Jussi Mäkinen ja Miika Nurminen

Kiuru-sovelluksen jatkokehitys

Tietotekniikan Erikoistyön raportti
26. elokuuta 2003

Jyväskylän yliopisto

Tietotekniikan laitos

Tekijä: Jussi Mäkinen ja Miika Nurminen
Yhteystiedot: jusmaki@cc.jyu.fi ja minurmin@cc.jyu.fi
Työn nimi: Kiuru-sovelluksen jatkokehitys
Title in English: Additional development of Kiuru
Työ: Tietotekniikan Erikoistyön raportti
Sivumäärä: 25
Tiivistelmä: Erikoistyössä jatkokehitettiin Kiuru-sovellusprojektin sovellusta. Uusia ominaisuuksia lisättiin, virheitä korjattiin ja sovelluksen rakennetta selkeytettiin.
English abstract: The Kiuru-application was developed as a special assignment. New features were added, errors were fixed and application structure was clarified.
Avainsanat: Salivaraus, kurssien opetustapahtumat, resurssivaraus, WWW-sovellus, Korppi-järjestelmä, jatkokehitys
Keywords: Room reservations, teaching events, resource reservations, WWW application, Korppi system

Sisältö

1 Johdanto

Kiuru-projekti toteutti syksyllä 2002 Jyväskylän yliopiston tietotekniikan laitoksella kehitettyyn Korppi-järjestelmään WWW-pohjaisen tilanvarausosion. Projektiryhmästä Jussi Mäkinen ja Miika Nurminen jatkokehittivät sovellusta tietotekniikan Erikoistyönä.

Yliopisto luopui Timmi-järjestelmän käytöstä salivarauksissa tammikuussa 2003. Raportin kirjoitushetkellä salivarauksiin käytetään edelleen vanhaa Sali-järjestelmää. Kiuru-sovellus on toiminnallisesti lähellä Sali-järjestelmää ja pystyy periaatteessa korvaamaan sen. Ennen Kiuru-sovelluksen käyttöön siirtymistä sovellusta täytyy kuitenkin parantaa käytettävyydeltään sekä lisätä uusia haku- ja raportointitoimintoja.

Erikoistyössä sovelluksesta korjattiin sovellusprojektin aikana havaitut virheet ja toteutettiin joitakin uusia ominaisuuksia. Lisäksi sovelluksen ja koodin rakennetta muokattiin käytännöllisempään suuntaan. Erikoistyöraportissa kuvataan erikoistyön aikana sovellukseen toteutettuja ominaisuuksia.

Erikoistyötä suoritettiin vuoden 2003 helmikuun alusta toukokuun loppuun. Tietotekniikan laitos antoi työhuoneeksi huoneen Ag C225.1, jonne siirrettiin kaksi Kiuru-projektin käyttämää Linux-pohjaista tietokonetta.

Luvussa 2 käsitellään JSP-sivuille tehtyjä muutoksia ja luvussa 3 käsitellään Java-pavut. Luvussa 4 käsitellään muita erikoistyössä suoritettuja tehtäviä ja määrityksiä.

2 JSP-sivuille tehdyt muutokset

Erikoistyön alussa määriteltiin ominaisuuksia, jotka tulisi toteuttaa käyttöliittymään erikoistyön aikana. Tässä kuvataan JSP-sivuille tehdyt muutokset.

2.1 addAppointment.jsp ja modifyGroup.jsp

Vapaan salin haku varausprosessin aikana muutettiin toimimaan siten, että käyttäjä antaa ensin kaikki tapahtumaan liittyvät tiedot. Jos käyttäjä painaa tietojen syöttämisen jälkeen *Tee varauspyyntö*-painiketta, prosessissa siirrytään hakemaan vapaata salia tai tekemään suoraan varausta/varauspyyntöä.

Hae sali-painike poistettiin sivuilta `addAppointment.jsp` ja `modifyGroup.jsp`. Salin valintalistaan lisättiin kohta *Hae sali*. Näin sovellus saatiin toteutukseltaan yksinkertaisemmaksi ja varmemmin toimivaksi.

2.2 timeGrid.jsp

Opettajan käyttöliittymän *kurssin ajat ja paikat* -osion *Yksittäiset tiedot* -sivu muutettiin toimimaan kuten muutkin salivaraussivut. Sivun tapahtumille tehdyt muutokset päivittyvät varausjärjestelmään, jos tapahtumat ovat salivarauksia. Tapahtumien valintalistaan lisättiin mm. *Hae sali*-valinta.

2.3 Salihaku ja osittain vapaiden salien näyttäminen

Jatkokehityksen tuloksena käyttäjä pystyy tekemään kalenterin kautta uuden salivarauksen ja lopuksi etsimään vapaan salin tapahtumalle. Salivarauksen ominaisuuksia voi rajoitetusti muokata myös jälkikäteen, mutta esimerkiksi varaukselle merkittyä aikaa ei voi kasvattaa.

Jos käyttäjä on tekemässä varausta usealle viikolle tai päivälle, käyttäjä voi halutessaan etsiä myös osittain vapaita saleja. Nämä ovat sellaisia saleja, jotka eivät ole vapaina kaikkina toivottuina ajankohtina, mutta jotka ovat vapaita jollakin kerralla. Käyttäjä saa eteensä radiopainikelistan, josta hän voi näppärästi valita salit tapahtumilleen.

Kaikissa saleihin kohdistettavissa hauissa voidaan valita, kohdistetaanko haku vain virallisiin saleihin vai kaikkiin saleihin. Sali on virallinen, jos sille on määritelty vastuuorganisaatio. Varauksen teon yhteydessä myös muut hakurajoitteet ovat mahdollisia.

2.4 Yhteyshenkilön valinta kalenterivarauksessa

Käyttäjä pääsee määrittelemään varaajan yhteyshenkilöksi tai organisaatioksi muita henkilöitä tai organisaatioita valitsemalla `addAppointment.jsp`-sivulla *Anna kaikki tiedot*-valintaruudun. `giveAllInformation.jsp`-sivulla käyttäjä valitsee organisaation ja yhteyshenkilön.

Käyttäjä voi lisätä myös uuden yhteyshenkilön, jolla ei välttämättä ole tunnusta Korppi-järjestelmään. Jatkokehitystehtävänä yhteyshenkilön tulisi voida liittää tapahtuma johonkin kurssiin opetustapahtumaksi.

Tuntemattomia yhteyshenkilöitä käsitellään tarkemmin luvussa 4.8.

2.5 Varausten selailu ja raportointi henkilöiden mukaan

Aikaisemmin käyttäjät olivat pystyneet katsomaan omia varauksiaan listana. Erikoistyössä lisättiin sivulle `myReservations.jsp` valinnainen parametri `person`, jonka avulla saadaan katsottua toisen henkilön varauksia. `Person`-parametriin annetaan tarkasteltavan henkilön `personID`. Varaukset ovat julkisia, joten tässä ei ole tarkistuksia henkilöiden eikä käyttäjäryhmien suhteen.

2.6 Päällekkäisten varausten näyttäminen

`showReservationRequests.jsp`-sivulle lisättiin päällekkäisten varausten näyttäminen. Taulukkoon lisättiin sarake, joka näyttää päällekkäisten varausten lukumäärän rivillä olevaan varaukseen nähden. Lukumäärälinkistä painamalla käyttäjä näkee valitun varauksen kanssa päällekkäiset varaukset.

2.7 Syötteiden syntaktinen tarkistus

`Reservation`-hakemiston sivut tutkivat niille annettuja syötteitä ja varmistavat, ettei sovellus kaadu väärän tyyppiseen syötteeseen. Näitä ovat esimerkiksi tyhjät nimet tai merkkijonot numerokentissä. Saleja ja saliryhmiä editoitaessa virheelliset syötteet johtavat virheilmoitukseen ja muokkaussivulle paluuseen.

2.8 Ilmoitustaulu- ja sähköpostiviestien lähetys

Käyttäjä pystyy omista asetuksistaan (löytyvät hakupuusta kohdan *Salivaraukset* alta) määrittämään, saako hän tietoa varauksien hyväksymisestä tai hylkäyksestä Korppi-järjestelmän ilmoitustaulun kautta. Kun varauksien vahvistaja vahvistaa tai hylkää varauksen, jossa käyttäjä on yhteyshenkilönä, Korpin ilmoitustaululle tulee uusi viesti, jossa kerrotaan varauksen vahvistamisesta tai hylkäyksestä. Viestin linkistä käyttäjä pääsee katsomaan tapahtumaa kalenterin kautta.

Käyttäjä saa halutessaan myös sähköpostiviestin varauksien vahvistamisesta tai hyväksymisestä.

2.9 Monikielisyys

`Reservation`-hakemiston JSP-sivuihin lisättiin teksteihin käännös käyttäjän käyttämän kielen mukaan. Näin myös englanninkieliset käyttäjät pääsevät tekemään salivarauksia. Käännös hoidetaan `user`-olion metodilla `T`.

Monikielistäminen vaatii yhtenäistämistä nykyiseen käytäntöön nähden. Osa teksteistä suomennetaan suoraan JSP-sivulla, osa Java-pavuissa. Käännösten toimivuuden laajamittainen testaus ei ollut mahdollista, koska huomattavaa osaa tekstien käännöksistä ei ole kirjoitettu.

2.10 JavaScript-koodi

JavaScript-koodien suhteen sovellettiin samoja periaatteita kuin Korppi-järjestelmän aiemmassa kehityksessä. Sivut toimivat tarvittaessa ilman JavaScriptia, mutta koodia käytettiin pienessä mittakaavassa sivujen yleisen käytettävyyden parantamiseksi.

Yleiskäyttöiset JavaScript-koodipätkät siirrettiin JSP-sivuilta `shared`-hakemistossa sijaitsevaan `scripts.js`-tiedostoon. Tiedosto liitetään jokaiselle JSP-sivulle. `HtmlBean`-

papuun lisättiin metodeja, jotka helpottavat JavaScript-koodin lisäystä sivuille. Merkittävin jaetussa käytössä oleva JavaScript-koodi liittyi kaikkien HTML-taulukossa olevien valintaruutujen valitsemiseen kerralla.

Koodien toiminta testattiin selaimilla Mozilla 1.1, Netscape 6, Opera 6 ja Internet Explorer 5. Testaus suoritettiin Red Hat Linux- ja Windows 2000 -käyttöjärjestelmissä.

2.11 Pikkukorjauksia

Erikoistyön aikana tehtiin myös runsaasti pieniä virhekorjauksia. Eräs eniten virheitä sisältänyt yksittäinen metodi oli salin liittäminen henkilön henkilökohtaiseen saliryhmään. Lisäksi salien ja saliryhmien hallintaan liittyville sivuille tehtiin korjauksia.

Korjausten lisäksi sivuille lisättiin myös pieniä käyttäjiä helpottavia ominaisuuksia. Esim. taulukkoja sisältäville sivuille lisättiin mahdollisuus järjestää tiedot minkä tahansa taulukon sarakkeen mukaan. Eri sivujen käyttöliittymiä yhtenäistettiin taulukkojen ja otsikoiden muotoilun sekä virheiden näyttämisen osalta.

3 Java-papuihin tehdyt muutokset

Sovellusprojektin aikana toteutettuja Java-papuja korjailtiin. Lisäksi luotiin uusia papuja. Tärkeinpänä muutoksena Java-papuihin nojaavaa JSP-sivujen käsittelylogiikkaa kehitettiin edelleen ja yhtenäistettiin. Tämä vaati `KiuruHandler`-luokan ja aliluokkien osittaista uudelleenmäärittelyä. Luvussa käsitellään vain erikoistyön aikana tehtyjä lisäyksiä ja muutoksia. Lisätietoja `kiurubeans`-paketin pavuista on Kiuru-projektin sovellusraportissa [2].

Kaikkien `kiurubeans`-paketin Java-papujen koodi dokumentointiin Javadoc-työkalun käyttämän kommentointityylin mukaisesti. Pavut on suunniteltu yleiskäyttöisiksi, joten ainakin osaa niistä voinee käyttää myös Korppi-järjestelmän jatkokehityksessä.

3.1 kotkabeans-pakettiin tehdyt muutokset

`kotkabeans`-paketti pyrittiin pitämään mahdollisimman muuttumattomana, koska paketin pavut ovat koko järjestelmän käytössä. Olemassaolevia metodeja ei muutettu, mutta luokkiin lisättiin muutamia uusia metodeja.

3.1.1 Encoder

`Encoder`-papua käytetään koodaamaan käyttäjältä kysyttäviä merkkijonoja. Esimerkiksi SQL-lauseisiin menevissä tiedoissa ei saa olla lauseen rakennetta sekoittavia tai muuten ”vaarallisia” merkkejä. Papu otettiin käyttöön Korppi-järjestelmän jatkokehityksessä.

Papuun lisättiin metodi `RegexpEncode`, joka tunnistaa syötemerkkijonosta Java-kielen säännöllisiin lausekkeisiin liittyvät ohjausmerkit ja muuttaa ne tavallisina merkkeinä tulostuvaan muotoon.

3.1.2 Tools

`Tools`-papu sisältää yleiskäyttöisiä metodeja. Se on ollut käytössä Kotka-projektista lähtien.

Papuun lisättiin metodi `redirectWithCurrentParams`, joka tyhjentää JSP-kirjoittajaolion puskurin ja vie käyttäjän toiselle sivulle nykyisen sivun parametreilla. Metodia kutsutaan JSP-sivulta, kutsun jälkeen toteuttajan pitää lisätä `return`-lause koodiin.

3.1.3 Event

`Event`-papu kapseloi tapahtumaan liittyvät tiedot. Se otettiin käyttöön Korppi-järjestelmän jatkokehityksessä.

Papuun lisättiin uusi konstruktori, joka ottaa parametrina tapahtuman id-numeron. Toteuttaja voi valinnaisesti asettaa pavun attribuuttien arvot suoraan tietokannasta.

3.2 kiurubeans-paketin rajapinnat

Yleiskäyttöisyyden maksimoimiseksi ja Java-kielen moniperintärajoitusten kiertämiseksi erikoistyön aikana määriteltiin muutamia rajapintoja `kiurubeans`-pakettiin. Rajapintojen käyttötarkoituksia selvitetään tarkemmin Java-papujen kuvausten yhteydessä luvuissa 3.3.6, 3.3.8 ja 3.3.10.

3.2.1 EnumType

`EnumType` mallintaa C-kielestä tuttua lueteltua tyyppiä. Tämänhetkisessä määrityksessä tyypin tila on sisäisesti kokonaisluku. Rajapinnan toteuttavien luokkien pitäisi merkitä mahdolliset tilansa julkisiksi staattisiksi muuttujiksi, jolloin tilaa voidaan kysyä rajapinnan metodien avulla. Rajapinta määrittelee seuraavat metodit:

`setEnumState`	asettaa olion uuteen tilaan.
`getEnumState`	palauttaa olion tämänhetkisen tilan.

3.2.2 OrderScheme

`OrderScheme` kuvaa jotain tapaa järjestää yksittäisen kentän tiedot SQL-kyselyn `ORDER BY` -osassa. Rajapinta määrittelee seuraavat metodit:

`setFieldName`	asettaa kentän nimen. Nimi voi olla toteuttavasta luokasta riippuen kentän varsinainen nimi tai alias.
`getFieldName`	palauttaa kentän nimen.
`getOrder`	palauttaa kenttään liittyvän SQL-merkkijonon, jonka voi liittää sellaisenaan kyselyn `ORDER BY` -osan jatkoksi.

3.2.3 Predicate

`Predicate` mallintaa logiikasta tuttua predikaattia. Rajapinnan avulla voidaan testata, onko jokin asia totta.

Rajapinta määrittelee vain metodin `test`, joka palauttaa syötemerkkijonosta (tai rajapinnan toteuttavan luokan tilasta) riippuvan totuusarvon.

3.3 kiurubeans-pakettin papuihin tehdyt muutokset

Kiuru-erikoistyössä toteutut uudet ja muokatut pavut jakautuvat karkeasti seuraaviin ryhmiin:

- yleiskäyttöiset pavut, kuten `HtmlBean` ja `SimpleDb`,
- kotka-tietokannan kohteiden käsittelijät, yliluokkana `EntityHandler` sekä
- hakutoimintojen käsittelijät, yliluokkana `SearchHandler`.

Osassa pavuista hyödynnettiin luvussa 3.2 määriteltyjä rajapintoja. Uusien papujen suunnittelussa ja toteutuksessa sekä aiempien papujen korjauksissa pyrittiin Kiuru-projektin tapaan hyödyntämään Java-kielen olio-ohjelmointia tukevia ominaisuuksia. Lisäksi pavuista pyrittiin tekemään mahdollisimman yleiskäyttöisiä.

3.3.1 HtmlBean

`HtmlBean`-papu sisältää HTML-sivujen generointia helpottavia yleiskäyttöisiä metodeja. Pavun perusrakenne on pääosin sovellusprojektin toteutuksen mukainen: joukko merkkijonoja palauttavia staattisia metodeita. Muutaman metodin osalta on kokeiltu merkkijonon palautuksen sijaan suoraa tulostusta `PrintWriter`-tyyliseen tietovirtaan. Tietovirta määritellään attribuutissa `writer`.

`HtmlBean`-papu sisältää attribuuttien `get`- ja `set`-metodien lisäksi seuraavat uudet ja muokatut metodit:

`begin_javascript` ja `end_javascript`	palauttavat HTML-elementit, joiden väliin voidaan kirjoittaa JavaScript-koodia. Jos käyttäjällä ei ole JavaScriptia käytössä, koodi jää kommentteihin.
`getCurrentPage`	palauttaa tämänhetkisen JSP-sivun nimen. Sivulle mahdollisesti annetut parametrit eivät ole mukana.
`getCurrentTime`	palauttaa merkkijonon, jossa on tietokoneen kellon arvo millisekunteina parametrissa `currentTime`. Merkkijono voidaan liittää suoraan URL:n loppuun, jolloin linkin takana oleva sivu ladataan varmasti uudestaan.
`getH`	palauttaa HTML-otsikkoelementin käyttäjän määräämällä sisällöllä, teksti käännetään.
`getJsButton`	palauttaa JavaScript-koodin, jossa tulostetaan HTML-lomakkeen painike. Painikkeen teksti käännetään automaattisesti ja sille voi määrätä `onClick`-tapahtumassa suoritettavan JavaScript-funktion.
`getLabel`	palauttaa (kääntämättömän) tekstin tyylillä, joka soveltuu käytettäväksi HTML-lomakkeen sisällä.
`getLink`	palauttaa HTML-linkin. Linkkitekstinä voidaan käyttää erillistä oletustekstiä jos varsinainen linkkiteksti on tyhjä.
`printHtmlTable`	tulostaa tietovirtaan `HtmlTable`-komponentin tiedot (katso luku 3.3.5).
`printOptionValues`	tulostaa HTML-lomakkeilla käytettävän `option`-elementtilistan `HashTable`-oliossa annettujen tietojen perusteella. Metodi on otettu Portal-osion JSP-sivulta `pageAddUser`.
`printSelectList`	luo HTML-lomakkeen valintalistan `HashTable`-oliossa annettujen tietojen perusteella. Lista tulostetaan tietovirtaan.
`printSubmit`	tulostaa tietovirtaan HTML-lomakkeen `submit`-painikkeen.
`translate`	kääntää annetun merkkijonon `User`-pavun `T`-metodilla.

3.3.2 KiuruString

`KiuruString`-papu sisältää merkkijonojen käyttöä helpottavia yleiskäyttöisiä metodeja. Se sisältää attribuuttien `get`- ja `set`-metodien lisäksi seuraavat uudet metodit:

`anyCharIndexOf`	vastaa `String`-luokan metodia `indexOf`, mutta sille voi antaa merkkijonon. Palauttaa annetun merkkijonon minkä tahansa merkin ensimmäisen esiintymän paikan.
`arrayToGeneralCSV`	muuntaa `String`-taulukon annetulla erotinmerkillä erotelluksi merkkijonoksi.
`csvToArray`	muuntaa pilkulla erotellun merkkijonon `Object`-taulukoksi.
`CSVToGeneralCSV`	vaihtaa pilkulla erotellun merkkijonon erottimen määritellyksi.
`isEmpty`	tutkii, onko merkkijono tyhjä. Myös `null`-viite tulkitaan tyhjäksi merkkijonoksi.
`jsQuote`	lisää merkkijonoon JavaScript-tyyliset lainausmerkit ympärille.
`stripChars`	poistaa merkkijonosta kaikki annettuun merkkijonoon kuuluvien merkkien esiintymät.

3.3.3 SqlSearchParser

Luokka on yleistys `kotkabeans`-paketin `Tools`-luokan `parseSearch`-metodille. Samalla luokka on toteutus erikoistyön aikana määritellylle hakulogiikalle (katso luku 4.7). Luokkaa käytetään merkkijonoarvoisen kentän haettavien arvojen rajoittamiseen.

Luokka sisältää yksikkötestin lisäksi vain staattisen metodin `search`, joka generoi SQL-kyselyn `WHERE`-osaan liitettävän hakulauseen. Syötteenä on haettavan kentän nimi ja käyttäjän antama hakumerkkijono.

3.3.4 PhraseStringTokenizer

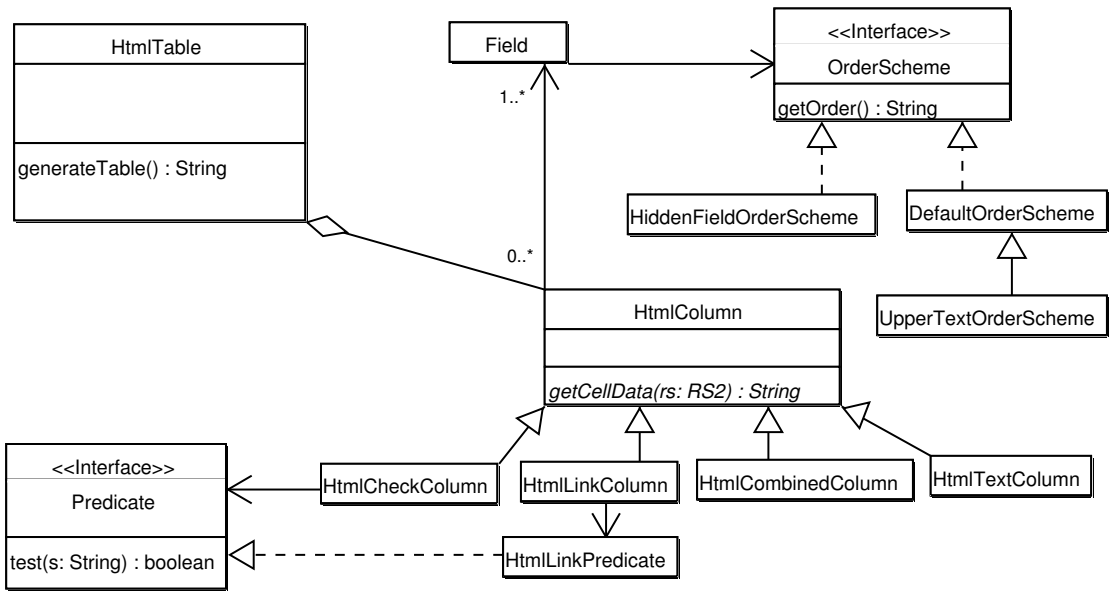
Luokka on yleistys Java-kielen `StringTokenizer`-luokalle sisältäen samannimiset metodit. Erona alkuperäiseen luokkaan on fraasin käsite. Fraasi on erityisillä erotinmerkeillä ympäröity osa merkkijonoa, joka tulkitaan yhdeksi sanaksi (token) riippumatta siitä, sisältääkö se tavallisia erotinmerkkejä.

`PhraseStringTokenizer` suunniteltiin `SqlSearchParser`-luokan käytöön, mutta on periaatteessa yleiskäyttöinen. Yksikkötestin lisäksi luokka sisältää seuraavat metodit:

`hasMoreElements`	testaa, sisältääkö merkkijono vielä sanoja.
`nextElement`	palauttaa seuraavan sanan (tai fraasin) merkkijonosta `Object`-muodossa.
`nextToken`	palauttaa seuraavan sanan merkkijonosta.

3.3.5 HtmlTable

`HtmlTable` ja sen apuluokat muodostavat laajimman erikoistyön aikana toteutetun yksittäisen komponentin. Komponentin tarkoitus on automatisoida tietokantapohjaisten HTML-taulukkojen generointia JSP-sivuille. Pelkän tietojen tulostuksen lisäksi taulukkoon on mahdollista sijoittaa linkkejä ja valintaruutuja. Taulukon voi järjestää minkä tahansa kentän mukaan otsikkorivillä olevien linkkien avulla. Komponentin rakennetta on havainnollistettu kuvassa 3.1.



Kuva 3.1: `HtmlTable`-luokkaan liittyvät suhteet UML-luokkakaaviona.

`HtmlTable`-luokka toimii oleellisesti säiliönä sarakkeita kuvaaville `HtmlColumn`-tyyppisille olioille. `HtmlTable`-luokan merkittävin metodi on `generateTable`, joka palauttaa HTML-taulukon koodin merkkijonossa. Ennen taulukon generointia oliolle täy-

tyy ilmaista näytettävät sarakkeet ja RS2-tietojoukko, josta tiedot luetaan.

Luokkaa on mahdollista yleistää nykyisestä. Eräs tärkeä laajennus olisi taulukon näytettävien arvojen rajoittaminen hakuehdon mukaan mielivaltaisen kentän perusteella. Tarve ominaisuudelle tuli esille haastatteluissa (katso luku 4.3), mutta sitä ei ehditty toteuttaa erikoistyön aikana.

3.3.6 HtmlColumn ja sarakeluokat

`HtmlColumn`-luokka kuvaa yksittäisen sarakkeen tiedot. Se toimii `HtmlTable`-luokan apuluokkana sekä on myös yliluokka luokille `HtmlCheckColumn`, `HtmlLinkColumn`, `HtmlCombinedColumn` ja `HtmlTextColumn`. Aliluokista ensimmäinen palauttaa valintaruutuja HTML-taulukkoon, `HtmlCombinedColumn`-tyyppisen sarakkeen arvo koostetaan useammasta tietokannan kentästä ja `HtmlLinkColumn`-arvo on linkki. Viimeinen aliluokka on tavanomainen tekstikenttä.

Tietokannan kentän määrittämiseen ja tietojen lukemiseen käytetään `Field`-tyyppistä attribuuttia. Luokat `HtmlCheckColumn` ja `HtmlLinkColumn` käyttävät `Predicate`-rajapinnasta perittyjä luokkia määrittämään, onko valintaruutu valittu tai palautetaanko linkki vai teksti. Luokat sisältävät attribuuttien `get-` ja `set`-metodien lisäksi seuraavat metodit:

`getCellData`	palauttaa HTML-taulukon soluun tulevan arvon tämänhetkisen RS2-tietojoukon arvon perusteella.
`getHeader`	palauttaa HTML-taulukon otsikkoon tulevan arvon.

3.3.7 Field ja kenttäluokat

`Field`-luokka abstrahoi tietokannan kentän. Luokkaa käytetään sekä `HtmlTable`- että `SearchHandler`-luokkien apuluokkana (katso kuvat 3.1 ja 3.2). `OrderScheme`-tyyppiä oleva attribuutti määrää, millä tavalla kentän arvoja järjestetään.

`Field` on yliluokka `TextField`- ja `DateField`-luokille. `TextField` on kuten `Field`, mutta järjestämiseen sovelletaan oletuksena `UpperTextOrderScheme`-tyyppistä attribuuttia. `DateField` palauttaa kentän arvon tiivissä muodossa, joka soveltuu suoraan tulostettavaksi. Luokat sisältävät attribuuttien `get-` ja `set`-metodien lisäksi seuraavat metodit:

`getOrderName`	palauttaa merkkijonon, jota käytetään SQL-kyselyn `ORDER BY` -osassa.
`getFieldValue`	palauttaa kentän arvon tämänhetkisellä `RS2`-tietojoukon rivillä.

3.3.8 OrderScheme-toteutukset

Kiuru-sovellus sisältää kolme eri toteutusta `OrderScheme`-rajapinnalle: `DefaultOrderScheme`, `UpperTextOrderScheme` ja `HiddenFieldOrderScheme`. Luokilla määritetään tapa `Field`-luokan kuvaaman kentän järjestämiseen.

Näistä ensimmäinen vastaa tavanomaista kentän nimen tai aliaksen lisäämistä `ORDER BY` -lauseeseen. `UpperTextOrderScheme`-luokkaa käytetään tilanteessa, kun ei haluta antaa kirjainkoon vaikuttaa järjestämisen tulokseen. Tällöin `ORDER BY` -lauseeseen lisätään ylimääräinen `UPPER`-funktion kutsu. Viimeistä järjestysmallia käytetään tilanteessa, kun jokin muu kenttä määrää näytettävän kentän järjestyksen. Tällöin `ORDER BY` -lausetta kutsutaan erikseen ilmaistun piilotetun kentän mukaan.

3.3.9 SimpleDb

`SimpleDb`-papu sisältää tietokannan käyttöä helpottavia yleiskäyttöisiä metodeja. Se sisältää attribuuttien `get-` ja `set`-metodien lisäksi seuraavat uudet metodit:

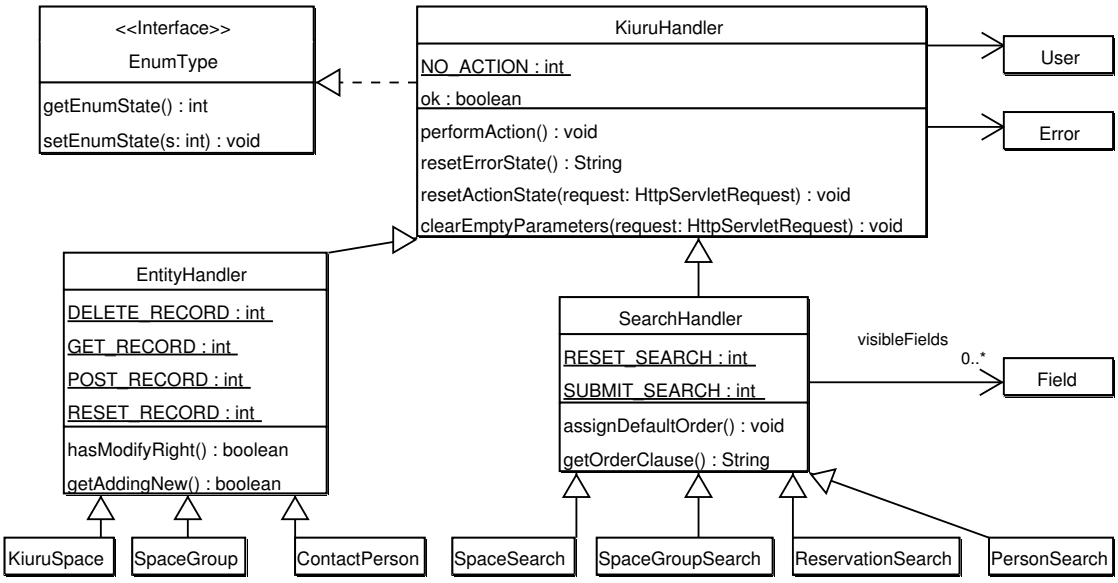
`getDateTimeString`	palauttaa `datetime`-tyyppiä olevan kentän arvon lyhyesti muotoiltuna.
`isEmpty`	tutkii, onko `RS2`-tietojoukko tyhjä. Myös `null`-viite tulkitaan tyhjäksi tietojoukoksi.

3.3.10 KiuruHandler

`KiuruHandler` on abstrakti yliluokka tietokannan tietokohteiden ja hakuluokkien käsittelijöille. Luokkien yhteisenä tekijänä on ”toiminnon” käsite. Toimintoja voivat olla esim. tietojen tallennus tietokantaan tai haku jollakin kriteerillä. Toiminnon määrää olion tila, joka määritellään `EnumType`-rajapinnan toteutuksena. Luokan ainoa toiminto on `NO_ACTION`, joka tietueluokilla ei tee mitään ja hakuluokilla tulkitaan nykyisen haun jatkamiseksi. Aliluokat voivat määritellä uusia toimintoja syrjäyttämällä `defaultAction`-metodin.

JSP-sivuja käytettäessä olion tila määräytyy HTTP-parametrien avulla ja toiminto ajetaan aina sivulle tultaessa. Toiminto yksinkertaistaa merkittävästi JSP-sivujen koodausta. JSP-käyttöliittymäsivulla ei tarvitse erikseen käsitellä, mitä sivun pitäisi tehdä, vaan riittää

kutsua `performAction`-metodia. Sama toimintalogiikka on yleistetty sekä tietokannan käsittelyä että tietojen hakua varten. `KiuruHandler`-luokasta perittyjen luokkien suhteita on havainnollistettu kuvassa 3.2.



Kuva 3.2: `KiuruHandler`-luokkaan liittyvät suhteet UML-luokkakaaviona.

Papu sisältää attribuuttien `get-` ja `set`-metodien lisäksi seuraavat metodit:

`performAction`	suorittaa toiminnon. Toiminnot määrätään yleensä JSP-sivujen parametreina.
`resetErrorState`	tyhjentää `Error`-olion sekä asettaa olion normaaliin tilaan, jos on oltu virhetilassa. Metodia kutsutaan yleensä JSP-sivun alussa.
`resetActionState`	varmistaa, että olio on aloitustilassa editoitaessa uutta tietuetta tai syötettäessä uutta hakua. Metodia kutsutaan JSP-käyttöliittymäsivun alussa ennen `performAction`-metodikutsua. Metodia ei pidä kutsua JSP-käsittelijäsivulla, koska käsittelijäsivun toiminta riippuu olion tilasta.
`clearEmptyParameters`	varmistaa, että tyhjiä HTTP-parametreja vastaavat attribuutit tyhjennetään. Metodia tarvitaan, jos käyttäjä asettaa HTML-lomakkeen johonkin kenttään tyhjän arvon. Tällöin kenttää kuvaavaa HTTP-parametria ei lähetetä ollenkaan, jolloin käsittelijäolion vastaava attribuutti ei päivity automaattisesti.

3.3.11 EntityHandler ja tietueluokat

`EntityHandler` on abstrakti yliluokka tietojen muokkaukseen liittyvien sivujen käsittelijöille. Se on peritty `KiuruHandler`-luokasta (katso luku 3.3.10). Luokkien yhteisenä tekijänä ovat tietueiden käsittelytoiminnot, joita ovat haku tunnisteen mukaan, tietojen tallennus, tietueen tyhjennys ja tietojen poisto.

`EntityHandler`-luokan toteutettuja aliluokkia ovat `KiuruSpace`, `SpaceGroup` ja `ContactPerson`. Jokainen niistä sisältää omia haku- ja käsittelymetodeita, mutta perustoiminnot ovat samat. Myös uusia aliluokkia voisi määritellä tarpeen mukaan. Papu sisältää attribuuttien `get-` ja `set`-metodien sekä tietueiden käsittelytoimintojen määrittelyjen lisäksi seuraavat metodit:

`hasModifyRight`	-metodilla voidaan testata käyttäjän oikeudet tietueeseen. Metodi palauttaa toden vain, jos käyttäjällä on oikeudet muokata kaikkia olion attribuutteja vastaavia kenttiä. Jotkut tietokohteet (esim. `SpaceGroup`) voivat määritellä käyttöoikeudet sisäisesti tarkemminkin. Yleisen tietokohdekohtaisen käyttöoikeusmallin puuttuessa metodi pakottaa kehittäjän määrittämään vähintään karkean tason käyttöoikeudet tietueeseeen.
`getAddingNew`	palauttaa tiedon, onko käyttäjä lisäämässä uutta tietuetta vai muokkaamassa olemassaolevaa.

3.3.12 SearchHandler ja hakuluokat

`SearchHandler` on `KiuruHandler`-luokasta (katso luku 3.3.10) peritty abstrakti yliluokka hakuluokkien käsittelijöille. Luokkien yhteisenä tekijänä ovat toiminnot haun tyhjennys ja hakujoukon muodostus. Lisäksi luokka määrittelee kenttien järjestämistiedot, joissa hyödynnetään `Field`-luokkaa ja `OrderScheme`-rajapintaa.

Hakuluokkia olisi tarpeen yleistää nykyisestä. Esimerkiksi haettavia kenttiä ja hakuehtoja pitäisi voida muokata yhteisellä rajapinnalla. Nykyisellään jokainen luokka määrittelee omat hakumetodinsa erikseen ja haettavia kenttiä ei voi muuttaa dynaamisesti.

`SearchHandler`-luokan toteutettuja aliluokkia ovat `SpaceSearch`, `SpaceGroupSearch`, `PersonSearch` ja `ReservationSearch`. Aliluokat sisältävät oletushakutoiminnon lisäksi useita erikseen määriteltyjä (osittain staattisia) hakumetodeja, joissa esim. palautettavat kentät vaihtelevat. `SearchHandler`-luokka sisältää attribuuttien `get-` ja `set-`metodien lisäksi seuraavat metodit:

`assignDefaultOrder`	järjestää haun kentät oletusjärjestykseen.
`getOrderClause`	palauttaa järjestysosan SQL-lauseesta.

4 Muut tehtävät

Luvussa kuvataan erikoistyön aikana toteutettuja yleisiä toimintoja Korppi-järjestelmään. Lisäksi käsitellään erikoistyöryhmälle annettuja tehtäviä ja kehitysehdotuksia.

4.1 Tehtävien jako ja ajankäyttö

Jussi Mäkinen kehitti salivarauksiin ja ilmoituksiin liittyviä toimintoja sekä kalenterin että opettajan käyttöliittymän puolella. Miika Nurminen keskittyi Java-papujen ja JSP-sivujen käsittelylogiikan kehittämiseen ja yleistämiseen.

Erikoistyön suoritukseen käytettiin yhteensä aikaa vajaat 200 tuntia. Työmäärä jakautui ryhmäläisten kesken melko tasaisesti. Yksityiskohtainen selvitys ajankäytöstä on merkitty erikoistyöläisten tuntikirjoihin Kiuru-projektin WWW-sivujen alla.

4.2 Kiuru-sovelluksen arkkitehtuuri

Erikoistyön kehityksessä sovellettiin samoja periaatteita kuin Kiuru-sovellusprojektissa syksyllä 2002. Käyttöliittymä ja sovelluslogiikka pyrittiin erottamaan entistä selkeämmin toisistaan ja uusia yleiskäyttöisiä papuja toteutettiin. Lisäksi olemassaolevaa koodia selkeytettiin ja yhtenäistettiin, erityisesti JSP-sivujen käsittelylogiikan ja HTML-taulukoiden generoinnin osalta. Kiuru-sovelluksen arkkitehtuuria ja kehitysehdotuksia on pohdittu tarkemmin Ohjelmistoarkkitehtuurit-kurssin tenttivastauksessa [8].

4.3 Haastattelut

Erikoistyön vaatimusten tarkentamiseksi erikoistyöryhmä haastatteli Hannele Säntti-Ahomäkeä ja Marita Heittolaa. Säntti-Ahomäen haastattelussa [7] ilmeni tarve tarkastella varauksia erityisesti yksittäisten salien osalta. Tämä vaatii hakutoimintojen kehittämistä nykyisestä varauslistan näytön osalta jatkokehityksessä. Heittolan haastattelussa [6] kartoitettiin mahdollisuuksia ottaa Korppi-järjestelmä hallinnon käyttöön Sali-järjestelmän tilalle.

4.4 Erikoistyön lähdekoodi ja versionhallinta

Erikoistyön alkuvaiheessa erikoistyöryhmä kehitti sovellusta omassa CVS-haarassaan. Kevään aikana Korppi-jatkokehitys teki huomattavia muutoksia Java-papuihin ja tietokantaan (esim. `Translation`-tauluista luopuminen). Erikoistyön ja Korppi-jatkokehityksen koodin synkronoinnin helpottamiseksi erikoistyökoodi yhdistettiin huhtikuussa 2003 Korppi-päähaaraan, jossa kehitys jatkuu. Yhdistämisen tarkoituksena on myös helpottaa salivarausjärjestelmän mahdollista käyttöönottoa Korppi-järjestelmän tuotantoversioon tulevaisuudessa.

Yliopiston luovuttua Timmi-järjestelmän käytöstä Timmi-rajapintaa käsittelevä koodi tuli tarpeettomaksi. Koodin selkeyttämisen takia Timmi-järjestelmään liittyvät pavut poistettiin `kiurubeans`-paketin kehitysversiosta toukokuussa 2003. Jos pavuille on myöhemmin tarvetta, ne ovat Kiuru-projektin CVS:ssä sekä Kiuru-projekin projektikansion [2] mukana olevalla CD-ROM:illa.

Kiuru-sovelluksen asennukset testikoneille tehtiin erikoistyön aikana manuaalisesti. Sovelluksen asennus vaatii Java-papujen kääntämisen ja Tomcat-palvelimen uudelleenkäynnistämisen. Erikoistyöryhmä suosittelee jatkossa Ant-työkalun [1] tutkimista sovelluksen kääntämisen ja asennuksen automatisointiin.

4.5 Testaus

Erikoistyön aikana erikoistyön tekijöiden lisäksi testauksessa oli mukana myös muita henkilöitä. Erityiskiitos Vesa Lappalaiselle sovelluksen testauksesta ja parannusehdotuksien esittämisestä. Löydetyt virheet korjattiin sitä mukaa, kun niitä ilmeni. Tarkistus käyttäjän oikeuksista muokata tapahtumaa, jolle ollaan varaamassa salia, on raporttia kirjoitettaessa kesken. Ominaisuus toteutetaan yhteistyössä Korppi-kehitystiimin kanssa.

Yksikkötestausta ei saatu erikoistyön aikana automatisoitua. Jatkossa yksikkötestauksessa kannattanee hyödyntää esim. `JUnit`-ympäristöä [3].

4.6 SQL-kyselyt

Erikoistyön aikana kävi ilmi, että jotkut SQL-kyselyt ovat liian hitaita tietomäärän kasvaessa suureksi. Kyselyjä optimoitiin muuttamalla liitoksia alikyselyiksi. Joitakin SQL-kyselyjä tarvinnee optimoida myöhemmin näiden taulujen rivien määrän kasvaessa.

Tietokantaan lisättiin myös sääntö, joka muuttaa `reservation`-taulun `inconsis-`

`tent`-lipun tarvittaessa pois päältä. Lisäksi muutamien kenttien määrityksiä tarkennettiin oletusarvojen ja `NULL`-arvon sallimisen osalta. Uudet SQL-määritykset tallennettiin osaksi Kiuru-sovellusprojektin CVS-haaraa.

4.7 Hakulogiikka

Korppi-järjestelmän hakutoimintoja varten määritettiin uusi hakulogiikka, joka on käytössä kaikilla salivarauksiin, saleihin ja raportteihin liittyvillä hakusivuilla. Hakulogiikkaa sovelletaan HTML-lomakkeiden tekstikomponenteissa, joihin käyttäjä voi syöttää haluamansa hakusanat. Hakulogiikka on pääpiirteissään seuraavanlainen:

- Jos on vain yksi hakusana, haetaan kentän alusta. Muuten hakusanat yhdistetään `AND`-operaattorilla niin, että haetaan oletuksena sanoja, joissa hakuehto esiintyy sanan osana. Useamman hakusanan tapauksessa välilyönti toimii erotinmerkkinä.
- Jos käyttäjä syöttää hakuehdon mukana jokerimerkkejä (`*`) tai (%), ne syrjäyttävät oletuslogiikan ko. sanan kohdalta.
- Lainaus- (¨) ja heittomerkkien (`'`) rajoittama sisältö tulkitaan yhdeksi hakusanaksi (fraasi) ja käsitellään edellä kuvattujen ehtojen mukaaan. Jos fraasissa ei ole jokerimerkkejä, se tulkitaan sellaisenaan.
- Pilkku (`,`), kauttaviiva (`/`), puolipiste (`;`) ja plus-merkki (`+`) sanojen tai fraasien välissä jakavat haun OR-operaattoreilla erotettuihin osiin. Osat käsitellään yllä kuvattujen ehtojen mukaan.

Tarkempi kuvaus hakulogiikasta on Korpin koodausstandardissa [5]. Hakulogiikan toteutus on `kiurubeans`-paketin luokassa `SqlSearchParser` (katso luku 3.3.3).

4.8 Tuntemattomat yhteyshenkilöt

Erikoistyön aikana tarkennettiin sovellusprojektissa ilmennyttä tarvetta tuntemattomille yhteyshenkilöille. Yhteyshenkilöitä tarvitaan annettaessa varauspyyntöä. Jos yhteyshenkilöllä ei ole tunnusta Korppi-järjestelmään tai jos yhteyshenkilö ei tiedossa, puhutaan tuntemattomista yhteyshenkilöistä. Myös kevään 2003 Kottarainen- ja Portti-sovellusprojekteissa ilmeni tarve henkilöille, joilla ei ole tunnusta korppiin.

Tuntemattomat henkilöt pidetään Kotka-tietokannan `Person`-taulussa. Erona tavallisiin henkilöihin `account`-kentän arvo on `NULL`. Kotiorganisaation on oltava määritelty myös tuntemattomille henkilöille. `Person`-taulun käyttämisen etuna on se, että tuntemattomat henkilöt voidaan helposti sisällyttää samoihin SQL-hakuihin kuin varsinaiset henkilöt. Lisäksi tunnuksen lisääminen tarvittaessa jälkikäteen on helppoa.

Tietokannassa olevat henkilöt jakautuvat yhteyshenkilöiden näkökulmasta kolmeen luokkaan:

- Varsinaiset käyttäjät, jotka pystyvät kirjautumaan Korppiin.
- Varsinaiset henkilöt, jotka eivät halua tai voi kirjautua Korppiin, mutta jotka liittyvät Korppi-järjestelmän tapahtumiin (myöhemmin heidät tulisi pystyä nostamaan käyttäjiksi). Näitä ovat esimerkiksi yliopiston ulkopuolisten järjestäjäorganisaatioiden yhteyshenkilöt.
- Abstraktit henkilöt, jotka eivät olemassaolemattomuutensa / monikäsitteisyytensä takia pysty kirjautumaan Korppiin, mutta joita käytetään henkilönimikkeinä esim. salivarauksissa. Abstrakteja henkiöitä käytetään myös, jos tapahtuman yhteyshenkilöstä tai pitäjästä ei ole tietoa salia varatessa.

Varsinaiset ja abstraktit henkilöt erotetaan toisistaan uudella `PersonParameter`-tietueella. Tämä mahdollistaa myös uusien henkilötyyppien määrittelyn tulevaisuudessa tarvittaessa.

Abstraktit yhteyshenkilöt määritellään organisaatiokohtaisesti, jolloin ne näkyvät salivarausta tehtäessä varauspyynnön tekijän organisaation valinnaisina yhteyshenkilöinä. Abstrakteja yhteyshenkilöitä ovat esimerkiksi *assistentti*, *tuntiopettaja* tai vaikka *tietotekniikan laitos*, sillä jotkut laitokset käyttävät laitoksen nimeä varaajanimikkeenä.

Tuntemattomien yhteyshenkilöiden lisääminen on toteutettu sivulle `giveAllInformation.jsp`. Abstrakteja yhteyshenkilöitä ja ylimääräisiä `PersonParameter`-arvoja ei ehditty toteuttaa. Tarkemmat tiedot tuntemattomien yhteyshenkilöiden määrityksistä ovat Kiuru-postituslistan arkistossa [4] vuoden 2003 maaliskuun viesteissä.

5 Yhteenveto

Erikoistyön aikana Kiuru-sovellus saatiin melko hyvään ja yksittäisten testien perusteella vakaaseen vaiheeseen, vaikka vikoja löytyikin matkan varrella. Sovelluksen taustalla olevaa toimintalogiikkaa muokattiin ja yhtenäistettiin. Lisäksi hakusivujen hakulogiikkaa paranneltiin.

Tietokantakyselyjä nopeutettiin ja korjailtiin. Parhaimmillaan 28 s kestäneet kyselyt, saadaan suoritettua parissa sekunnissa.

Kiuru-sovellus on toiminnallisesti lähellä Sali-järjestelmää ja pystyy periaatteessa korvaamaan sen. Ennen Kiuru-sovelluksen käyttöön siirtymistä sovellusta täytyy kuitenkin parantaa käytettävyydeltään ja lisätä uusia haku- ja raportointitoimintoja. Erikoistyön aikana tehdyissä haastatteluissa kartoitettiin mahdollisuuksia Sali-järjestelmän korvaamiseen Kiuru-sovelluksella.

Jussi Mäkinen kehittää kesän 2003 aikana sovellusta edelleen. Tällä hetkellä sovellus ei tue mm. oheisvarattavia. Lisäksi erityyppisiä raportteja tarvittaneen lisää myöhemmin. Suurten varausmäärien tehokasta hallintaa varten tarvittaneen myös WWW-käyttöliittymää käytettävämpää GUI-pohjaista käyttöliittymää.

Lähteet

[1] "Apache Ant", Apache Software Foundation, 2003, <URL: http://ant.apache.org/>.

[2] Hilpinen Toni, Koivuniemi Marko, Mäkinen Jussi ja Nurminen Miika, ”Kiuru-projektin projektikansio”, Jyväskylän yliopisto, tietotekniikan laitos, 2003.

[3] "JUnit", Testing Resources for Extreme Programming Object Mentor Inc, 2002, <URL: http://junit.org/>.

[4] Kiuru-projektin postituslistan arkisto, Jyväskylän yliopisto, tietotekniikan laitos, 2003, <URL: http://korppi.it.jyu.fi/list-archive/kiuru/>.

[5] Kujala Pauli, Lappalainen Vesa et al., "Korpin koodausstandardi", tekninen raporttiluonnos, Jyväskylän yliopisto, tietotekniikan laitos, 2003, <URL: https://jamshedpur.it.jyu.fi/kotka/korppicoding.txt>.

[6] Mäkinen Jussi, "Marita Heittolan haastatteluraportti", Jyväskylän yliopisto, tietotekniikan laitos, 2003, <URL: http://kotka.it.jyu.fi/kiuru/erikoistyo/haastattelut/heittola.txt>.

[7] Mäkinen Jussi ja Nurminen Miika, "Hannele Säntti-Ahomäen haastatteluraportti", Jyväskylän yliopisto, tietotekniikan laitos, 2003, <URL: http://kotka.it.jyu.fi/kiuru/erikoistyo/haastattelut/haastattelu.txt>.

[8] Nurminen Miika, *Kiuru-sovellusprojektin arkkitehtuuri*, Ohjelmistoarkkitehtuurit-kurssin tenttivastaus, Jyväskylän yliopisto, tietotekniikan laitos, 2003, <URL: http://kotka.it.jyu.fi/kiuru/erikoistyo/arkkitehtuuri/vastaus.pdf>.