen opettajan koulutuskassaa päivitetään automaattisesti. Muu myynti -toiminnon avulla voidaan järjestelmään syöttää myös muita tuloja kuin varauksista saatavia. Tällainen tulo voi olla esimerkiksi suksinarikasta perittävä maksu.

Lasku/kuitti-painikkeen avulla tulostetaan asiakkaalle käteiskuitti tai lasku. Mikäli valittu varaus ei ole maksettu-tilassa, käyttäjää muistutetaan siitä ja ohjelma antaa mahdollisuuden tehdä kuittauksen tulostuksen yhteydessä.

Varauskalenteri-ikkuna näyttää kalenterista valitun päivän varaukset graafisesti aloitusajan ja keston mukaisesti. Kalenterin ajat esitetään 15 minuutin tarkkuudella. Varauksen tiedoista näytetään tuntilaji- ja muoto, asiakkaan nimi ja maksun tila. Varauskalenterissa voidaan aloittaa uuden varauksen lisääminen maalaamalla halutun opettajan sarakkeesta varauksen aikaväli ja valitsemalla Lisää varaus -painike. Ikkunasta voidaan myös avata päivystäjien ylläpitoikkuna sekä kuitata opettajien päivystykset pidetyksi.



Kuva 14. *Varauskalenterin ikkuna*

Perustietojen ylläpito

Tiedosto-valikko sisältää ohjelman lopetustoiminnon. Ylläpito-valikon kautta päästään ohjelman käyttöön liittyviin ylläpitotietoihin, joita ovat ohjelman- ja opettajien perustiedot, päivystykset, koulutuskassa ja ohjelman käyttäjien tiedot. Valikon kautta päästään myös suoraan muokkaamaan käytettävässä tietokannassa olevia varaus- ja päivystystietoja. Tämän toiminnon avulla on mahdollista esimerkiksi palauttaa virheellisesti maksetuksi kuitattu varaus maksamattomaksi.

Perustiedot-ikkunassa ylläpidetään tuntimuotojen, tuntilajien, maksutapojen ja kurssien luetteloita, joissa olevia tietoja käytetään ohjelman muissa toiminnoissa. Luettelot on eritelty kahdelle välilehdelle selkeyttämään käyttöä. Tuntimuodot ovat omalla välilehdellään, jossa niitä voidaan lisätä, muokata ja poistaa. Poistamisen yhteydessä ohjelma varmistaa, ettei kyseessä olevaa tuntimuotoa ole käytetty tallennetuissa varauksissa ja tarvittaessa estää sen poistamisen. Seuraavalla välilehdellä ylläpidetään tuntilajien, maksutapojen ja kurssien tietoja. Niille on luotu vastaavat toiminnot kuin tuntimuodoille varmistuksineen. Yhteystiedot-välilehdellä täytetään eri tulosteilla käytettävät yhteystiedot. Muutokset näihin tietoihin tallentuvat tietokantaan automaattisesti.

Opettajien tiedot -ikkunassa hoidetaan heidän yhteystietojen ja kurssisuoritusten ylläpito. Opettajat-luettelosta valitsemalla tietoja voidaan selata ja poistaa. Toiminnot uuden opettajan lisäämiseen ja uuden kurssin lisäämiseen opettajalle löytyvät myös tästä ikkunasta.

Päivystykset-ikkunaan on sijoitettu opettajille merkittyjen päivystysvuorojen ylläpito. Tarkasteltava päivä valitaan kalenterista, jolloin Päivystäjät-luettelo päivittyy kyseisen päivän mukaiseksi. Vapaat opettajat -luettelossa näkyvät vastaavasti sellaiset opettajat, joille päivystysvuoroja voidaan lisätä. Valitun kalenterikuukauden kaikki päivystykset on nähtävissä Tallennetut päivät -luettelossa.

Koulutuskassa-ikkunan kautta hoidetaan opettajakohtaista koulutusrahastoa. Rahasto karttuu pidettyjen tuntien perusteella. Tietoja päivitetään rahaston

käytön mukaisesti. Opettajanumeron ja kassan aloituspäivämäärän tallennus tietokantaan tapahtuu automaattisesti, kun hänen ensimmäinen opetustuntinsa on kuitattu pidetyksi. Ohjelma laskee kassalle loppupäivämäärän kolmen vuoden päähän.

Käyttäjät-ikkunan toiminnoilla voidaan lisätä, muokata tai poistaa ohjelman käyttäjiä. Ohjelman peruskäyttö ei vaadi erityisiä käyttäjätunnuksia tai kirjautumiskäytäntöjä. Kuitenkin muutamat toiminnot edellyttävät käyttäjätunnuksen ja salasanan käyttöä tietokannan turvaamisen vuoksi. Tällä on haluttu rajata koulutuskassaan, kassatilitykseen, käyttäjiin ja perusylläpitotietoihin liittyvät toiminnot pois peruskäyttäjiltä.

Kassatilitys-ikkunassa tallennetaan päivän kassatilitys. Muokkaa toiminnon avulla päästään tutkimaan ja muokkaamaan aiemmin tallennettuja kassatilityksiä. Pankkitilitys-ikkunassa tallennetaan noin kerran viikossa tehtävät pankkitilitykset.

Raportit

Seurantaraportit ovat hiihtokoulun toiminnan ja palkanmaksun apuvälineitä. Raporteille tulostuvat vain niiden varausten tiedot, jotka on kuitattu maksetuiksi ja pidetyiksi. Raportteja on mahdollista rajata päivämäärävälillä (alku- ja loppupäivämäärä) ja opettaja tiedoilla. Tuntilaji-raportti lajittelee ja laskee yhteen varaukset opettajittain sekä tuntilajeittain. Tuntimuoto-raportille tulostuvat samat edellä mainitut tiedot tuntimuodoittain lajiteltuna. Raportti nimeltään 'Lajittelu yhteensä' on hierarkkinen kolmessa tasossa (opettaja, tuntimuoto ja tuntilaji). Päiväraportille lasketaan valitun päivän myynti lajiteltuna maksutavoittain markkoina ja euroina. Päivystyslistaan tulostuvat opettajittain lajiteltuna päivystysvuoropäivät ja tiedot siitä, ovatko päivystykset suoritettu. Pankkitilitys- ja kassatilitysraporteille tulostuvat tallennetut päiväkohtaiset tiedot.

5.6 Testaus

Koko projektin ajan ohjelman kehittämisessä ja lopullisen version toteuttamisessa käytetty ohjelmointitapa oli protoileva. Protoilevassa tyylissä kehitetään yritysten ja erehdysten kautta useita eri versioita, joita kokeillaan ja parannetaan niin kauan että toimiva versio on saatu kehitetyksi. Tämä tarkoittaa sitä, että ohjelmaa testattiin koko ajan koodauksen yhteydessä.

Projektin ulkopuolisina testaajina toimivat Jouni Ahola (Project Manager, TietoEnator Oyj), Timo Korhonen (MediaKeel), Timo Komulainen (toimitusjohtaja, Tietohippu Oy) ja Kai Parviainen (Vuokatin hiihtokoulu). Testihenkilöt kirjasivat kommenttejaan ja parannusehdotuksiaan muistiin, joiden perusteella ohjelmaa korjailtiin ja lisättiin siihen uusia toimintoja.

5.7 Asennus

HiiVa-ohjelman asennusta varten laadittiin erillinen asennusohje (Liite 7). Laitteistovaatimukset ohjelman käyttämiseen ovat vähintään

- Käyttöjärjestelmä: Windows ’95
- Prosessori: Intel Pentium 200 MHz
- Keskusmuistia 16 Mb, suositeltavaa 64 Mb
- Kovalevytilaa 20 Mb
- 15” näyttö (käytettävä resoluutio 800x600).

5.8 Käyttöönotto

HiiVa-ohjelman käyttöoikeus luovutetaan Vuokatin hiihtokoululle projektin päättymispäivänä 9. lokakuuta 2000. Hiihtokoulu ottaa sen käyttöönsä kauden 2000 - 2001 aikana. Käyttöönottoa hidastaa toistaiseksi hiihtokoulun oman tietokoneen puuttuminen. Pienimuotoinen käyttöönottokoulutus ja ohjelman esittely on suunniteltu pidettäväksi joulukuun alussa hiihtokoulun opettajien lumileirin aikana, jolloin suurin osa opettajista on yhtä aikaa läsnä.

6 POHDINTAA

Tässä osiossa pohdimme toteuttamamme ohjelmistoprojektin perusteella Visual Basicia ohjelmointiympäristönä, omia kokemuksiamme projektin toteuttamisesta ja toteutumisesta sekä HiiVan tulevaisuuden näkymiä.

6.1 Visual Basic työvälineenä

Visual Basic on helposti hahmotettavissa oleva ja looginen ohjelmointiympäristö. Se on hyvä työkalu ensimmäisen todellisen ohjelman tekemiseen. Selkeän graafisen käyttöliittymän ansiosta Visual Basicia on myös varsin helppo käyttää. Visual Basic perustuu valmiisiin ohjelmoitaviin komponentteihin, joita sijoitellaan tarpeen mukaan lomakkeille. Ohjelmarungon tekeminen on huomattavasti nopeampaa, koska käyttäjä saa valmiiksi ohjelmapohjan, johon ohjelmoijan tarvitsee vain lisätä halutut komponentit ja tehdä niihin toiminnot. Visual Basicin syntaksintarkistustoiminto edesauttaa koodaamista ja virheiden minimoimista koodirivien suhteen.

Toinen Visual Basicilla koodaamista nopeuttava ominaisuus on se, että se ehdottaa automaattisesti käytettäville objekteille joitakin niihin liittyviä ominaisuuksia ja metodeja, kuitenkaan ei aina kaikkia mahdollisia. Sen vuoksi joitakin hyvin toimivia valmiita ominaisuuksia tai metodeja voi valitettavasti jäädä käyttämättä, koska ohjelmoija ei ole tietoinen niiden olemassa olosta. Tällöin ohjelmoijan itse kehittelemät pitkät monimutkaiset koodaustavat hidastavat ohjelman toimintaa, eikä koodi ole optimaalista. Visual Studio sisältää kuitenkin laajan On line -helpin MSDN Library:n, josta uusia metodeja ja niihin liittyviä

tietoja voi etsiä. MSDN on suhteellisen helppotajuinen ja kun sitä oppii käyttämään, se voi tuoda avun moneen ongelmaan. Internetistä löytyy runsaasti hyödyllistä materiaalia ja ohjelmointivinkkejä, koodin analysointiohjelmia sekä erilaisia uutis- ja keskusteluryhmiä. Visual Basicin pitkän historian vuoksi julkaisuja ja kirjoja on paljon ja niitä on hyvin saatavilla.

HiiVassakin käytetty, tietokantayhteyksiä varten kehitetty uusi työkalu Data Environment antaa yksinkertaisen ja vaivattomasti hallittavan rajapinnan tietokantaan tai tietokantoihin, jotka ovat käytössä koko ohjelmassa. Toinen uutena työkaluna versioon 6.0 tullut Data Report -raporttityökalu on vielä varsin kehittymätön ja muita vastaavaan tarkoitukseen tehtyjä ohjelmia, kuten Crystal Reports, jäljessä. Raporttien luominen Data Report -työkalua käyttämällä aiheutti suuressa määrin ongelmia. Laajempaan ja monihierarkkisempaan raportointiin sen avulla ei vielä ole saatavilla materiaalia.

HiiVan toimimisen kannalta tarvittavat lisätiedostot (esimerkiksi OCX:t ja DLL:t) ja niiden asentaminen osoittautuivat ongelmallisiksi. Ratkaisua siihen, miten kaikki ohjelman vaatimat tiedostot voidaan asentaa yhteen ohjelmakansioon, ei löytynyt.

Visual Basicin suosio ohjelmointiympäristönä ja suuren, yli miljoonan käyttäjän lukumäärän perusteella voidaan olettaa, että se ei tule häviämään markkinoilta. Edellä mainitut asiat lienevät myös syitä siihen, että uusimmat Windows -maailman teknologiat ovat olennaisena osana tätä ympäristöä. Näin ollen HiiVan jatkokehitys samalla työkalulla on turvattu.

6.2 Projektista ja omasta työstä

Projekti kokonaisuutena oli mielenkiintoinen ja opettavainen. Alkuperäisen projektin aikataulu perustui tavoitteeseen, että valmistuisimme jo keväällä 2000. Tiesimme aikataulun olevan tiivis, mutta silti päätimme yrittää, koska vielä tammikuussa opinnot näyttivät jakaantuvan tasaisesti koko kevään ajalle. Emme myöskään osanneet arvioida oikein opiskelujemme ohella tekemiemme ansiotöiden määrää. Koulu- ja työkiireiltämme emme edenneet suunnitellussa

aikataulussa ja huhtikuussa päätimme muuttaa projektin aikataulua siten, että projekti saataisiin päätökseen jouluun 2000 mennessä. Tämä muutos ei ohjelman suunnitellun käyttöönottoajankohdan vuoksi vaikuttanut haitallisesti hiihtokoulun aikatauluun.

Projekti eteni parityönä kesäkuun 2000 loppuun saakka, jonka jälkeen teimme työnjaon kummankin muuttaessa eri paikkakunnille. Työnjako suunniteltiin siten, että Anne aloitti kirjallisen julkaisun suunnittelun sekä toteutuksen ja Juhalle siirrettiin päävastuu ohjelman jatkokehityksestä. Käytännön syistä tämä vaikeutti ja hidasti projektin etenemistä. Yhteydenpito kesäkuun jälkeen tapahtui pääasiassa sähköpostilla ja puhelimella.

Ohjelman suunnitteluun olisi pitänyt käyttää enemmän aikaa ja voimavaroja. Vaikeutena suunnittelussa oli se, että emme olleet perillä Visual Basicin kaikista käyttökelpoisista komponenteista. Ehdimme kokeilla joidenkin meille uusien komponenttien toimivuutta niin pitkälle, että kyseessä olevaan tilanteeseen paremmin soveltuvan komponentin löytyessä emme kuitenkaan halunneet aloittaa koko suunnittelua uudelleen alusta asti. Pidemmälle viedyn suunnittelun tarve ilmeni myös muutamien toimintojen kohdalla, joissa olisi ollut järkevämpää käyttää aliohjelmia. Nyt ohjelmassa on samanlaisia toimintoja useammassa kohdassa, mikä ohjelmointiteknisesti ajatellen ei ole suotavaa.

6.3 Tulevaisuus

Seuraava itsestään selvä askel HiiVa-projektin jatkokehitystä ajatellen olisi verkkoversio eli Client-Server -sovellus. Sen tavoitteena olisi saada hiihtokoulun varausten tilanne näkymään reaaliaikaisena myös Vuokatin rinteiden lipunmyynnissä. Tähän tarvittaisiin lähiverkko, tietokantapalvelin ja perusteellinen ohjelman läpikäyminen sekä ohjelman osittain uudelleen koodaaminen. Toisena vaihtoehtona verkkokäytölle olisi tehdä intranettiin web-liittymä. Verkkoversiossa voitaisiin edelleen käyttää ohjelmointiympäristönä Visual Basicia. Tietokanta täytyisi vaihtaa dynaamisemmaksi ja sellaiseksi, jossa useampi kuin yksi käyttäjä voisi olla yhtä aikaa yhteydessä siihen reaaliaikaisesti.

Yleensä sovelluksiin sisältyy reskontra olennaisena osana laskutusta. Sen suunnittelu ja toteuttaminen olisi yksi tärkeä osa-alue ohjelman laajennustoimista. Ohjelmasta saatavien raporttien määrää pitäisi lisätä ja mahdollistaa käyttäjän luoda omia haluamiaan raportteja tarvitsemistaan tiedoista. Tällä hetkellä ohjelman valuuttoina käytetään rinnakkain markkoja ja euroja. Tulevaisuudessa markkojen käytön voinee poistaa. Lisää ohjelman käyttämiseen liittyviä toimintaa helpottavia ja nopeuttavia toimintoja (helppokäyttötoimintoja) voisi toteuttaa muun muassa Kalenteri-ikkunassa tekemällä vedä-ja-pudota-toiminnon varausten muokkaamiseen ja siirtämiseen. Pikanäppäinten ja näppäinyhdistelmien vähyyden vuoksi niiden lisääminen parantaisi näppäimistöltä käytettävyyttä, koska tällä hetkellä tilanne kyseisen asian osalta on alkutekijöissään. Lisää jatkokehitystarpeita tullee, kun ohjelma saadaan tuotantokäyttöön. Ohjelman jatkokehittäjää ei tällä hetkellä vielä ole. Projektin jatkaminen voisi olla hyvä opinnäytetyön aihe jollekin aiheesta kiinnostuneelle opiskelijalle.

Tietokannan varmuuskopiointi ja uuden tyhjän tietokannan vaihtaminen kauden alussa on sovittu hiihtokoulun omalle vastuulle. Toimintoja tullaan helpottamaan kuitenkin vielä siten, että hiihtokoululle suunnitellaan ja toimitetaan komentojonotiedosto ”kausivaihto.bat”, joka pakkaa kaudelta tulleen tietokannan levykkeille ja vaihtaa uuden tietokannan ohjelman pohjalle.

Ohjelma voisi olla mahdollista saada kaupalliseen levitykseen edellä mainittujen jatkokehitystoimintojen jälkeen. Se voisi soveltua muillekin hiihtokouluille tai vastaavaa toimintaa harrastaville yhteisöille.

KÄSITTEET:

ADO ActiveX Data Objects on Microsoftin objektimalli, jossa ohjelmoitavat objektit edustavat kaikkia paikallisia ja etätietolähteitä, mitkä ovat järjestelmän saavutettavissa.

Command Sisältää sen käskyn, joka suoritetaan tietolähteessä kuten tietokannan taulussa. Se on useimmiten SQL-käsky tai talletettu proseduuri.

Connection Sisältää tosiasiallisen yhteyden tietolähteeseen. Se sisältää polun, salasanan ja muut yhteysvalinnat.

DAO Data Access Objects on Microsoftin objektimalli, joka oli ensimmäinen objektipohjainen rajapinta, jolla ohjelmoijat pystyivät muokkaamaan Microsoft Jet -tietokantamoottoria.

Database Designer

Data Environment Designerin avulla voidaan luoda itse objektikomentoja uusilla dynaamisilla tavoilla.

Data Environment

Data Environment yksinkertaistaa huomattavasti liittymän luomista ja tarjoaa selkeän ohjelmointiympäristön tietolähdeliityntöihin. Sillä voidaan pitää projektin informaatio yhdessä helposti. Data

Environment -tiedosto voi sisältää linkkejä ja yhteyksiä, Connection- ja Command-objekteja, ADO-tapahtumien koodeja ja Command -objektin ryhmittelyjä ja yhdistelyjä.

Data Environment Designer

Data Environmentiin kuuluva työkalu, jonka avulla voidaan luoda objektikomentoja, joilla voidaan muokata ja järjestää uudelleen tietokantaa.

DataLink DataLink luodaan DataView -ikkunassa ja se on käytettävissä missä tahansa Visual Basic -ympäristössä. Sen luomisen jälkeen sitä voidaan käyttää sen osoittaman tietolähteen rakenteen tutkimiseen.

Data Report

Raporttien tekemiseen luotu objekti, joka tulostaa raportin kyselyjen tuloksien perusteella ammattimaiseen muotoon. Se tallennetaan lomakkeen tapaan tiedostoon dsr-tunnisteella.

Data View Tietokantojen käsittelyyn liittyvä menetelmä, joka saadaan näkyviin missä tahansa projektissa. Se näyttää kaikki käynnissä olevan Visual Basic 6-ohjelman tietokantayhteydet.

Jet-tietokantamoottori

Microsoft Jet Engine on teknologia, jota käytetään Microsoft Access -tauluissa ja muissa tiedon lähteissä sijaitsevan tiedon saamiseksi.

ODBC Open DataBase Connectivity on teollisuusstandardi, joka tarjoaa yleisen liittymän tietokantoihin. ODBC-ohjaimia on saatavilla lähes kaikkiin tietokantajärjestelmiin (kuten Oracle, Sybase, Informix, SQL Server ja niin edelleen).

OLE Object Linking and Embedding -tekniikka on oliotekniikka, jota käyttäen ohjelma saadaan toimimaan toisen ohjelman osana samassa sovelluksessa.

Proseduuri Tallennettu proseduuri on ohjelma, joka sijaitsee itse tietokannassa ja voidaan suorittaa kuten pieni ohjelma. Niitä käytetään enimmäkseen isoissa tietokantajärjestelmissä kuten Oracle.

Query Designer

SQL-kyselyjä suorittava objekti, joka yksinkertaistaa kyselyjen luomista ja optimointia. Sen avulla voidaan suorittaa ja muokata esimerkkikyselyjä ennen niiden koodiin asettamista.

RDO Remote Data Objects on Microsoftin objektimalli, joka on objektipohjainen rajapinta ODBC-lähteisiin.

Recordset Sisältää ne tietueet, jotka vastaavat kyselyyn asetettuja parametrejä ja kursorin tietueiden selaamiseen.

Trigger Trigger huolehtii tietokannan eheydestä. Esimerkiksi haluttaessa poistaa tietokannasta asiakas, tietokantaan voidaan luoda trigger, joka huolehtii kaikkien asiakasta koskevien tietojen poistamisesta.

LÄHTEET

Microsoft Press. 1998. Microsoft Visual Basic 6.0 Language Reference. Washington

Microsoft Press. 1998. Microsoft Visual Basic 6.0 Controls Reference. Washington

Microsoft Press. 1998. Microsoft Visual Basic 6.0 Component Tools Guide. Washington

Visual Basic 6 Complete. 1999. Sybex inc.

Halvorson, M. 1998. Microsoft Visual Basic 6.0 Professional Step by Step. Suom. A. Kuvaja. Helsinki: IT Press.

Rahmel, D. 1999. Visual Basic 6 Tietokantaohjelmointi. Suom. J. Arola. Helsinki: IT Press.

Microsoft MSDN Library Visual Studio 6.0 CD-Rom 1 ja 2. 1992 - 1998. Microsoft Corporation. Laitteistosuositus: PC/Windows 95 tai uudempi tai Windows NT 4.0 tai uudempi, 486-prosessori, 16 MB keskusmuistia (Windows 95) tai 24 MB (Windows NT 4.0).

Internet lähteet:

Project Analyzer Visual Basicille.
http://aivosto.com/project/suomi.html (Luettu 10.5.2000)

http://msdn.microsoft.com (Luettu 14.2 - 5.10.2000)

http://support.microsoft.com/support/news (Luettu 14.2 - 5.10.2000)

http://www.vbworld.com (Luettu 23.4.2000)

http://www.visualbasic.com (Luettu 14.2 - 5.10.2000)

LIITELUETTELO

LIITE 1	Toimeksianto
LIITE 2	Luettelo sovelluksen toiminnoista (Toimeksiannon liite)
LIITE 3	HiiVan toimintakaavio
LIITE 4	Ohjelman kuvauskaavio
LIITE 5	Varauskalenterin kuvauskaavio
LIITE 6	HiiVan kantayhteydet
LIITE 7	Vuokatin Hiihtokoulun Varaustenhallinta- ja raportointiohjelma (Hiiva): asennusohje
LIITE 8	Käyttöoikeuden luovutussopimus
LIITE 9	HiiVa-projektin ajankäyttökaavio

Vuokatin Hiihtokoulu
Vuokatinrinteet
88610 VUOKATTI

TOIMEKSIANTO

4.1.2000

VUOKATIN HIIHTOKOULUN VARAUSTENHALLINTA- JA RAPORTOINTIOHJELMA

Nimi	HiiVa-projekti
Toiminta-ajatus	Tällä hetkellä Vuokatin Hiihtokoulun jäsenrekisteri, päivystysvuorolista, opetustuntien varaukset ja seuranta hoidetaan manuaalisesti. Tulevaisuudessa HiiVa-projektin tuloksena syntyvällä ohjelmalla ylläpidetään edellä mainittuja toimintoja. Lisäksi ohjelmalla hoidetaan opetustoimintaan ja muuhun myyntiin liittyvät raportointitarpeet. Ohjelma helpottaa ja nopeuttaa yhdistyksen toiminnan seurantaa.
Yhdistyksen edustaja	Heidi Kelho
Ohjaava opettaja	Ritva Pihlajaniemi
Osallistuvat henkilöt	Juha Arffman Anne Lahti
Aikarajat suunnittelulle	Projektisuunnitelma valmis 31.1.2000.
Läpiviennin aikarajat	Projekti on valmis 2.5.2000.

Omistus- ja käyttöoikeus

Valmiin ohjelman omistusoikeus on Juha Arffmannilla ja Anne Lahdella. Vuokatin Hiihtokoululla on ohjelman käyttöoikeus.

Ohjelman päivitys — Ohjelmaa päivitetään erillisillä sopimuksilla.

VUOKATIN HIIHTOKOULU RY

Kajaanissa 4.1.2000

Heidi Kelho
puheenjohtaja

LIITE — luettelo ohjelman toiminnoista

TOIMEKSIANNON LIITE

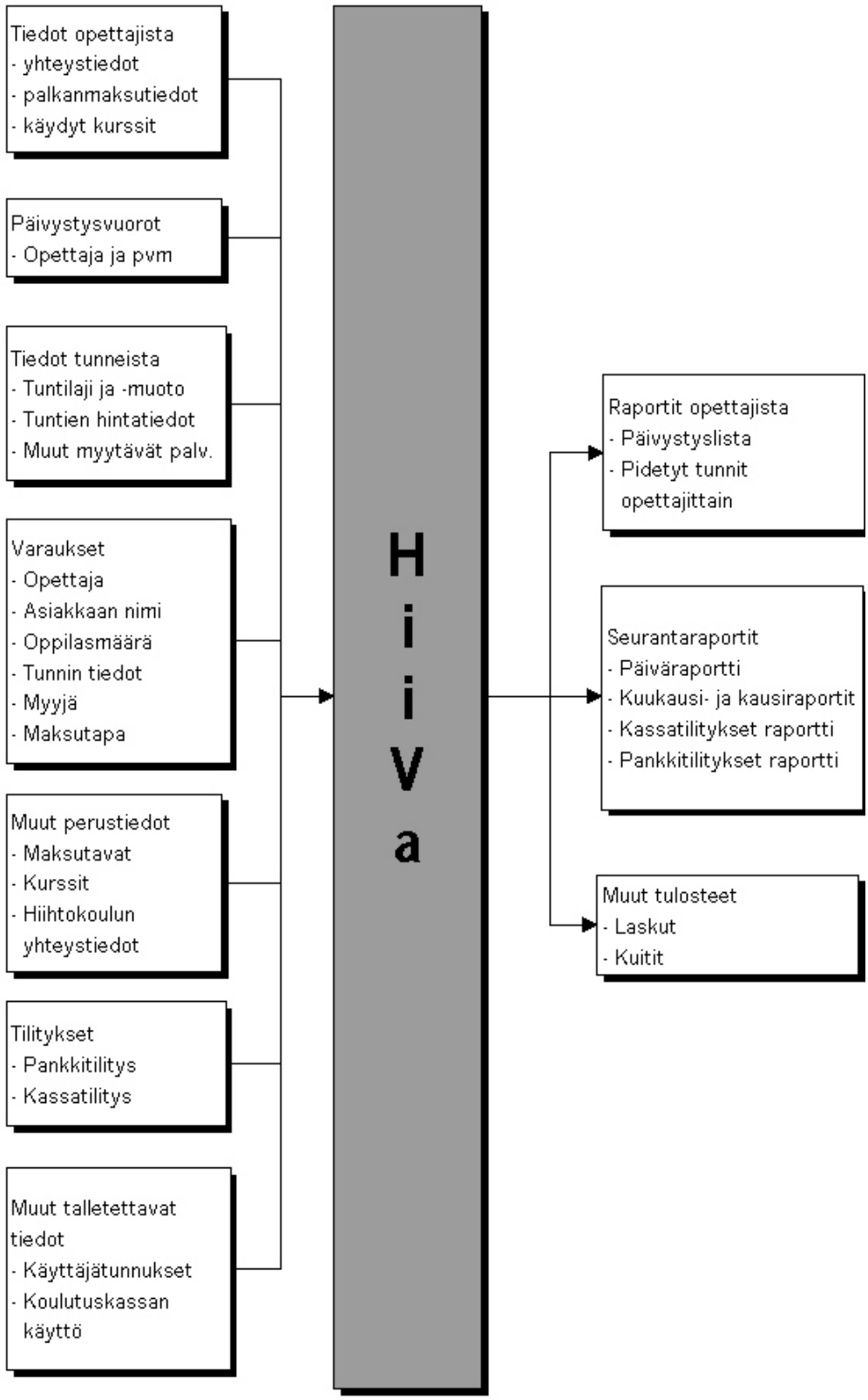
LUETTELO OHJELMAN TOIMINNOISTA

Ohjelmalle rajataan seuraavat toiminnot:

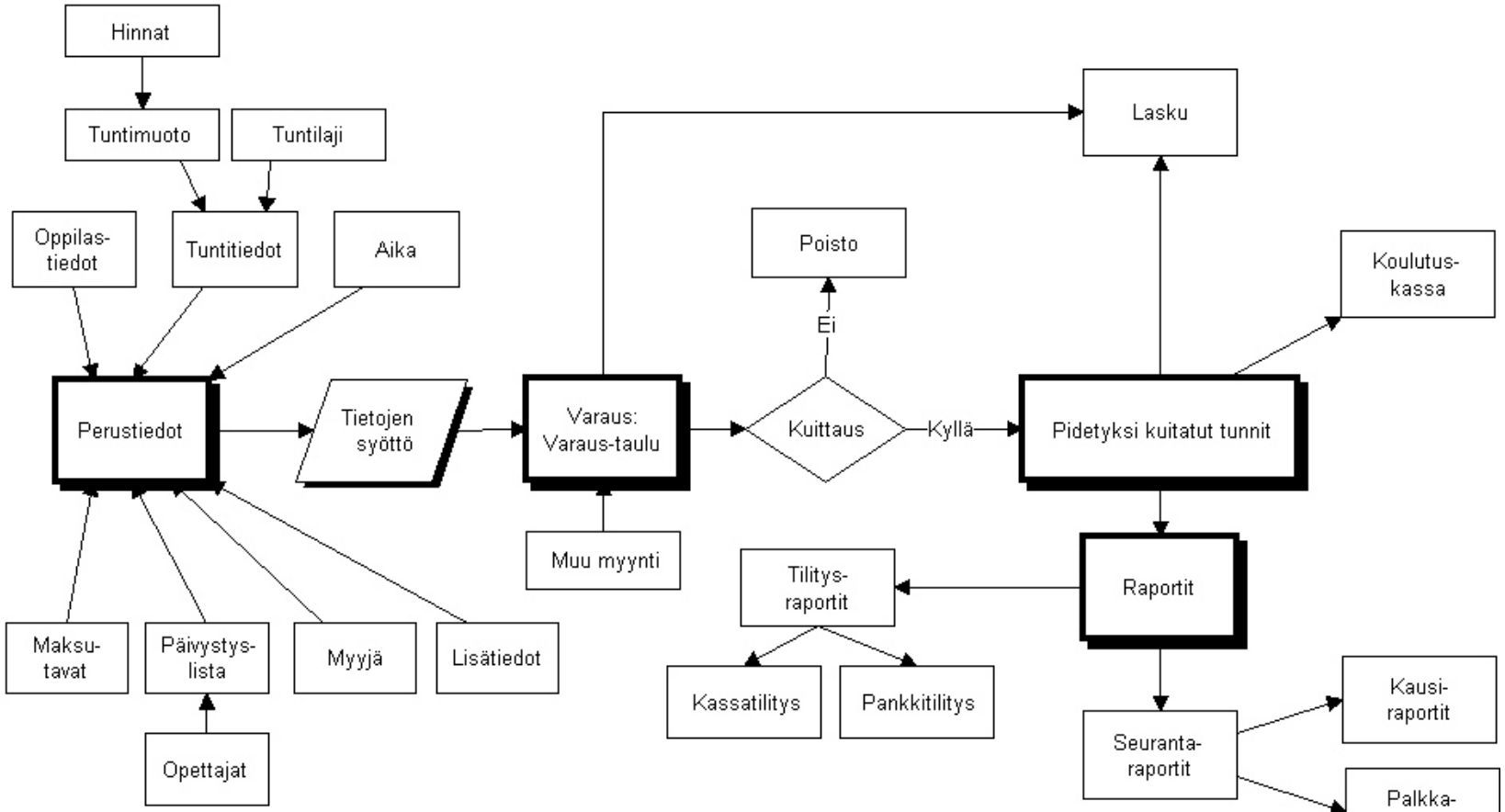
- varauskirjan ja kassakoneen korvaaminen
  - päivystysvuorojen tallennus ja ylläpito sekä päivystyslistan tulostus
  - opetustuntien tallennus ja ylläpito
- laskujen tekeminen ja tulostus
- opettajarekisterin ylläpito
  - yhteystiedot
  - käydyt kurssit
  - koulutuskassa
- käyttäjätasot (2 tasoa)
- myyntiraportointi
  - päivätilitysraportti
  - kuukausiraportti opettajittain
  - vuosiraportti
  - tuntijakaumat (laji/ryhmä)

Ohjelman ensimmäiseen versioon ei sisälly reskontra, verkkokäyttömahdollisuus tai muu sellainen mitä ei yllä ole mainittu.

HIIVA:N TOIMINTAKAAVIO

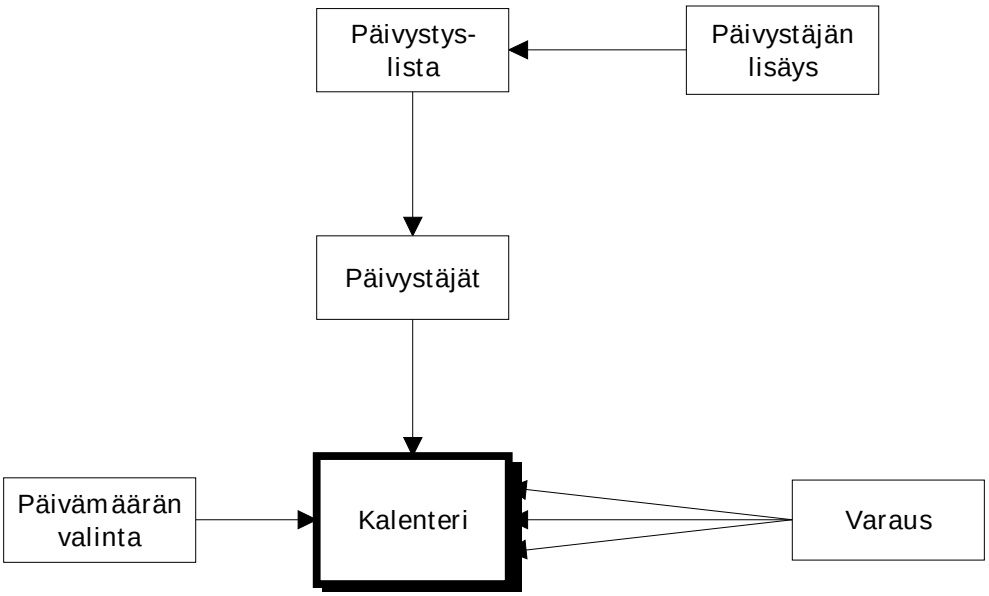


HiiVa: Ohjemakuvaus

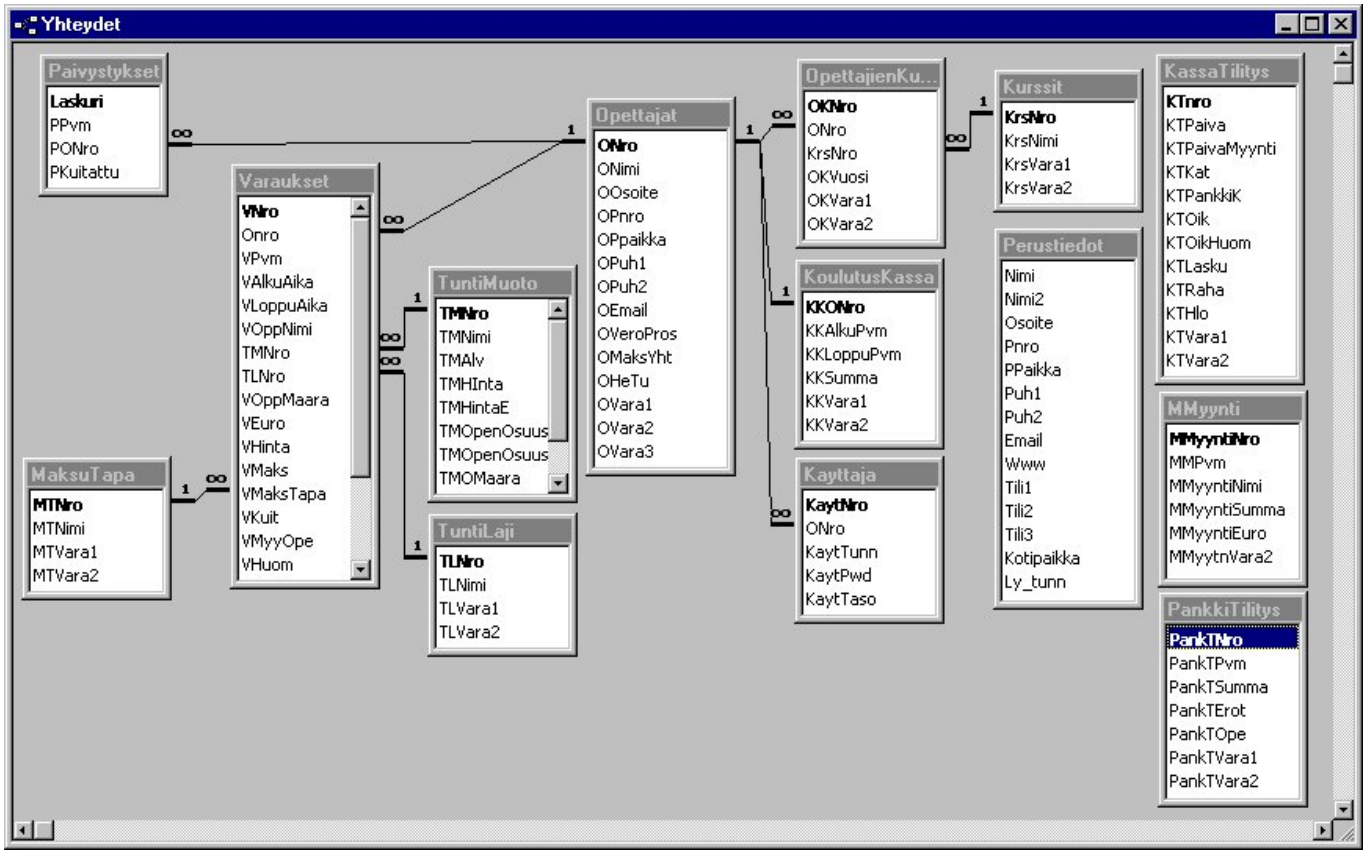


LIITE 4

HiiVa-projekti: Kalenterin kuvauskaavio



HIIVA:N KANTAYHTEYDET:

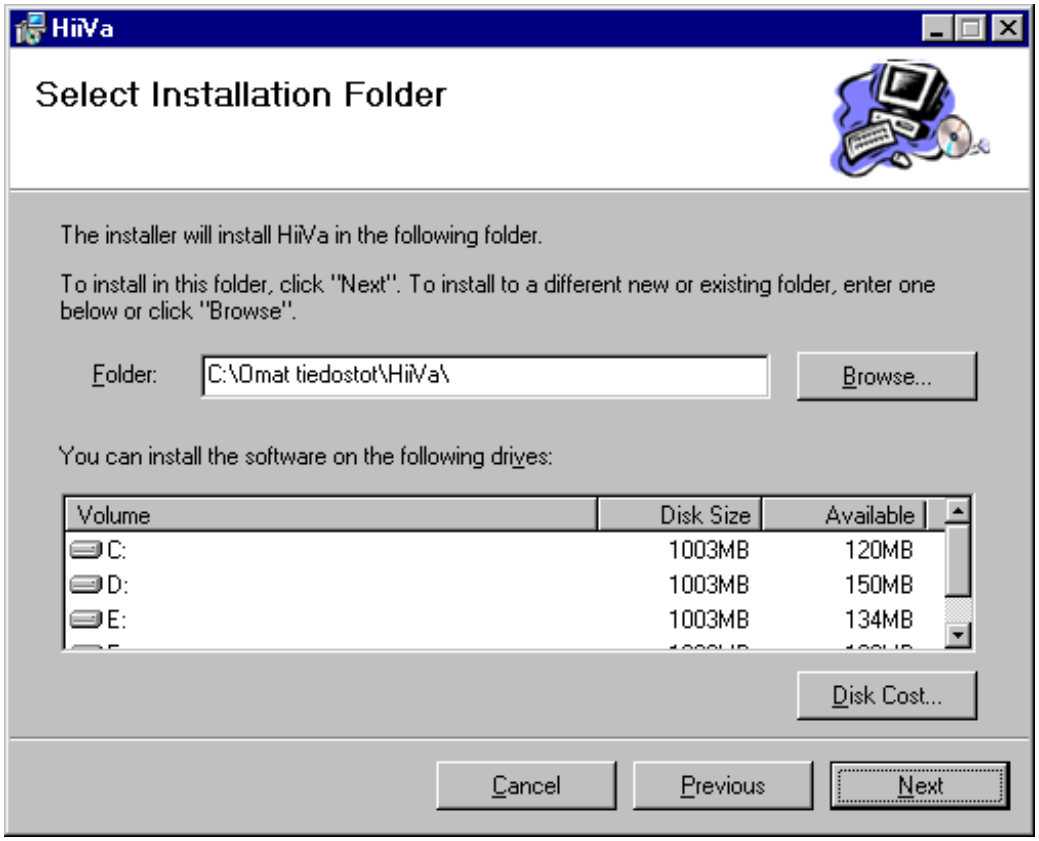


Vuokatin Hiihtokoulun Varaustenhallinta- ja raportointiohjelma (HiiVa): asennusohje

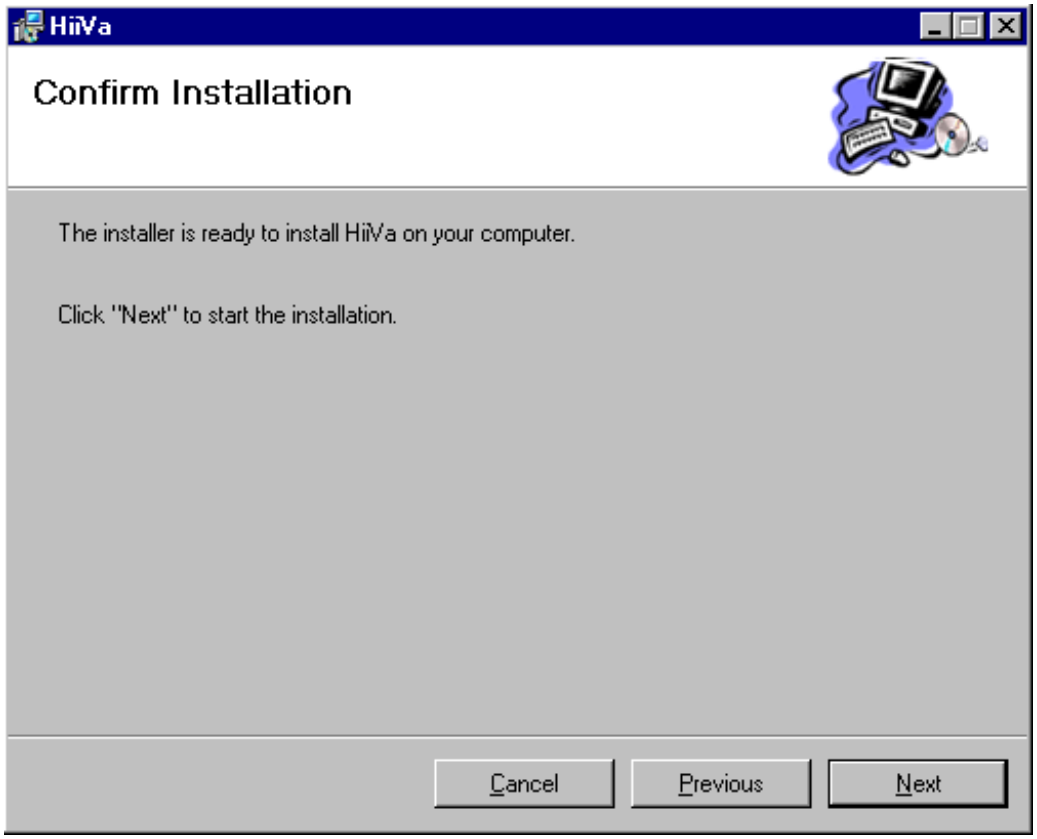
HiiVa:n asennusohjelma käynnistyy suorittamalla Setup.exe -tiedoston. Jos koneeseen ei ole asennettu Microsoft Data Access Components versiota 2.5, asennusohjelma ilmoittaa siitä ja keskeyttää ohjelman asennuksen. Microsoft Data Access Components versio 2.5 asennetaan käynnistämällä Mdac_typ.exe -tiedosto. Se asentaa tarvittavat tiedostot ja pyytää käynnistämään koneen uudelleen. Käynnistyksen jälkeen HiiVa:n asennus käynnistetään suorittamalla uudelleen Setup.exe.

Jos koneeseen ei ole asennettu Microsoft Installer -ohjelmaa, asennusohjelma asentaa sen automaattisesti ja jatkaa sen jälkeen HiiVa:n asennusosioon.

Käyttäjä voi halutessaan määrätä ohjelman asennuskansion valitsemalla levyaseman ja kohdekansion Select Installantion Folder -ikkunassa. Valinta hyväksytään klikkaamalla Next-painiketta.



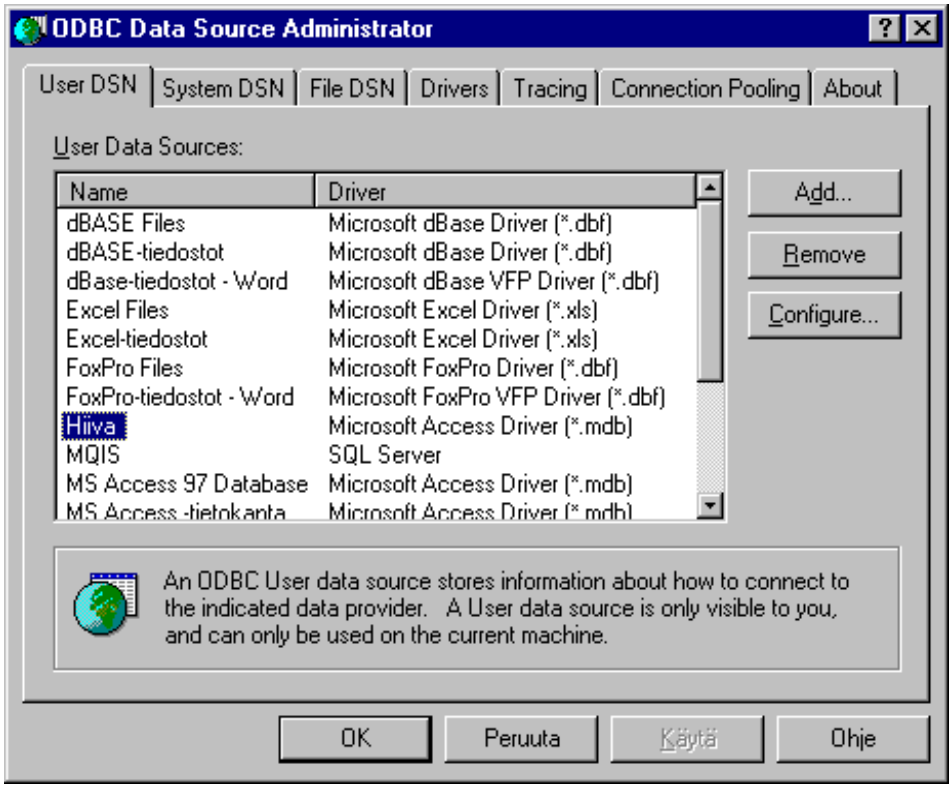
Asennuskansion määrityksen jälkeen ohjelman tiedostojen asennus alkaa klikkaamalla Next-painiketta Confirm Installation -ikkunassa.



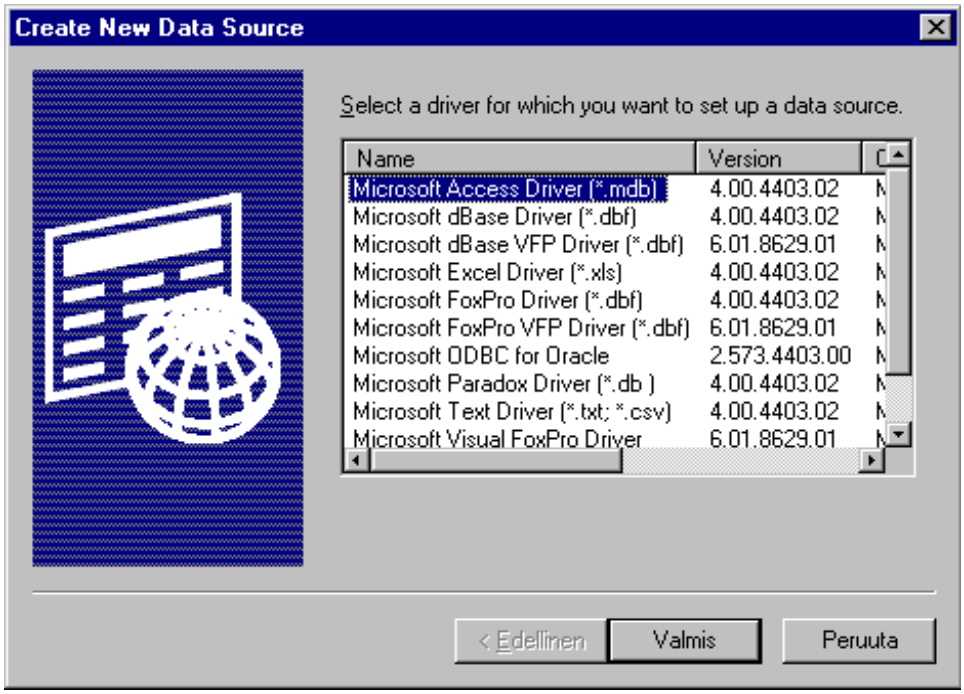
Asennusohjelma ilmoittaa kun asennus on päättynyt ja Close-painikkeella lopetetaan asennusohjelma.

ODBC-yhteyden luominen:

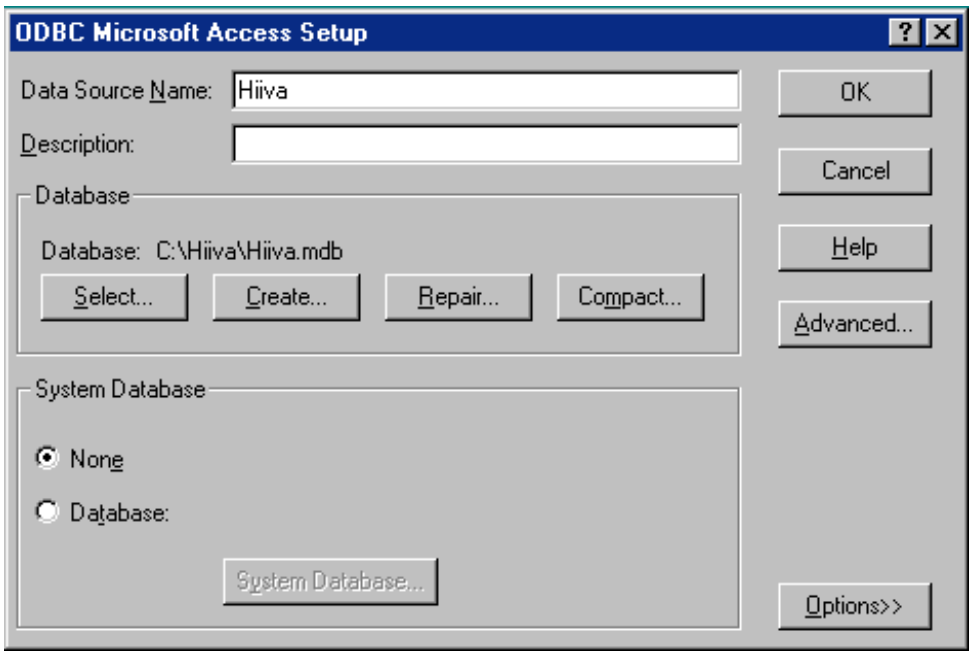
HiiVa tarvitsee toimiakseen Hiiva-nimisen ODBC-tietokantayhteyden, joka luodaan ohjauspaneelin ODBC Data Source Administrator -ikkunasta. Uuden yhteyden luominen aloitetaan klikkaamalla Add-painiketta.



Seuraavaksi valitaan tietokanta-ajuriksi Microsoft Access Driver ja klikataan Valmis-painiketta.



Data Source Name -kohtaan kirjoitetaan yhteyden nimeksi ”Hiiva” ja klikkaamalla Select-painikketta käydään valitsemassa HiiVa:n ohjelmakansiosta Hiiva.mdb -tiedosto yhteyden kohdetietokannaksi. Lopuksi uusi tietokantayhteys tallennetaan OK-painikkeella. Tämän jälkeen ODBC-yhteydet ikkunan luettelossa näkyy luotu yhteys ja ikkunan voi sulkea ja avata Hiihtokoulun varaus- ja raportointiohjelman.



Juha Arffman
Anne Lahti

LUOVUTUSSOPIMUS

9.10.2000

Vuokatin Hiihtokoulu

Juha Arffmannin (270670) ja Anne Lahden (241173) tekemän ja omistaman Vuokatin Hiihtokoulun Varaustenhallinta- ja raportointiohjelman käyttöoikeus luovutetaan Vuokatin Hiihtokoululle. Käyttöoikeuden lisäksi Vuokatin Hiihtokoulu saa oikeuden Vuokatin Hiihtokoulun Varaustenhallinta- ja raportointiohjelman lähdekoodin muuttamiseen omaan käyttöönsä. Vuokatin Hiihtokoulun Varaustenhallinta- ja raportointiohjelman ja sen lähdekoodin myyminen ja levittäminen ilman Vuokatin Hiihtokoulun Varaustenhallinta- ja raportointiohjelman omistajien lupaa on tekijänoikeuslain nojalla kielletty.

Vuokatin Hiihtokoulun Varaustenhallinta- ja raportointiohjelman omistus- ja lähdekoodioikeudet ovat Juha Arffmannilla ja Anne Lahdella.

Tätä sopimusta on tehty kolme kappaletta, yksi kaikille sopijapuolille.

Kajaanissa 9.10.2000

Vuokatin Hiihtokoulun Varaustenhallinta- ja raportointiohjelman tekijät ja omistajat

_____ _____
Juha Arffman Anne Lahti

Vuokatin Hiihtokoulu

_____
Heidi Kelho

HIIVA -PROJEKTIN AJANKÄYTTÖKAAVIO:

HiiVa-projektin suunniteltu ajankäyttö

Tapahtuma / viikko	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Projektin aloitus ja suunnittelu																		
Sovelluksen kuvaus ja suunnittelu																		
Sovelluksen ohjelmointi 1. Vaihe																		
1. Testaus																		
Sovelluksen ohjelmointi 2. Vaihe																		
2. Testaus																		
Sovelluksen ohjelmointi 3. Vaihe																		
Projektin päättäminen																		

HiiVa-projektin toteutunut ajankäyttö

Tapahtuma / viikko	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41
Projektin aloitus ja suunnittelu																																									
Sovelluksen kuvaus ja suunnittelu																																									
Sovelluksen ohjelmointi 1. Vaihe																																									
1. Testaus																																									
Sovelluksen ohjelmointi 2. Vaihe																																									
2. Testaus																																									
Sovelluksen ohjelmointi 3. Vaihe																																									
Projektin päättäminen																																									

Kokonaistyöajan jakautuminen:

- Sovelluksen toteutus n. 700 tuntia
- Julkaisun toteutus n. 110 tuntia

LIITE 9