SAVONIA-AMMATTIKORKEAKOULU
LIIKETALOUS, KUOPIO

SELAINKÄYTTÖLIITTYMÄN VALINTA JA TOTEUTUS HERALES RAPORTOINTIIN

Anna-Leena Miettinen

Tradenomin opinnäytetyö

Tietojenkäsittelyn koulutusohjelma

Marraskuu 2010

SAVONIA-AMMATTIKORKEAKOULU
LIIKETALOUS, KUOPIO

Koulutusohjelma, suuntautumisvaihtoehto (jos on)

Tietojenkäsittelyn koulutusohjelma

Tekijä(t)

Anna-Leena Miettinen

Työn nimi

Selainkäyttöliittymän valinta ja toteutus Herales Raportointiin

Työn laji	Päiväys	Sivumäärä
Opinnäytetyö	19.11.2010	28 + 4

Työn ohjaaja(t)	Toimeksiantaja
Jyrki Linja	Herales Oy

Tiivistelmä

Opinnäytetyössä kuvataan Herales Raportointi-sovelluksen selainkäyttöliittymän valinta- ja toteutusprosessia. Herales Raportointi on Herales Ketjukonseptiin kuuluva johdon raportointijärjestelmä, joka on suunniteltu erityisesti suurten tietomäärien analysointiin ja vertailuun. Kehitettävän sovelluksen tuli olla sellainen, että sitä pystytään käyttämään internetin kautta käyttäjätunnuksella ja salasanalla.

Kehitysympäristöä valittaessa ensimmäisenä vaihtoehtona tutkittiin ASP.NET-teknologian ja Web-palveluiden soveltuvuutta selainkäyttöliittymän toteuttamiseen. Toimeksiantajalla oli valmiita Progress-kehittimellä tehtyjä Web-palveluja. Tiedonkulku Progress- ja ASP.NET-kehitysympäristöjen välillä osoittautui kuitenkin ongelmalliseksi, joten ASP.NET:in ja Web-palveluiden käytöstä sovelluksen toteuttamisessa päätettiin luopua. Tästä syystä sovellus päätettiin kehittää toisena vaihtoehtona olleella Progress WebSpeed-teknologialla.

Opinnäytetyössä esitellään valmis sovellus, joka sisältää sisäänkirjautumissivun, valikkosivun ja yhden raporttinäytön sekä lopetussivun. Sovelluksen ohjelmakoodi on toimeksiantajan pyynnöstä määrätty salassa pidettäväksi, joten sitä ei opinnäytetyössä esitellä. Opinnäytetyötä varten tehty sovellus toimii esimerkkinä myöhemmin kehitettävälle tuotantoon tulevalle sovellukselle.

Asiasanat

Selainkäyttöliittymä, ASP.NET, Web-palvelut, Progress WebSpeed

Huomioitavaa

SAVONIA UNIVERSITY OF APPLIED SCIENCES
UNIT OF BUSINESS AND ADMINISTRATION, KUOPIO

Degree Programme, option

Computer Science

Author(s)

Anna-Leena Miettinen

Title of study

Selecting and Making a Web-based User Interface for the Herales Reporting System

Type of project	Date	Pages
Thesis	19.11.2010	28 + 4

Supervisor(s) of study	Executive organisation
Jyrki Linja	Herales Oy

Abstract

This thesis describes the selection and implementation process of a web-based user interface for the Herales Reporting System. The system belongs in the Herales Chain Concept and is a tool for management reporting. It has been designed especially for analyzing and comparing a large amount of data. The objective was to develop an application that could be accessed via the Internet with a username and password.

When the development environment was chosen the first option was to examine the suitability of ASP.NET-technology and Web Services for development of a web-based user interface. The executive organization already employed Web Services resources which were made using Progress application generator. The flow of information between Progress and ASP.NET environments proved to be problematic, so it was decided to abandon ASP.NET and Web Services in the application development and to create the application with Progress WebSpeed-technology, which was the second option as a development environment.

This thesis presents the complete application which consists of a login page, a menu page, one report display and a page for quitting the application. The source code of the application was classified as confidential at the request of the executive organization, so it is not revealed in this thesis. The application made for the thesis serves as a template for a production-ready application which will be developed later.

Keywords

Web-based User Interface, ASP.NET, Web Services, Progress WebSpeed

Note

SISÄLLYS

1 JOHDANTO 5
1.1 Työn toimeksiantaja 5
1.2 Herales Raportointi 5
1.3 Kehitettävä sovellus 6
1.4 Rajaus 7
2 ASP.NET, MICROSOFT VISUAL WEB DEVELOPER 2008 EXPRESS EDITION JA WEB-PALVELUT MAHDOLLISENA TOTEUTUSVÄLINEENÄ 8
2.1 Web-palvelut 8
2.2 Web-palveluiden käyttö selainkäyttöliittymän ohjelmoinnissa 9
3 TOTEUTUS PROGRESS WEBSPEED-TEKNOLOGIALLA 13
3.1 Ohjelmointikielten sukupolvet 13
3.2 Progress 4GL 13
3.3 Progress WebSpeed 14
3.4 Selainkäyttöliittymän toteutus 15
3.4.1 Myyntiraportin toimintalogiikka 17
3.4.2 Selainkäyttöliittymän tekninen toteutus 21
4 POHDINTA 26
LIITE 1 TableFloaterTitle.js 29

1 JOHDANTO

Tämän työn tarkoituksena on kuvata selainkäyttöliittymän valinta- ja toteutusprosessia Herales Raportointiin. Työssä selvitetään ensin ASP.NET-teknologian ja Web-palveluiden soveltuvuutta selainkäyttöliittymän toteutukseen. Edellä mainittua selvitystyötä tein lähinnä itsenäisesti ja raportoin havainnoista muulle projektiryhmälle. Koska ASP.NET ja Web-palvelut eivät soveltuneet selainkäyttöliittymän ohjelmointiin, sovellus päädyttiin kehittämään Progress WebSpeed-teknologialla. Tässä kehitystyössä vastasin lähinnä HTML-kielen kirjoittamisesta ja sen upottamisesta Progress 4GL-koodiin. Tein myös selainkäyttöliittymän CSS-tyylimäärittelyt. Muu projektiryhmä, jolla oli aikaisempaa kokemusta Progress 4GL-ohjelmoinnista, vastasi kehitystyön tästä osuudesta.

1.1 Työn toimeksiantaja

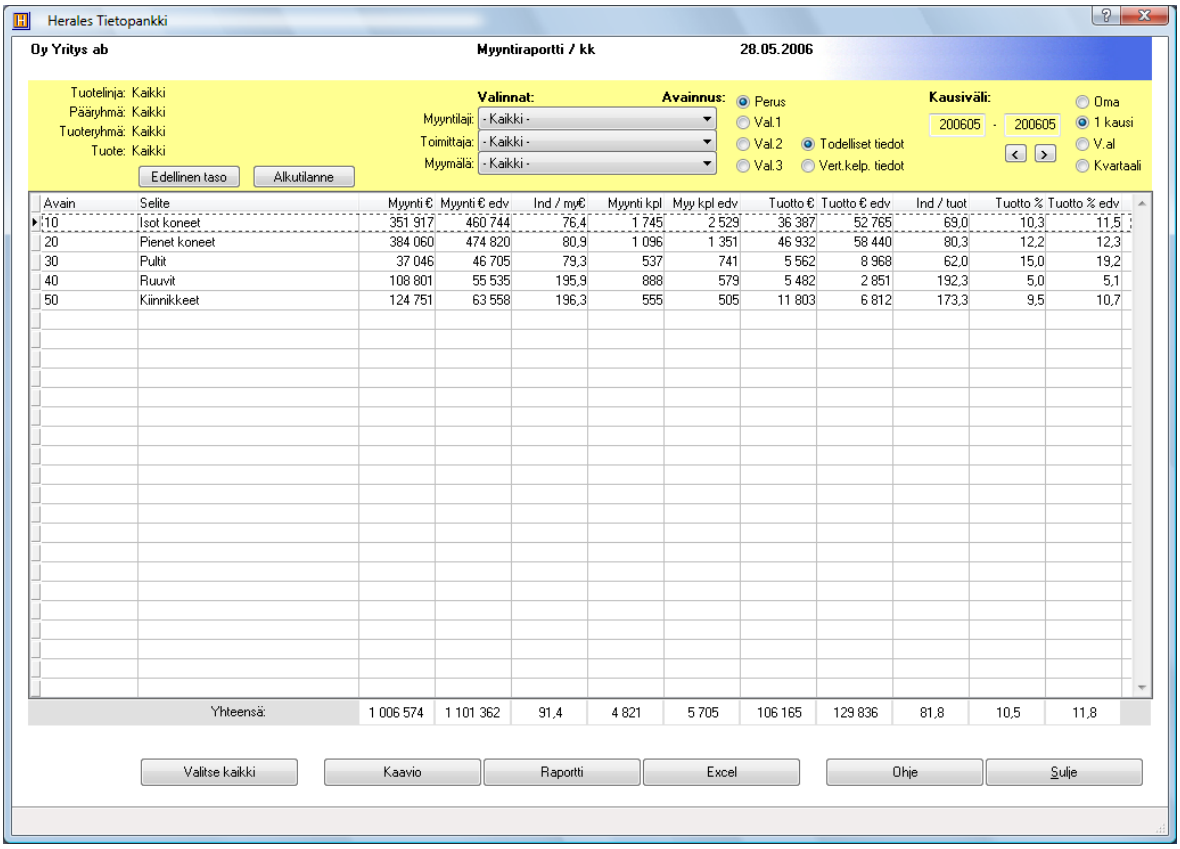
Herales Oy on ohjelmistotuotantoon ja palvelukeskustoimintaan erikoistunut kuopiolainen atk-palvelutalo. Yritys on perustettu vuonna 1990. Herales Oy:n asiakaskunta koostuu tukku-, erikoistavara- ja päivittäistavarakaupan sekä teollisuuden toimialoilla toimivista liikeyrityksistä. (Herales Oy, 2010)

Herales Oy:n tuotteita ovat Herales Ketjukonsepti-tuoteperhe sekä sen ohella myös yksilölliset ohjelmaratkaisut. Herales Ketjukonsepti on yhdistelmä ohjelmatuotteita ja niihin liittyviä palveluja. Ohjelmatuotteet ovat yksilöitäviä ja ne sopivat erityisesti ketjujen ja monimyymäläyritysten käyttöön. Herales Ketjukonseptiin kuuluvia tuotteita ovat Herales Raportoinnin lisäksi mm. Herales Työaikasuunnittelu sekä Herales Tuotehallinta. (Herales Oy, 2010)

1.2 Herales Raportointi

Herales Raportointi on Herales Ketjukonseptiin kuuluva johdon raportointijärjestelmä. Se on suunniteltu erityisesti suurten tietomäärien analysointiin ja vertailuun. Sovelluksen perustana ovat vakiomuotoiset raporttiruudut (ks. Kuva 1), joiden sisällä liikutaan tiedon eri tasoilla. Ohjelmiston rakenne on sellainen, että sitä voidaan

käyttää joustavasti eri raportointikohteisiin. Tyypillisiä käyttökohteita Herales Raportoinnille ovat mm. myynnin, ostojen ja varaston päivittäinen seuranta monimyymälä- tai monitulosyksikköympäristössä. Sovellus on toteutettu Progress 4GL-kehittimellä. Työasemien ja palvelimen välinen tietoliikenne on toteutettu Progress WebClient-tekniikan avulla. (Herales Oy, 2010)



Kuva 1. Herales Raportointi-järjestelmän myyntiraportti-näyttö.

1.3 Kehitettävä sovellus

Projektin tavoitteena oli nykyisen sovelluksen pohjalta kehittää puhtaasti selainpohjainen käyttöliittymä Herales Raportointiin. Nykyisellään Herales Raportointi vaatii käyttäjän koneelle asennettavaksi Progress WebClient-ohjelman, joten sovellusta voidaan käyttää vain niissä koneissa, joissa on ko. ohjelma. Kehitettävän sovelluksen tuli siis olla ehdottomasti sellainen, ettei se vaadi mitään asennettavia ohjelmistoja käyttäjän tietokoneeseen, vaan sitä pystyy käyttämään internetin kautta käyttäjätunnuksella ja salasanalla. Tämä mahdollistaisi sovelluksen käytön myös mobiililaitteilla.

1.4 Rajaus

Opinnäytetyötä varten tehdyn sovelluksen tarkoituksena on toimia esimerkkinä myöhemmin kehitettävälle ”todelliselle” selainpohjaiselle sovellukselle. Projekti on siis opinnäytetyön osalta saatettu loppuun, kun on saatu toteutettua yksi valmis ja toimiva raporttinäyttö, joka käyttää ttp-nimistä testitietokantaa.

Toimeksiantajan pyynnöstä selainkäyttöliittymän ohjelmakoodi on kokonaisuudessaan määrätty salassa pidettäväksi, joten opinnäytetyössä ei tästä syystä käydä läpi sovelluksen ohjelmakoodia.

2 ASP.NET, MICROSOFT VISUAL WEB DEVELOPER 2008 EXPRESS EDITION JA WEB-PALVELUT MAHDOLLISENA TOTEUTUSVÄLINEENÄ

Projektin alussa, mietittäessä mahdollisia kehitysympäristöjä, päätettiin ensimmäiseksi kokeilla Microsoft Visual Web Developer 2008 Express Editionin sopivuutta sovelluksen kehittämiseen. Kyseisen sovelluskehittimen kokeilemiseen päädyttiin lähinnä siitä syystä, että se on ilmainen ja helppokäyttöinen, mutta sen avulla on mahdollista toteuttaa vaativiakin dynaamisia selainpohjaisia ratkaisuja ASP.NET-teknologialla. Helppokäyttöisyyttä lisää mm. WYSIWYG (What You See Is What You Get) -editori, jonka avulla mm. sivuston ulkoasua voidaan muokata helposti ilman käsin kirjoitettavaa koodia.

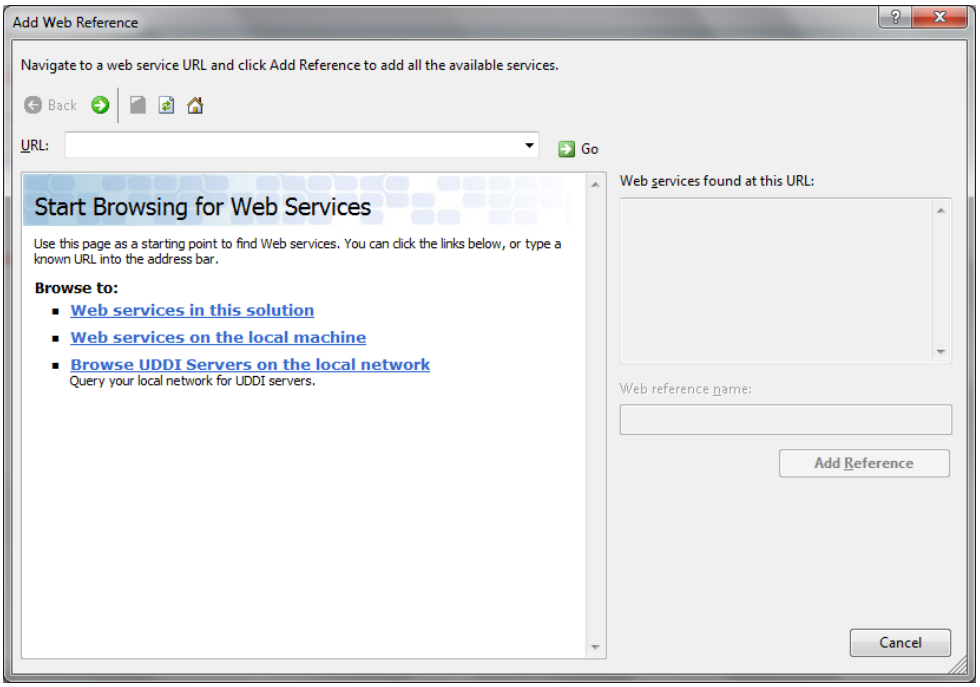
2.1 Web-palvelut

W3C määrittelee Web-palvelun (engl. Web service) seuraavasti: *”A Web service is a software system designed to support interoperable machine-to-machine interaction over a network”*. Web-palvelu on siis ohjelmistojärjestelmä, joka on suunniteltu tukemaan keskenään yhteensopivaa tietokoneiden välistä vuorovaikutusta tietoverkon yli. Tämä tarkoittaa sitä, että Web-palvelu mahdollistaa tiedonsiirron keskenään erilaisten sovellusten ja sovellusalustojen välillä. (W3Schools, 2010)

Web-palveluiden sovellusalusta koostuu kolmesta perusosasta, jotka ovat SOAP, WSDL ja UDDI. SOAP (Simple Object Access Protocol) on XML-pohjainen protokolla, jonka avulla eri sovellukset voivat vaihtaa tietoja http-protokollan yli. WSDL (Web Services Description Language) on XML-pohjainen Web-palveluiden kuvauskieli; WSDL-dokumentti sisältää joukon palvelua kuvaavia määrityksiä, kuten palvelun käyttämät tietotyypit ja sen suorittamat toiminnot. UDDI (Universal Description, Discovery and Integration) puolestaan on Web-palveluiden rekisteröintiin ja etsimiseen tarkoitettu hakemistopalvelu. (W3Schools, 2010)

2.2 Web-palveluiden käyttö selainkäyttöliittymän ohjelmoinnissa

Koska valmis sovellus tulisi käyttämään Progress-tietokantaa, jouduttiin miettimään, millä tavoin Progress- ja ASP.NET-kehitysympäristöjen välinen tiedonkulku saataisiin onnistumaan. Herales Oy:lla oli valmiita Progress-kehittimellä tehtyjä Web-palveluja, joiden käyttämistä päätettiin kokeilla ASP.NET-ympäristössä. Jos palveluiden käyttäminen onnistuisi vaivattomasti, olisi selainkäyttöliittymän ohjelmointi helppoa; Visual Web Developerilla toteutettaisiin sovelluksen ulkoasu ja valmiiksi tehdyillä Web-palveluilla haettaisiin Progress-tietokannasta halutut tiedot jokaiseen komponenttiin. Jos esimerkiksi haluttaisiin täyttää jokin käyttöliittymän pudotusvalikko myymälöiden tiedoilla, liitettäisiin myymälätietojen hakemiseen tarkoitettu Web-palvelun metodi pudotusvalikkoon. Aluksi Web-palveluiden liittämistä ASP.NET-ympäristöön kokeiltiin hakemalla internetistä sivustolta http://www.xmethods.com/ erilaisia Web-palveluita. Visual Web Developerin Add Web Reference-työkalulla voidaan etsiä Web-palveluita internetistä url-osoitteen perusteella (ks. Kuva 2).



Kuva 2. Web-palveluiden etsintä.

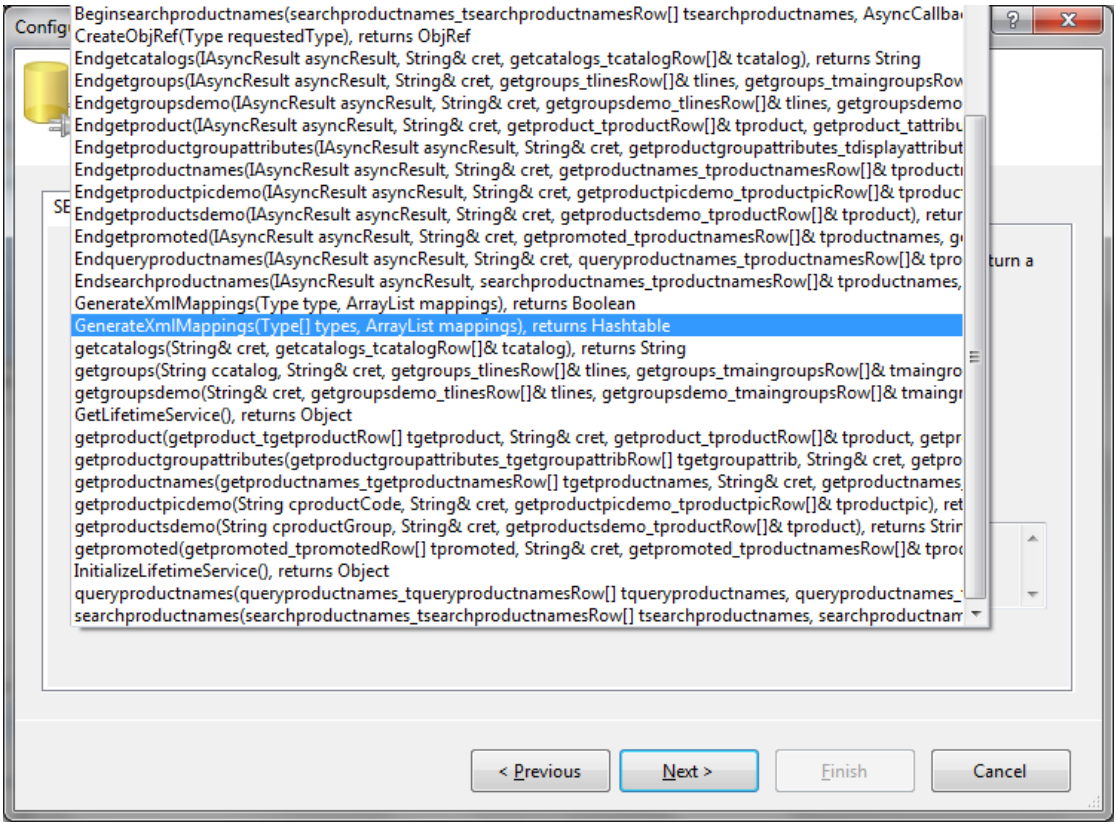
Esimerkkinä voidaan hakea internetistä GBNIR Holiday Service, joka palauttaa irlantilaisten juhlapäivien tietoja. Esimerkissä käytetään GridView-komponenttia tietojen näyttämiseen. Kun palvelu on haettu Visual Web Developerilla luodun www-sivun App_WebReferences-kansioon, raahataan näytölle GridView-komponentti, jolle

määritetään DataSourceksi ObjectDataSource. Seuraavaksi valitaan Choose your business object-valikosta com.holidaywebservice.www.GBNIRHolidayService ja sille esimerkiksi metodi GetHolidaysAvailable(), joka palauttaa DataSet-komponentin. Tämän jälkeen saadaan www-sivulle näkyviin kuvassa 3 näkyvä taulukko, joka sisältää haetut tiedot.

Name	Key	Type
New Year's Day	NEW_YEARS	Bank Holiday
Good Friday	GOOD_FRIDAY	Bank Holiday
Easter Monday	EASTER_MONDAY	Bank Holiday
May Day (Early May Bank Holiday)	MAY_DAY	Bank Holiday
Spring Bank Holiday	SPRING_BANK_HOL	Bank Holiday
Summer Bank Holiday	SUMMER_BANK_HOL	Bank Holiday
Christmas	CHRISTMAS	Bank Holiday
Boxing Day	BOXING_DAY	Bank Holiday
New Year's Eve	NEW_YEARS_EVE	Notable Date
Robert Burns Night (Burns Night)	BURNS_NIGHT	Notable Date
Holocaust Memorial Day	HOLOCAUST	Notable Date
Shrove Tuesday (Pancake Day)	SHROVE_TUES	Notable Date
Ash Wednesday	ASH_WEDS	Religious Date
Valentines Day	VALENTINES	Notable Date
St David's Day	SAINT_DAVIDS	Religious Date
Mothering Sunday (Mothers Day)	MOTHERS	Notable Date
St Patricks Day	ST_PATRICKS_DAY	Bank Holiday

Kuva 3. GBNIR Holiday Service.

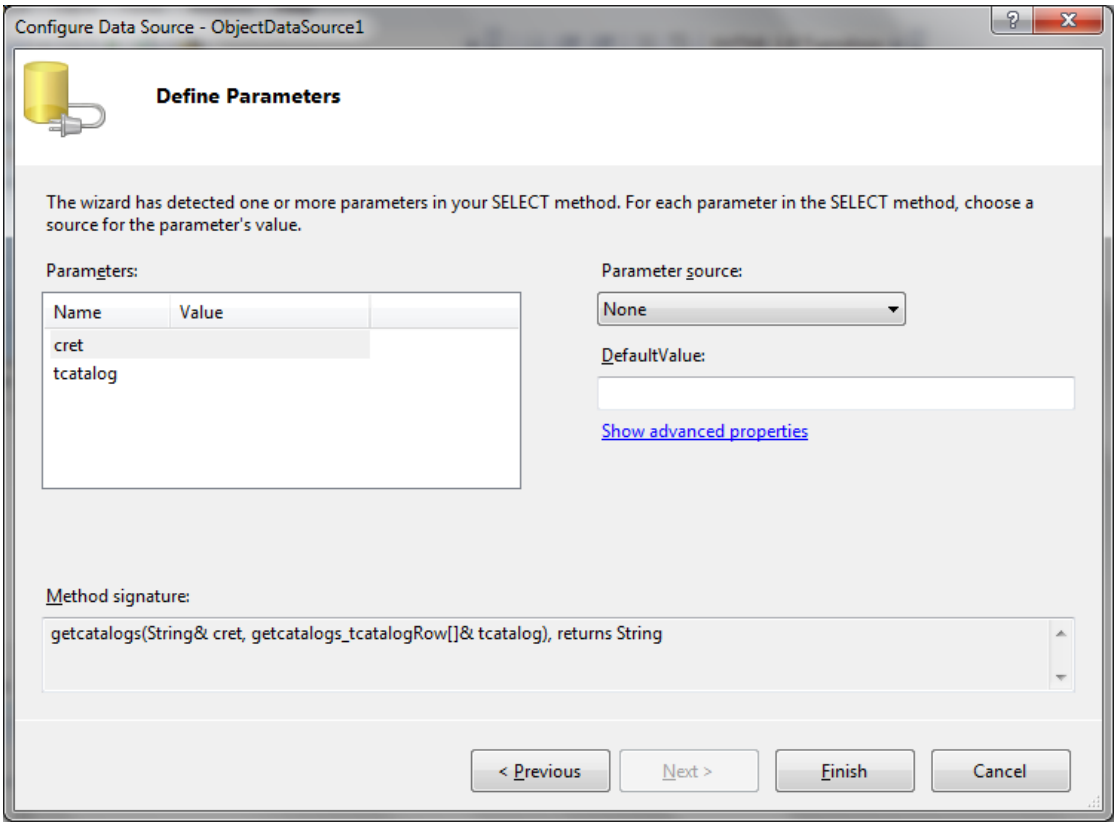
Kun Web-palveluiden käyttöönottoa oli testattu internetistä haetuilla palveluilla, siirryttiin seuraavaksi testaamaan Herales Oy:n omia, Progress-kehittimellä toteutettuja palveluja. Tässä vaiheessa ei ollut väliä sillä, missä muodossa tiedot saataisiin esitettyä; pääasia oli, että saataisiin muodostettua Web-palvelun avulla jonkinlainen yhteys tietokannan ja käyttöliittymän välille. App_WebReferences-kansioon haettiin Herales Web Store-palvelu ja samalla tavoin kuin yllämainitussa esimerkissä, näytölle raahattiin GridView-komponentti ja sille määritettiin DataSourceksi ObjectDataSource sekä Choose your business object-valikosta valittiin WebReference.HeralesWebStoreService. Tässä vaiheessa alkoi esiintyä ongelmia; palvelun metodit (ks. Kuva 4) olivat vaikeaselkoisia ja ne näyttivät palauttavan täysin vääränlaisia tietotyyppejä.



Kuva 4. Herales Web Store-palvelun metodeja.

Kun metodiksi valittiin getcatalogs(), ohjelma ilmoitti, että metodilla on yksi tai useampi parametri, ja että parametrien arvoille tulee määrittää lähde. Jos tässä vaiheessa (ks. Kuva 5) painetaan Finish-painiketta, ei www-sivulle tule näkyviin mitään.

On mahdollista, että ongelma olisi saatu ratkaistua, mutta tämä olisi vaatinut syvällistä xml-osaamista sekä todennäköisesti paljon manuaalista ohjelmointia. Koska kenelläkään projektiryhmän jäsenistä ei ollut mahdollisuutta alkaa tutkimaan ongelmaa syvällisemmin, päätettiin ASP.NET:in ja Web-palveluiden käyttö selainkäyttöliittymän ohjelmoinnissa hylätä.



Kuva 5. Lähteiden määrittäminen parametrien arvoille.

3 TOTEUTUS PROGRESS WEBSPEED-TEKNOLOGIALLA

3.1 Ohjelmointikielten sukupolvet

Ohjelmointikielet voidaan jakaa eri sukupolviin sen mukaan, kuinka etäällä kieli on ohjelmaa suorittavasta laitteesta ja toisaalta kuinka lähellä se on itse ohjelmoinnin kohdetta eli sovellusta (Knuutila, 2001). Ensimmäisen sukupolven kielet ovat puhtaasti konekielisiä, binäärimuotoisia komentoja. Toisen sukupolven kielet ovat symbolisia konekieliä, joissa yhtä konekielistä käskyä vastaa tekstisymboli. Symboliset konekielet käännetään konekieleksi erityisen ohjelman, assemblerin, avulla. Kolmannen sukupolven kielet ovat korkean tason ohjelmointikieliä, joiden kääntämiseen tarvitaan kääntäjä tai tulkki. Nykyiset ohjelmointikielet ovat suurimmaksi osaksi kolmannen tason kieliä; yleensäkin oliokielet lasketaan kolmannen tason kieliksi. Neljännen sukupolven kieliä ovat mm. verkkoympäristöjen kielet ja tietokantojen kyselykielet. Neljännen sukupolven kielten syntaksi on lähempänä luonnollisia kieliä, joten niitä on helpompi kirjoittaa. Viidennen sukupolven kielet, tekoälykielet, muistuttavat syntaksiltaan yleensä luonnollista kieltä. (Vesanen, 2005)

3.2 Progress 4GL

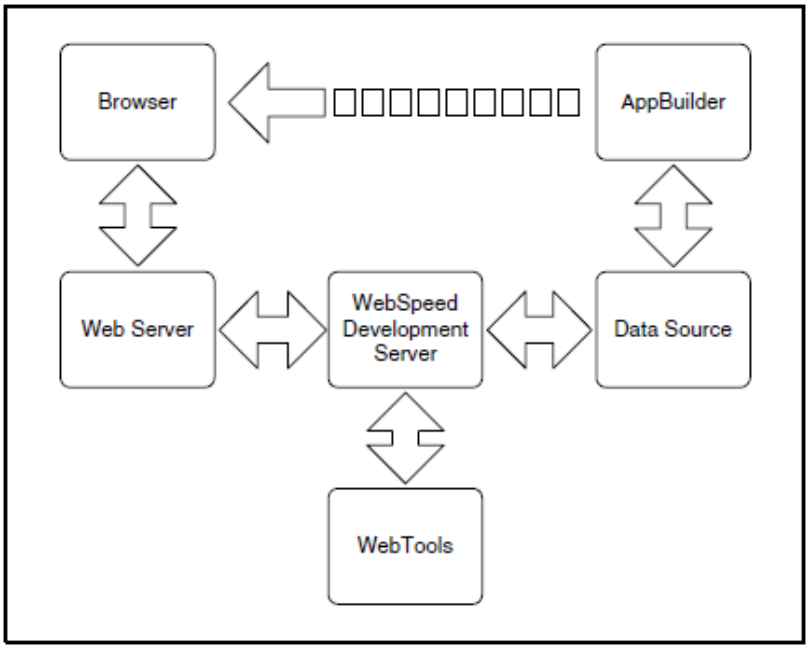
Progress 4GL (fourth-generation language, neljännen sukupolven kieli) on Progress Software Corporationin kehittämä proseduraalinen korkean tason ohjelmointikieli, joka on tarkoitettu erityisesti yritysten tietojärjestelmien ohjelmointiin. Progress 4GL on joustava ohjelmointikieli. Sen etuna verrattuna tavallisiin kolmannen sukupolven kieliin ovat tehokkaat lauseet ja avainsanat; siinä missä 3GL:lla (mm. Visual Basic, Java) tarvittaisiin kymmeniä tai jopa satoja koodirivejä, 4GL:lla yksittäinen lause voi hoitaa saman asian. Toisaalta Progress 4GL mahdollistaa ohjelmoinnin suurella tarkkuudella; jopa yksittäisten bittien poimiminen tietovirrasta on mahdollista.
(Sadd, 2003)

3.3 Progress WebSpeed

WebSpeed on dynaamisten selainpohjaisten sovellusten ohjelmointiin tarkoitettu kehitysympäristö. WebSpeedin kehitys- ja käyttöönottoympäristö koostuu seuraavista sovelluskomponenteista (ks. myös Kuva 6):

- WebSpeed Workshop
  - AppBuilder, graafinen työkalu WebSpeed-sovellusten rakentamiseen
  - WebTools, kokoelma selainpohjaisia palveluja
  - WebSpeed Development Server, palvelin, joka ajaa AppBuilderilla kehitettyjä sovelluksia
- Web-selain
- Web-palvelin
- Tietokantapalvelin

(WebSpeed Developer's Guide, 2001)



Kuva 6. WebSpeed-kehitysympäristön komponentit (WebSpeed Developer's Guide, 2001).

WebSpeed mahdollistaa sovellusten ohjelmoinnin seuraavilla tekniikoilla: upotettu SpeedScript (Embedded SpeedScript), HTML Mapping ja CGI Wrapper (josta tarkemmin luvussa 3.4.2). SpeedScriptia voidaan upottaa HTML-koodiin käyttämällä

<SCRIPT>-tagia (ks. Kuva 7). SpeedScript toimii kuitenkin eri tavalla kuin esim. JavaScript; JavaScript ajetaan selaimessa, kun taas SpeedScript ajetaan palvelimessa. HTML Mapping-tekniikka puolestaan mahdollistaa käyttöliittymän ja sovelluslogiikan erottamisen erillisiin tiedostoihin; sen avulla voidaan linkittää määritellyn HTML-lomakkeen kentät määriteltyihin tietokannan kenttiin. (WebSpeed Developer's Guide, 2001)

escript1.htm

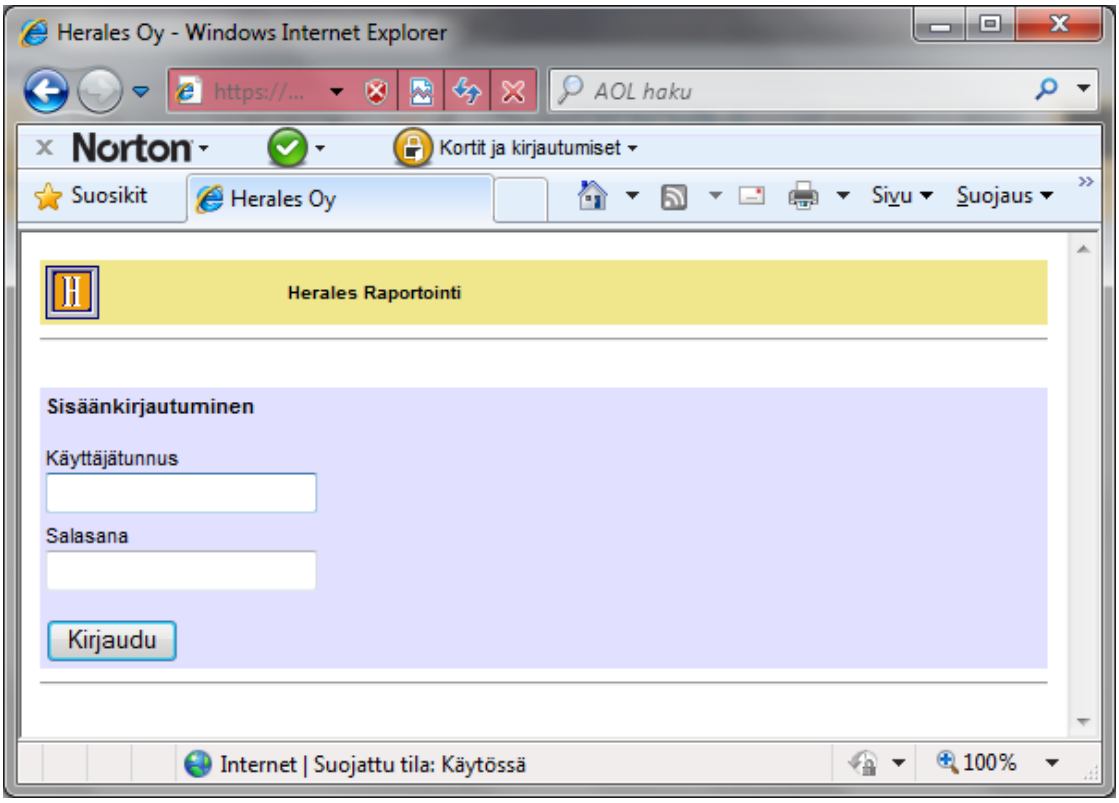
```
<HTML>
<HEAD>
<TITLE>My First Embedded SpeedScript File</TITLE>
</HEAD>
<BODY>
<H1>My First Embedded SpeedScript File</H1>

<SCRIPT LANGUAGE="SpeedScript">
FOR EACH Customer WHERE Name BEGINS "s":
   DISPLAY {&WEBSTREAM} Customer.
END.
</SCRIPT>
<HR>
</BODY>
</HTML>
```

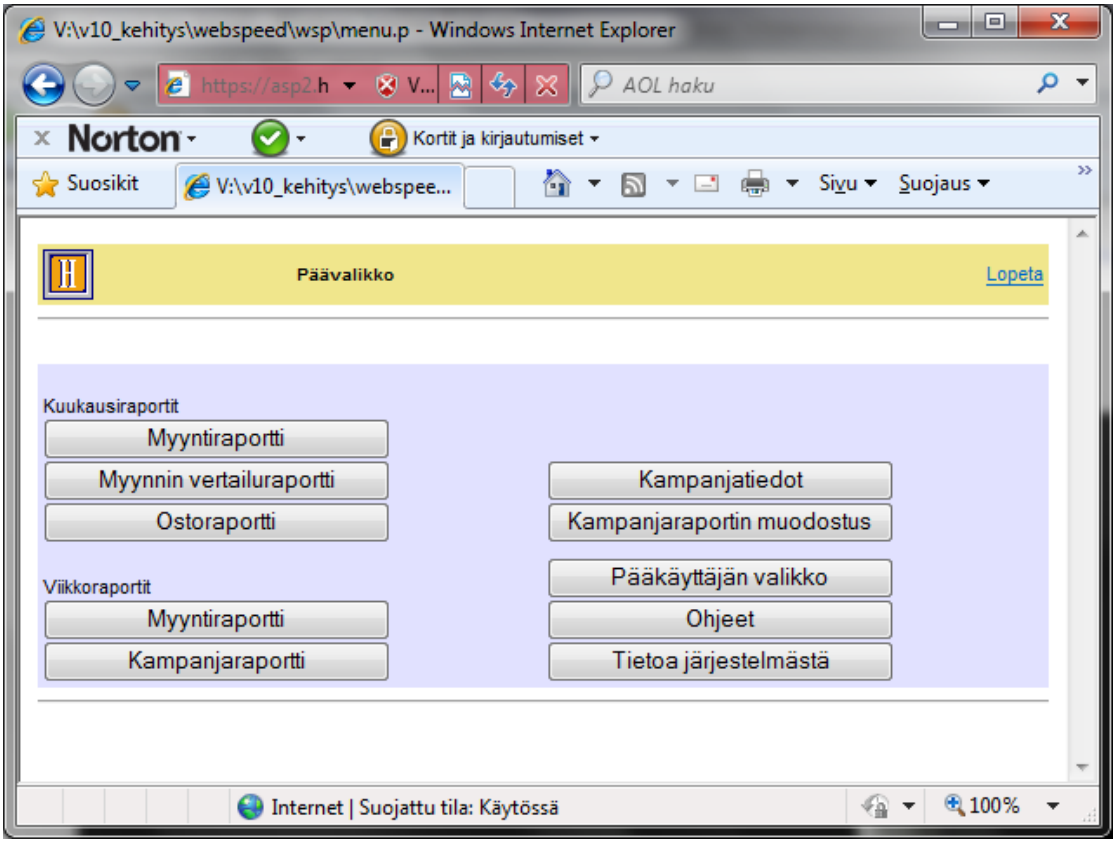
Kuva 7. Esimerkki upotetusta SpeedScriptista (WebSpeed Developer's Guide, 2001).

3.4 Selainkäyttöliittymän toteutus

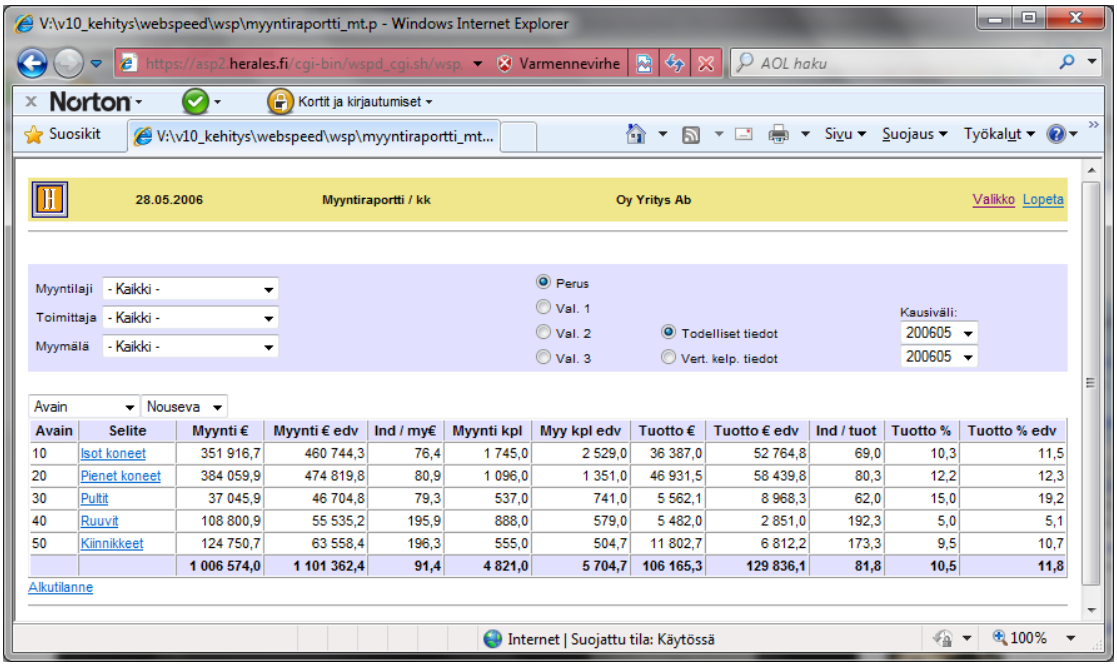
Valmis selainkäyttöliittymä sisältää neljä www-sivua: sisäänkirjautumissivun (ks. Kuva 8), Päävalikko-sivun (ks. Kuva 9), Myyntiraportti-sivun (ks. Kuva 10) sekä lopetussivun (ks. Kuva 11). Kun käyttäjä on syöttänyt oikean käyttäjätunnuksen ja salasanan sisäänkirjautumissivulle ja painanut Kirjaudu-painiketta, avautuu hänelle päävalikko, josta hän voi valita haluamansa raportin. Opinnäytetyötä varten tehty sovellus sisältää pelkän myyntiraportin, jota pääsee katselemaan painamalla Myyntiraportti-painiketta Kuukausiraportit-kohdasta. Jos käyttäjä haluaa Myyntiraportti-sivulta takaisin päävalikkoon, hän pääsee sinne painamalla ruudun oikeassa yläkulmassa olevaa Valikko-linkkiä. Jos käyttäjä haluaa lopettaa koko sovelluksen käytön, hänen tulee painaa Lopeta-linkkiä. Tällöin hänelle avautuu lopetussivu.



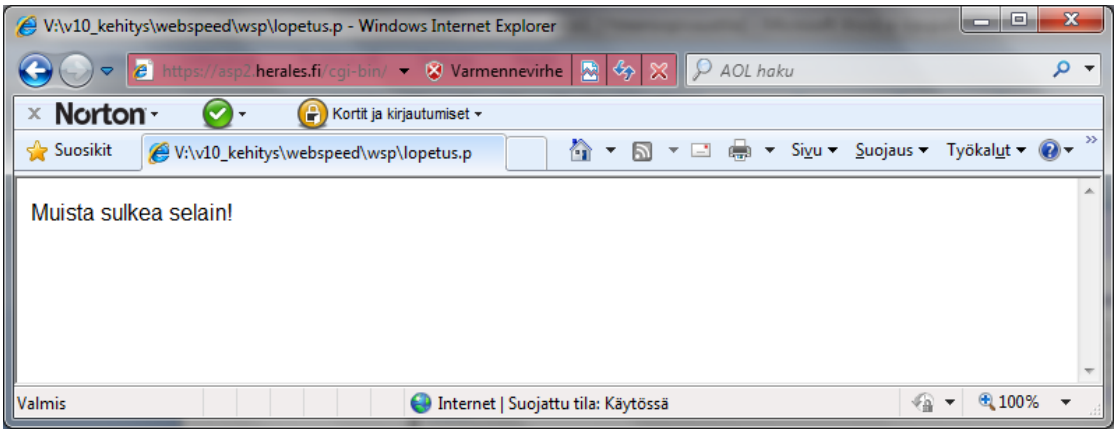
Kuva 8. Sisäänkirjautuminen.



Kuva 9. Päävalikko.



Kuva 10. Myyntiraportti.



Kuva 11. Lopetus.

3.4.1 Myyntiraportin toimintalogiikka

Myyntiraportti-sivulla on taulukko, jonka tarkoitus on näyttää tuotteiden myyntiin liittyviä tietoja. Taulukon Selite-sarakkeen linkkejä klikkaamalla päästään tuotetietojen seuraavalle tasolle; jos klikataan esim. Isot koneet-linkkiä, näytetään kaikki ko. luokkaan kuuluvat tiedot (ks. Kuva 12).

Avain | Nouseva

Avain	Selite	Myynti €	Myynti € edv	Ind / my€	Myynti kpl
110	Kärryt	10 577,8	14 035,6	75,4	59,0
112	Veneet	29 028,5	92 262,1	31,5	67,0
114	Kompressorit	111 985,0	34 625,5	323,4	86,0
115	Teollisuusimurit	14 510,6	9 358,9	155,0	25,0
120	Laiturit	27 053,3	24 358,9	111,1	219,0
130	Hitsauslaitteet	28 922,9	51 407,3	56,3	174,0
140	Mopot	50 083,8	132 082,7	37,9	142,0
150	Mönkijät	25 311,7	28 082,0	90,1	114,0
160	Trailerit	10 656,4	14 866,1	71,7	121,0
170	Uppopumput	33 339,0	36 606,5	91,1	283,0
190	Sirkkelit	4 213,6	8 450,8	49,9	3,0
900	Venetarvikkeet	2 574,6	7 979,5	32,3	336,0
910	Muut isokonetarvikkeet	3 659,5	6 628,4	55,2	116,0
		351 916,7	**460 744,3**	**76,4**	**1 745,0**

Alkutilanne

Kuva 12. Isot koneet-luokkaan kuuluvat tiedot.

Edelleen jos klikataan esim. Kärryt-linkkiä, näytetään kaikki ko. luokkaan kuuluvat tiedot (ks. Kuva 13). Tämän jälkeen porautumista voidaan jatkaa vielä seuraavalle tasolle klikkaamalla esim. Kärryt ryhmä 3-linkkiä (ks. Kuva 14). Viimeisellä tasolla näytetään valitun yksittäisen tuotteen tiedot (ks. Kuva 15). Taulukon tietoja voidaan myös lajitella; kahdesta pudotusvalikosta voidaan valita sarake, jonka mukaan tiedot lajitellaan sekä lajittelujärjestykseksi joko nouseva tai laskeva.

Avain | Nouseva

Avain	Selite	Myynti €	Myynti € edv	Ind / my€
11025	Kärryt ryhmä 2	0,0	0,0	0,0
11030	Kärryt ryhmä 3	9 610,6	11 327,9	84,8
11050	Kärryt ryhmä 4	967,2	2 707,7	35,7
		10 577,8	**14 035,6**	**75,4**

Alkutilanne

Kuva 13. Kärryt-luokkaan kuuluvat tiedot.

Avain | Nouseva

Avain	Selite	Myynti €	Myynti € edv	Ind / my€	Myynti kpl
1004	Tuote 1004	200,8	0,0	0,0	1,0
1006	Tuote 1006	454,9	1 219,7	37,3	2,0
1007	Tuote 1007	0,0	844,5	0,0	0,0
1008	Tuote 1008	0,0	294,3	0,0	0,0
1009	Tuote 1009	0,0	1 305,7	0,0	0,0
1010	Tuote 1010	179,5	233,6	76,8	1,0
1011	Tuote 1011	0,0	402,4	0,0	0,0
1012	Tuote 1012	1 800,8	4 370,2	41,2	9,0
1013	Tuote 1013	0,0	418,0	0,0	0,0
1964	Tuote 1964	213,1	686,9	31,0	1,0
2194	Tuote 2194	188,5	1 323,2	14,2	1,0
2195	Tuote 2195	333,6	229,5	145,4	2,0
2368	Tuote 2368	196,7	0,0	0,0	1,0
2456	Tuote 2456	1 449,2	0,0	0,0	9,0
2532	Tuote 2532	3 065,6	0,0	0,0	20,0
2814	Tuote 2814	792,6	0,0	0,0	4,0
2916	Tuote 2916	735,3	0,0	0,0	3,0
		9 610,6	**11 327,9**	**84,8**	**54,0**

Alkutilanne

Kuva 14. Kärryt ryhmä 3-luokkaan kuuluvat tiedot.

Avain | Nouseva

Avain	Selite	Myynti €	Myynti € edv	Ind / my€
2194	Tuote 2194	188,5	1 323,2	14,2
		188,5	**1 323,2**	**14,2**

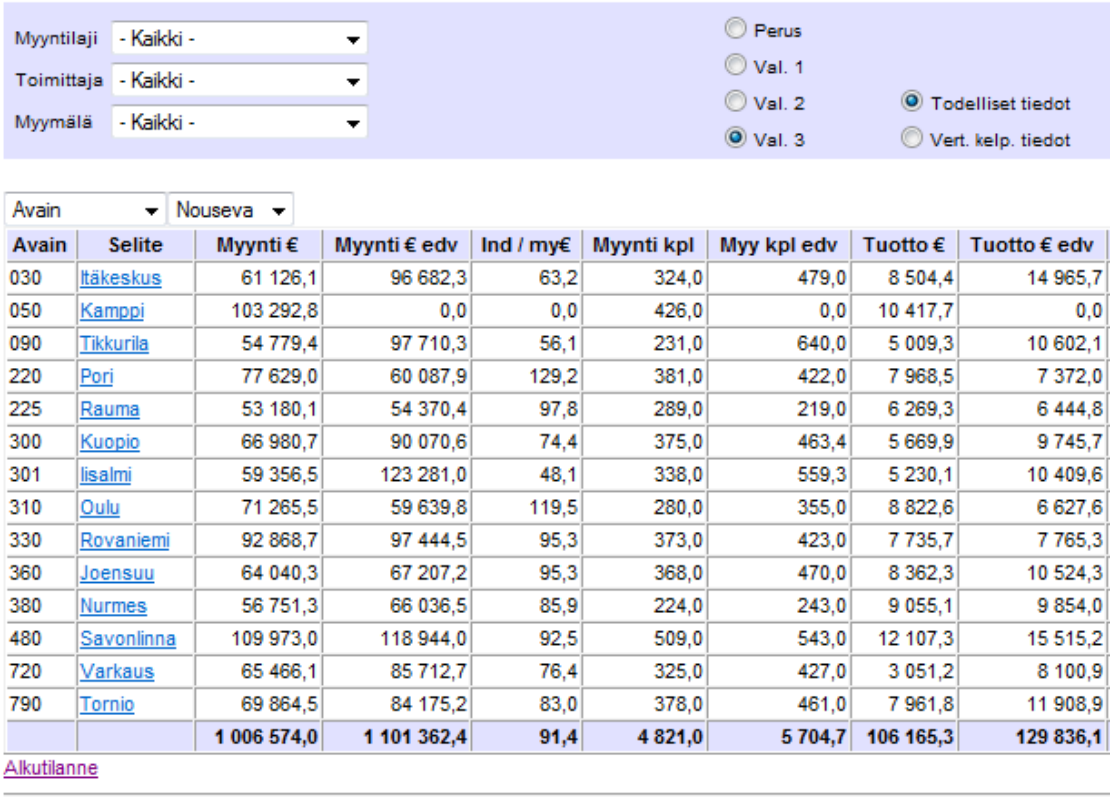
Alkutilanne

Kuva 15. Yksittäisen tuotteen tiedot.

Taulukon yläpuolella on valintapalkki (ks. Kuva 16), josta voidaan valita mitä tietoja taulukossa näytetään. Jos esim. Toimittaja-pudotusvalikosta valitaan jokin tietty toimittaja, taulukossa näytetään vain niiden tuotteiden tietoja, joita kyseisellä toimittajalla on. Jos taas esim. Myymälä-pudotusvalikosta valitaan jokin tietty myymälä, näytetään vain kyseisessä myymälässä olevien tuotteiden tietoja. Jos esim. Toimittaja-pudotusvalikosta sekä Myymälä-valikosta on molemmista valittu yksittäinen toimittaja ja myymälä, näytetään taulukossa vain niiden tuotteiden tiedot, joita on sekä valitulla toimittajalla että valitussa myymälässä.

Kuva 16. Valintapalkki.

Valintapalkissa on myös kaksi RadioButton-listaa. Ensimmäisestä voidaan valita avainnus; tämä tarkoittaa sitä, että taulukkoon saadaan haluttaessa näkyviin tiedot myyntilajin, toimittajan tai myymälän mukaan. Kun valitaan esim. Val. 3 (ks. Kuva 17), taulukossa näytetään myymälät ja niiden tiedot. Jos taulukosta tällöin valitaan esim. Kuopio, taulukossa näytetään samat tiedot kuin silloin, jos Myymälä-pudotusvalikosta olisi valittu Kuopio (ks. Kuva 18). Samalla avainnuksen RadioButton vaihtuu automaattisesti Perus-kohtaan.



Avain	Selite	Myynti €	Myynti € edv	Ind / my€	Myynti kpl	Myy kpl edv	Tuotto €	Tuotto € edv
030	Itäkeskus	61 126,1	96 682,3	63,2	324,0	479,0	8 504,4	14 965,7
050	Kamppi	103 292,8	0,0	0,0	426,0	0,0	10 417,7	0,0
090	Tikkurila	54 779,4	97 710,3	56,1	231,0	640,0	5 009,3	10 602,1
220	Pori	77 629,0	60 087,9	129,2	381,0	422,0	7 968,5	7 372,0
225	Rauma	53 180,1	54 370,4	97,8	289,0	219,0	6 269,3	6 444,8
300	Kuopio	66 980,7	90 070,6	74,4	375,0	463,4	5 669,9	9 745,7
301	Iisalmi	59 356,5	123 281,0	48,1	338,0	559,3	5 230,1	10 409,6
310	Oulu	71 265,5	59 639,8	119,5	280,0	355,0	8 822,6	6 627,6
330	Rovaniemi	92 868,7	97 444,5	95,3	373,0	423,0	7 735,7	7 765,3
360	Joensuu	64 040,3	67 207,2	95,3	368,0	470,0	8 362,3	10 524,3
380	Nurmes	56 751,3	66 036,5	85,9	224,0	243,0	9 055,1	9 854,0
480	Savonlinna	109 973,0	118 944,0	92,5	509,0	543,0	12 107,3	15 515,2
720	Varkaus	65 466,1	85 712,7	76,4	325,0	427,0	3 051,2	8 100,9
790	Tornio	69 864,5	84 175,2	83,0	378,0	461,0	7 961,8	11 908,9
		1 006 574,0	**1 101 362,4**	**91,4**	**4 821,0**	**5 704,7**	**106 165,3**	**129 836,1**

Kuva 17. Myymälöiden tiedot.

Myyntilaji - Kaikki -
Toimittaja - Kaikki -
Myymälä Kuopio

Perus
Val. 1
Val. 2
Val. 3
Todelliset tiedot
Vert. kelp. tiedot

Avain Nouseva

Avain	Selite	Myynti €	Myynti € edv	Ind / my€	Myynti kpl	Myy kpl edv	Tuotto €	Tuotto € edv
10	Isot koneet	23 920,7	37 153,8	64,4	142,0	218,0	1 817,1	4 313,1
20	Pienet koneet	22 587,4	44 009,3	51,3	66,0	120,0	2 053,3	4 580,8
30	Pultit	3 410,7	2 407,0	141,7	66,0	76,0	382,1	355,6
40	Ruuvit	6 240,2	1 986,9	314,1	54,0	33,0	217,7	193,6
50	Kiinnikkeet	10 821,8	4 513,7	239,8	47,0	16,4	1 199,8	302,6
		66 980,7	**90 070,6**	**74,4**	**375,0**	**463,4**	**5 669,9**	**9 745,7**

Alkutilanne

Kuva 18. Kuopion myymälän tiedot.

Toisesta RadioButton-listasta voidaan valita näytettäväksi joko todelliset tiedot tai vertailukelpoiset tiedot. Kausiväli-pudotusvalikoista (ks. Kuva 19) voidaan valita kausi, jonka ajalta tietoja halutaan tarkastella.

Kausiväli:
200502
200605
200502
200503
200504

Kuva 19. Kausiväli-pudotusvalikko.

3.4.2 Selainkäyttöliittymän tekninen toteutus

Selainkäyttöliittymä on ohjelmoitu WebSpeedin AppBuilderilla käyttäen CGI Wrapper-tekniikkaa (ks. Kuva 20). CGI Wrapperia käytettäessä HTML-koodi ikään kuin upotetaan Progress 4GL-koodiin {&OUT}-lauseella. Suoritettaessa CGI Wrapper Web object luo dynaamisesti sivun HTML-sisällön, joka palautetaan asiakasohjelman selaimelle (WebSpeed Developer’s Guide, 2001).

w-forcst.w

```
PROCEDURE process-web-request :

  RUN output-header.

  {&OUT}
    "<HTML>":U SKIP
    "<HEAD>":U SKIP
    "<TITLE>Sports Customer List</TITLE>":U SKIP
    "</HEAD>":U SKIP
    "<BODY BGCOLOR=~"#FFFFFF~">":U SKIP
    .

  /* Output your custom HTML to WEBSTREAM here (using {&OUT}).      */

  {&OUT}
    "<H1>Customer List</H1>":U SKIP
    "<TABLE BORDER>":U SKIP
    "<TR>":U SKIP
    "    <TH>Customer ID</TH>":U SKIP
    "    <TH>Customer Name</TH>":U SKIP
    "    <TH>Phone Number</TH>":U SKIP
    "</TR>":U SKIP
    .

  FOR EACH Customer:
    {&OUT} "<TR>":U SKIP
           "     <TD ALIGN=LEFT>":U CustNum "</TD>":U SKIP
           "     <TD ALIGN=LEFT>":U Name "</TD>":U SKIP
           "     <TD ALIGN=LEFT>":U Phone "</TD>":U SKIP
           "</TR>"
           .
  END.

  {&OUT}
    "</TABLE>":U SKIP
    "</BODY>":U SKIP
    "</HTML>":U SKIP
    .

END PROCEDURE.
```

Kuva 20. Esimerkki CGI Wrapper-tekniikasta (WebSpeed Developer’s Guide, 2001).

Testikäytössä sovelluksen tietokanta sekä web-palvelin pyörivät samalla testikoneella. Tuotannossa tietokanta tulee olemaan omalla palvelimellaan ja web-palvelin on oma palvelinkoneensa.

Kun käyttäjä on syöttänyt salasanan ja käyttäjätunnuksen, sovellus tarkastaa, ovatko ne oikeat. Jos tunnukset ovat oikeat, tallennetaan istunnon tietoihin käyttäjätunnus ja käyttöoikeustaso sekä aikaleima. Istunnolle on määritelty pituus (tässä tapauksessa 3000000 sekuntia), jonka päättymisen jälkeen istunto katkeaa. Jokaiseen ohjelmaan lisätään ”RUN pwritesession.”-käsky, jolla istunnon tiedot kirjoitetaan levylle. ”RUN preadsession” puolestaan tarkastaa istunnon tietojen oikeellisuuden.

Menu.p-ohjelmaan on tehty aliohjelma prunp, joka kääntää .p-tiedostot (kääntämätön versio) .r-tiedostoiksi (käännetty versio). Kehitysympäristössä voidaan ajaa molempia versioita, mutta kun sovellus laitetaan tuotantoon, vain .r-tiedostoja ajetaan. Muualla sovelluksessa, kun .p-tiedostoja on kutsuttu, Progress osaa itse ajaa käännetyn version (.r), jos sellainen löytyy. Näin ollen prunp-aliohjelman käytöstä on hyötyä vain siten, että sitä käyttämällä voidaan virhekäsittely tehdä halutulla tavalla. Jos aliohjelmaa ei käytetä, saadaan virhetilanteessa Progressin oletusvirheilmoitus. Jatkokehitystä ajatellen voitaisiin prunp-aliohjelma lisätä sovelluksen kaikkiin sellaisiin kohtiin, joissa .p-tiedostoja kutsutaan.

Myyntiraportissa käyttäjän tekemät valinnat kulkevat URL-osoitteen muuttujissa, eli niitä ei tallenneta levylle. Set-top-value-funktio vie url-osoitteessa siirtyvät arvot ajonaikaiseen väliaikaistauluun, josta ne voidaan helposti hakea get-top-value-funktiolla. URL-osoitteeseen liittyy kuitenkin seuraavanlainen ongelma: URL-osoite on aina yhden pykälän myöhässä eikä tätä ongelmaa saatu ratkaistua. Asialla on merkitystä lähinnä silloin, kun käyttäjä haluaa tallentaa jonkin tietyn tuotteen tai tuoteryhmän tiedot; hän ei voi siis tallentaa URL-osoitetta sellaisenaan, koska se tallentaa raporttinäytöllä yhtä sivua aikaisemmin olleet tiedot. Jos käyttäjä haluaa esim. tallentaa ruuvien tiedot toimittajien mukaan ja ottaa URL-osoitteen talteen (ks. Kuva 21) ja sen jälkeen liittää em. URL-osoitteen osoiteriville, näytölle tulevat pykälää aiemmin olleet tiedot, eli tässä tapauksessa ruuvit ilman toimittajien tietoja (ks. Kuva 22). Jatkokehitystä ajatellen tämän ongelman korjaaminen on melko olennainen asia.

28.05.2006 Myyntiraportti / kk

Myyntilaji - Kaikki -
Toimittaja - Kaikki -
Myymälä - Kaikki -

Perus
Val. 1
Val. 2
Val. 3

Avain Nouseva

Avain	Selite	Myynti €	Myynti € edv	Ind / my€	Myynti kpl	Myy k
1410	Maamerkki Oy	200,0	950,8	21,0	3,0	
92020	Kajalservice Oy	2 022,1	0,0	0,0	13,0	
99100	Bronson Oy	106 578,7	54 584,4	195,3	872,0	
		108 800,9	**55 535,2**	**195,9**	**888,0**	

Alkutilanne

Kuva 21. Ruuvit toimittajien mukaan.

28.05.2006 Myyntiraportti / kk

Myyntilaji - Kaikki -
Toimittaja - Kaikki -
Myymälä - Kaikki -

Perus
Val. 1
Val. 2
Val. 3

Avain Nouseva

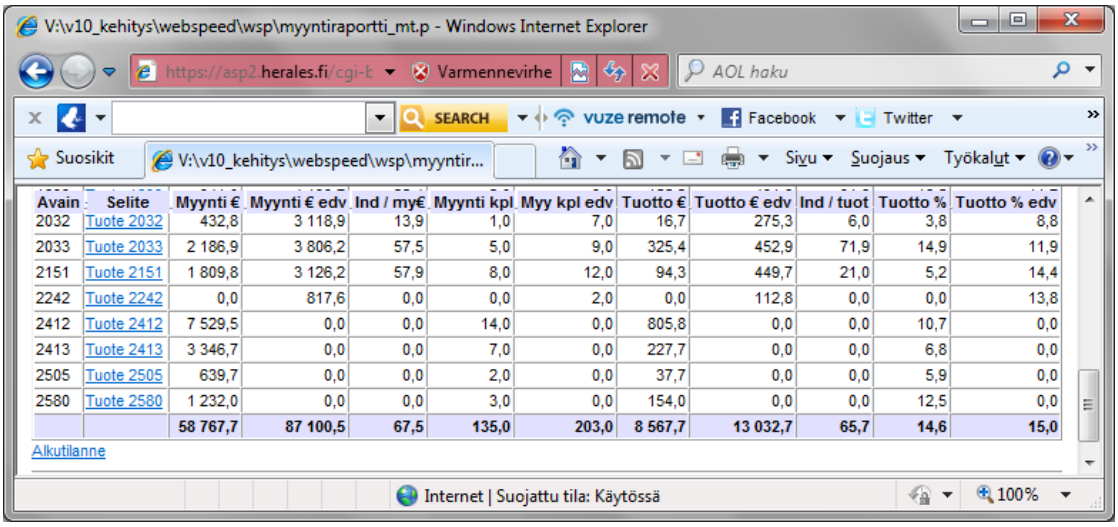
Avain	Selite	Myynti €	Myynti € edv	Ind / my€	Myynti kpl	Myy k
410	Puuruuvit	103 847,6	48 310,5	215,0	693,0	
450	Peltiruuvit	200,0	950,8	21,0	3,0	
940	Ruuvien varaosat	4 753,2	6 273,9	75,8	192,0	
		108 800,9	**55 535,2**	**195,9**	**888,0**	

Alkutilanne

Kuva 22. Ruuvit ilman toimittajien tietoja.

Laske-calc-funktio laskee myyntiraportin sarakkeisiin halutut arvot laskentakaavojen mukaan. Laskentakaavat voidaan sopia jokaiselle asiakkaalle erikseen. Laskentakaavoissa käytetyt luvut saadaan asiakkaiden kassajärjestelmistä joka yö.

Myyntiraportti-sivun taulukon otsikkorivi on tehty siten, että jos tuotteita tai tuoteryhmiä on niin paljon, etteivät kaikki niiden tiedot mahdu näkymään näytöllä yhtä aikaa, otsikkorivi liukuu alaspäin siten, että se on koko ajan näkyvissä (ks. Kuva 23). Ominaisuus on toteutettu JavaScript-funktiolla TableFloaterTitle.js (ks. Liite 1), joka on löydetty sivustolta http://stansight.com/.



Avain	Selite	Myynti €	Myynti € edv	Ind / my€	Myynti kpl	Myy kpl edv	Tuotto €	Tuotto € edv	Ind / tuot	Tuotto %	Tuotto % edv
2032	Tuote 2032	432,8	3 118,9	13,9	1,0	7,0	16,7	275,3	6,0	3,8	8,8
2033	Tuote 2033	2 186,9	3 806,2	57,5	5,0	9,0	325,4	452,9	71,9	14,9	11,9
2151	Tuote 2151	1 809,8	3 126,2	57,9	8,0	12,0	94,3	449,7	21,0	5,2	14,4
2242	Tuote 2242	0,0	817,6	0,0	0,0	2,0	0,0	112,8	0,0	0,0	13,8
2412	Tuote 2412	7 529,5	0,0	0,0	14,0	0,0	805,8	0,0	0,0	10,7	0,0
2413	Tuote 2413	3 346,7	0,0	0,0	7,0	0,0	227,7	0,0	0,0	6,8	0,0
2505	Tuote 2505	639,7	0,0	0,0	2,0	0,0	37,7	0,0	0,0	5,9	0,0
2580	Tuote 2580	1 232,0	0,0	0,0	3,0	0,0	154,0	0,0	0,0	12,5	0,0
		58 767,7	**87 100,5**	**67,5**	**135,0**	**203,0**	**8 567,7**	**13 032,7**	**65,7**	**14,6**	**15,0**

Kuva 23. Liukuva otsikkorivi.

4 POHDINTA

Selainkäyttöliittymän valinta ja toteutus Herales Raportointiin oli melko haasteellinen projekti. Minulla ei ollut selainpohjaisten sovellusten ohjelmoinnista paljoakaan aikaisempaa kokemusta, joten erityisesti projektin alussa tämä tuotti vaikeuksia; web-ohjelmointi nimittäin poikkeaa jonkin verran ns. tavallisesta ohjelmoinnista. Jokin asia, joka on helppo toteuttaa esim. C#:lla ohjelmoituun perinteiseen Windows-ohjelmaan, tuntui aluksi lähes mahdottomalta toteuttaa web-ohjelmoinnin tekniikoilla. Kun opin ymmärtämään paremmin selainympäristön toimintaa ja web-teknologioita, selainkäyttöliittymän ohjelmointi ei enää tuntunut niin vaikealta. HTML-kieli ja CSS-tyylimäärittelyt olivat minulle tuttuja, joten niiden käsitteleminen ei tuottanut vaikeuksia.

Siinä vaiheessa kun selvitin ASP.NET:in ja Web-palveluiden soveltuvuutta selainkäyttöliittymän toteuttamiseen, projekti jäi pitkäksi aikaa junnaamaan paikoilleen. Vaikka tein kaikkeni Herales Oy:n Web-palveluiden toimimiseksi ASP.NET-ympäristössä, en saanut mitään tuloksia aikaan. Tämä tuntui erityisen turhauttavalta, vaikka näin jälkeenpäin mietittynä tästäkin selvitystyöstä oli myös paljon hyötyä: Web-palvelut ja niiden toimintaperiaatteet tulivat tutuiksi.

Myöskään Progress 4GL-ohjelmointikieli ei ollut minulle lainkaan tuttu, joten senkin ymmärtäminen vei oman aikansa. Siinä vaiheessa, kun selainkäyttöliittymän toteuttamisessa siirryttiin käyttämään WebSpeedia, pidettiin koko projektiryhmälle yhteinen koulutus WebSpeedin perusteista ja sen käyttämisestä selainkäyttöliittymän ohjelmointiin. Tästä johtuen ohjelmointi WebSpeed-teknologialla ei tuntunut ylitsepääsemättömän vaikealta; lisäksi muilta projektiryhmän jäseniltä sai tarvittaessa apua.

Opinnäytetyöraportin kirjoittamisessa oli omat vaikeutensa. Suuri osa lähdemateriaalista oli englanninkielistä ja sisälsi paljon sellaista tekstiä, jonka kääntäminen suomeksi oli hankalaa. Ennen kirjoittamisen aloittamista jouduin miettimään mistä näkökulmasta raportin kirjoitan. Aluksi ajattelin jättää ASP.NET:in ja Web-palveluiden käsittelyn lähes kokonaan pois ja keskittyä kuvaamaan pelkästään valmista WebSpeed-teknologialla toteutettua sovellusta. Tulin kuitenkin siihen

tulokseen, että olisi hyvä käydä läpi sovelluksen koko kehityskaari ja selittää, miksi juuri WebSpeedista tuli selainkäyttöliittymän lopullinen toteutusväline. Opinnäytetyöraportin kirjoittamista vaikeutti myös se, että selainkäyttöliittymän koko ohjelmakoodi oli määrätty salassa pidettäväksi; näin ollen en voinut sisällyttää raporttiin edes pätkiä ohjelmakoodista. Selainkäyttöliittymän koko ohjelmakoodin lähetin luettavaksi pelkästään opinnäytetyön ohjaajalle.

Vaikka opinnäytetyön tekeminen oli haasteellista, olen tyytyväinen siihen, että tein opinnäytetyöni sellaisesta aiheesta joka ei ollut minulle kovinkaan tuttu ja jota varten joutui opiskelemaan paljon uusia asioita. Nykymaailmassa web-ohjelmointitaito on melko tärkeää, sillä yhä useampi sovellus toimii selainkäyttöliittymän kautta. Toivon, että tämän opinnäytetyön tekemisestä tulee olemaan hyötyä työelämässä.

Selainkäyttöliittymä saatiin toteutettua siltä osin kuin projektin alussa oli sovittu. Sovellus toimii kaikilta osin ilman mitään kohtuuttoman suuria puutteita. Ainut ongelma sovelluksessa on se, että myyntiraportin URL-osoite on aina yhden pykälän myöhässä. Tähän ongelmaan on varmasti jokin ratkaisu, mutta ongelma jää toistaiseksi ratkaisematta. Herales Oy pystyy jatkamaan sovelluksen kehittämistä tässä opinnäytetyössä tehdyn esimerkkisovelluksen pohjalta ja em. ongelma tulee toivottavasti tulevaisuudessa ratkeamaan.

LÄHTEET

Herales Oy 2010
Verkkodokumentti. Luettu 8.3.2010.
http://www.herales.fi

Knuutila, T. 2001
Verkkodokumentti. Luettu 24.3.2010.
Ohjelmointikielten periaatteet.
http://guardian.cs.utu.fi/knuutila/Courses/okp/default.html

Sadd, J. 2003
OpenEdge Development: Progress 4GL Handbook. Progress Software Corporation.

Vesanen, A. 2005
Verkkodokumentti. Luettu 24.3.2010.
Ohjelmointikielten periaatteet. Ohjelmointikielten historiaa.
http://www.tol.oulu.fi/kurssit/okp/Luennot/Kalvot/OKP_Historia.pdf

Webspeed Developer’s Guide 2001
Progress Software Corporation.

W3C 2010
Verkkodokumentti. Luettu 10.3.2010.
Web Services Architecture.
http://www.w3.org/TR/ws-arch

W3Schools 2010
Verkkodokumentti. Luettu 10.3.2010.
Web Services Tutorial.
http://www.w3schools.com/webservices

LIITE 1 TableFloaterTitle.js

```
//
*****************************************************************************
*
// ** TableFloaterTitle.js
// **
// ** Unobtrusive JavaScript function which will keep the titles of a table
on
// ** screen as the user scrolls down.  The titles will 'float' keeping in
view
// ** as long as any part of the table remains in view.
// **
// ** Author: Stan Slaughter
// ** Web: http://www.StanSight.Com/
//
*****************************************************************************
*

//////////////////////////////////////////////////////////////////////////////
//
// startFloatTitle - Start tables floating
//
//  This function will scan the page looking for every "table" on it.
//  Any table which has a class of "FloatTitle" will have its ID value
//  passed on to the function for the actual float process
//////////////////////////////////////////////////////////////////////////////
//

function startFloatTitle() {

    var cnt = 0;
    var allTableIDs = new Array();
    allTables = document.getElementsByTagName("table");

    // Scan page for all tables with a class of "FloatTitle"
    for (i = 0; i < allTables.length; i++) {
        if (allTables[i].className == "FloatTitle") {

            // Get the ID. If no ID exists then assign one
            tableID = allTables[i].id;
            if (tableID == "") {
                tableID = "tmpFloatTitleTableId" + cnt;
                allTables[i].id = tableID;
            }

            // Save ID's
            allTableIDs[cnt] = tableID;
            cnt++;
        }
    }

    // Start floating the tables title
    for (i = 0; i < allTableIDs.length; i++) {
        floatTitle(allTableIDs[i], i);
    }
}

//////////////////////////////////////////////////////////////////////////////
//
// floatTitle - Start the float process
//  @param tableID - ID of table that floating title will be created for
//  @param cnt - Number representing table - for tracking seperate float
events
//
//  This function calls the functions which create the floating title and
//  start the floating process
//////////////////////////////////////////////////////////////////////////////
//

var floatTitleTimer = new Array();
function floatTitle(tableID, cnt) {

    // Stop any old Loops
    if (typeof (floatTitleTimer[cnt]) == 'undefined') floatTitleTimer[cnt] =
0;
    clearTimeout(floatTitleTimer[cnt]);
```

```
    // Create title object then start float loop
    titleID = createTitleObj(tableID);
    floatTitleLoop(tableID, titleID, cnt);

}

////////////////////////////////////////////////////////////////////////////
//
// createTitleObj - Create the floating object for this table
//  @param tableID - ID of table that floating title will be created for
//  @return titleWrapperID - ID of div which contains a copy of tables THEAD
//
//  Clone the THEAD from the table, create a new table to place it in then
//  create a DIV wrapper to place it in. The ID for the DIV is returned to
//  the calling function so it can be used to position the object in the
//  floating function.
////////////////////////////////////////////////////////////////////////////
//

function createTitleObj(tableID) {

    var titleWrapperID = tableID + "Title";
    var titleTableID = tableID + "TitleTable";

    // If Title has not been created yet
    if (document.getElementById(titleWrapperID) == null) {

        // "clone" the Table's thead
        tableObj = document.getElementById(tableID);
        clonedtHead = tableObj.tHead.cloneNode(true);

        // Create wrapper div
        titleWrapperObj = document.createElement("div");
        titleWrapperObj.setAttribute("id", titleWrapperID);

        // Create the Title table
        TitleTable = document.createElement("table");
        TitleTable.setAttribute("id", titleTableID);

        // Append Cloned thead to the Title table
        TitleTable.appendChild(clonedtHead);

        // Append table to div container
        titleWrapperObj.appendChild(TitleTable);

        // Insert wrapper div into the DOM before Table
        parentDiv = tableObj.parentNode;
        parentDiv.insertBefore(titleWrapperObj, tableObj);

        // Initially position container
        titleWrapperObj.style.position = "absolute";
        titleWrapperObj.style.top = "0px";
        titleWrapperObj.style.zIndex = "1";
    }

    tableObj = document.getElementById(tableID);
    titleTableObj = document.getElementById(titleTableID);

    // Set Title width to be the same as Table
    titleWrapperObj = document.getElementById(titleWrapperID);
    titleWrapperObj.style.width = tableObj.offsetWidth + 'px';

    // Set widths of Title TD's to same as Table TD's
    tableCells = tableObj.tHead.rows[0].cells;
    titleTableCells = titleTableObj.tHead.rows[0].cells;

    for (var i = 0; i != tableCells.length; i++) {
        titleTableCells[i].style.width = (tableCells[i].clientWidth) + 'px';
        titleTableCells[i].style.cursor = 'normal';
    }

    return titleWrapperID;
}

////////////////////////////////////////////////////////////////////////////
//
// floatTitleLoop - Float the titles up and down the table as the page
scrolls
//  @param tableID - ID of table that wants floating title
```

```
//  @param titleID - ID of div that will float (contains copy of tableID's
THEAD)
//  @param cnt - Number representing table - for tracking seperate float
events
//
//  Keep the title object in view as the table postion changes as a user
//  scrolls up and down in the window
//////////////////////////////////////////////////////////////////////////
//

function floatTitleLoop(tableID, titleID, cnt) {

    // If Table and Title objects exist
    if (document.getElementById(tableID) != null &&
document.getElementById(titleID) != null) {

        // Set value to be number of pixels from screen top that you wish
        // scrolling to start at. 0=top, 10=10 pixels down from top, etc..
        // Useful for those who happen to have top screen banners
        var startOffsetTop = 0;

        // Get start and stop float positions from table
        var tableObj = document.getElementById(tableID);
        var tablePos = FindPosition(tableObj);
        var startTop = tablePos[1] - startOffsetTop;
        var endTop = startTop + tableObj.clientHeight;

        // Get new positon of body scroll top and keep it in bounds
        var newTop = (document.documentElement.scrollTop > 0) ?
document.documentElement.scrollTop : document.body.scrollTop;
        if (newTop < startTop) newTop = startTop;
        if (newTop > endTop) newTop = endTop;

        // Move the title to new postion
        var titleObj = document.getElementById(titleID);
        if (newTop > startTop && newTop < endTop) {

            // Display "Title" if it is not all ready being displayed
            if (titleObj.style.display != 'block') titleObj.style.display =
'block';

            // Apply offset to get newTop position
            newTop = newTop + startOffsetTop;

            // Apply Ease-In effect
            easeInWanted = true;
            if (easeInWanted) {

                // Get current location and get the difference from where
it's
                // at to where you want it to go
                oldTop = parseInt(titleObj.style.top);
                topDiff = newTop - oldTop;

                // Calaculate how far you want to move it - then set that to
new postion
                moveTop = 0;
                if ((topDiff < (-1)) || (topDiff > (1))) { moveTop =
Math.round(topDiff / 3); }
                newTop = oldTop + moveTop;
            }

            // Move to new top position
            if (newTop < 0) newTop = 0;
            titleObj.style.top = newTop + "px";
        } else {
            // Else hide "Title" if it is not all ready hidden
            if (titleObj.style.display != 'none') {
                titleObj.style.display = 'none';
                titleObj.style.top = "0px";
                titleObj.style.zIndex = "1";
            }
        }

    }

    // Execute this function again in 40 milliseconds (thousandths of a
second)
    cmd = "floatTitleLoop('" + tableID + "','" + titleID + "','" + cnt +
"')";
```

```
    floatTitleTimer[cnt] = window.setTimeout(cmd, 40);

}

////////////////////////////////////////////////////////////////////////////
//
// FindPosition - find the Top and Left postion of an object on the page
//  @param obj - object of element whose position needs to be found
//  @return array - Array whoose first eleemnt is the left postion and
whoose
//                  second is the top position
////////////////////////////////////////////////////////////////////////////
//

function FindPosition(obj) {
    // Figure out where the obj object is in the page by adding
    // up all the offsets for all the containing parent objects
    if (obj == null) return [0, 0];

    // Assign the obj object to a temp variable
    tmpObj = obj;

    // Get the offsets for the current object
    var obj_left = tmpObj.offsetLeft;
    var obj_top = tmpObj.offsetTop;

    // If the current object has a parent (ie contained in a table, div,
etc..)
    if (tmpObj.offsetParent) {

        // Loop through all the parents and add up their offsets
        // The while loop will end when no more parents exist and a null is
returned
        while (tmpObj = tmpObj.offsetParent) {
            obj_left += tmpObj.offsetLeft;
            obj_top += tmpObj.offsetTop;
        }
    }
    return [obj_left, obj_top];
}

////////////////////////////////////////////////////////////////////////////
//
// Start floating titles after window finishes loading
////////////////////////////////////////////////////////////////////////////
//

window.onload = startFloatTitle;

////////////////////////////////////////////////////////////////////////////
//
// Start floating titles when window is resized
////////////////////////////////////////////////////////////////////////////
//

window.onresize = startFloatTitle;
```