Suunnitteludokumentti

Kotkat-ryhmä

Helsinki 18.5.2004

Ohjelmistotuotantoprojekti

HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

Kurssi

581260 Ohjelmistotuotantoprojekti (6 ov)

Projektiryhmä

Katja Astikainen

Manta Jääskeläinen

Riikka Kaven

Leena Laivaara

Säde Seppälä

Marja Silenti

Asiakas

Heikki Lokki

Johtoryhmä

Juha Taina

Turjo Tuohiniemi

Kotisivu

`http://www.cs.helsinki.fi/group/kotkat`

Versiohistoria

Versio	Päiväys	Tehdyt muutokset
0.1	18.03.2004	Ensimmäinen versio
0.2	23.03.2004	Toimitettu asiakkaalle FTR:ää varten
1.0	01.04.2004	Korjattu FTR:n perusteella
1.1	17.05.2004	Muutoksia toteutusvaiheen jälkeen

Sisältö

1 Johdanto **1**

1.1 Tuotteen tausta ja tarkoitus 1

1.2 Tärkeimmät vaatimukset 2

1.3 Erikoissanasto ja käytetyt lyhenteet 2

2 Suunnittelun ja toteutuksen rajoitteet **5**

2.1 Noudatettavat standardit ja tarvittavat ohjelmat 5

2.2 Luokkakirjastot ja ajurit 5

2.3 Ohjelmointikieli ja -tyyli 5

3 Järjestelmän yleiskuvaus **6**

4 Arkkitehtuurikuvaus **7**

4.1 Komponenttien väliset suhteet 8

4.2 Luokkien väliset suhteet 8

4.3 Pakkaus hali.db 9

4.3.1 ConnectionPool 10

4.3.2 Table 13

4.3.3 OperationResults 17

4.3.4 SearchResults 18

4.3.5 DatabaseOperation 20

4.3.6 SearchOperation 22

4.3.7 InsertOperation 24

4.3.8 UpdateOperation 25

4.3.9 DeleteOperation 27

4.4 Pakkaus haliaeetus.servlet 28

4.4.1 HaliaeetusControllerServlet 28

4.5 Pakkaus haliaeetus.helper 29

4.5.1 HaliProperties 29
4.5.2 CheckHelper 31
4.5.3 NestInspectionHelper 32
4.5.4 CheckPoisonHelper 32
4.5.5 CheckPreyHelper 33
4.5.6 CheckHistoryHelper 34
4.5.7 CheckInspectorHelper 34
4.5.8 CheckTerritoryHelper 35
4.5.9 CheckReportHelper 36
4.5.10 Coords 36
4.5.11 MunicipatyHelper 36
4.6 Pakkaus haliaeetus.command 37
4.6.1 HaliaeetusGeneral 39
4.6.2 CommandDispatcher 42
4.6.3 CommandFactory 43
4.6.4 CommandInterface 44
4.6.5 LoginScreenCommand 45
4.6.6 MainScreenCommand 46
4.6.7 NestInspectionCommand 49
4.6.8 RepearNestInfoCommand 51
4.6.9 NestInformationCommand 53
4.6.10 PoisonCommand 55
4.6.11 PrayCommand 56
4.6.12 HistoryCommand 58
4.6.13 TerritoryCommand 60
4.6.14 MunicipatyCommand 61
4.6.15 InspectorCommand 66
4.6.16 HelperTableCommand 67

4.6.17 ReportCommand 70
4.7 Pakkaus haliaeetus.log 73
4.7.1 LoggingRulesInitializer 73
4.7.2 HaliaeetusLogger 73
4.8 Sekvenssikaaviot 75
4.9 Muut tiedostot 75
4.9.1 Template-tiedostot 76
4.9.2 Properties-tiedostot 79
4.9.3 Tyylitiedosto 81
4.9.4 Staattiset html-tiedostot 81

5 Käyttöliittymä **81**
5.1 Kuvasarjat 84
5.1.1 Käyttäjän sisäänkirjautuminen 84
5.1.2 Uuden pesän lisääminen 84
5.1.3 Aputaulun päivitys 85
5.1.4 Vanhan pesän tarkastus 85
5.1.5 Historia-näyttö 91
5.1.6 Kunnat-näyttö 91
5.1.7 Myrkyt-näyttö 93
5.1.8 Saaliit-näytöt 93
5.1.9 Tarkastajat-näyttö 93
5.1.10 Reviirit-näyttö 93

6 Testaussuunnitelma **94**
6.1 Komponenttitestaus 94
6.2 Integrointitestaus 95
6.3 Järjestelmätestaus 95
6.4 Testicaset 95

6.4.1 Käyttäjän tunnistus 95
6.4.2 Uuden pesän lisääminen 96
6.4.3 Vanhan pesän tarkastus 96
6.4.4 Pesän hakeminen 97
6.4.5 Raporttien tuottaminen 97
6.4.6 Aputaulun päivittäminen 98
6.4.7 Pesän tietojen muuttaminen 99
6.4.8 Reviiritietojen päivittäminen 100

Lähteet **101**

A Sekvenssikaaviot **0**

Liitteet

Liitteet

B navi.ftl **17**

C aputaulut.ftl **18**

D mainscreen.ftl **20**

E municipality.ftl **23**

F pesa.ftl **24**

G report.ftl **44**

H reviirit.ftl **46**

I login.ftl **49**

J tarkastajat.ftl **50**

1 Johdanto

Tämä suunnitteludokumentti kuvaa toteutettavan Haliaeetus-järjestelmän teknisen toteutuksen näkökulmasta. Dokumentin perusteella kuka tahansa teknisesti osaava henkilö voisi toteuttaa kuvatun järjestelmän. Suunnitteludokumentti on siis ohje siitä, kuinka järjestelmä tulisi kasata. Dokumentti kuvaa järjestelmän luokkarakenteen ja käyttöliittymän. Toteutusvaihe aloitetaan tämän dokumentin hyväksymisen jälkeen.

1.1 Tuotteen tausta ja tarkoitus

WWF:n merikotkatyöryhmän johtama suojelutyö merikotkan pelastamiseksi Suomessa sukupuuton partaalta elinvoimaisiksi populaatioksi saaristossa ja Pohjois-Suomessa on eräs luonnonsuojelun menestystarinoita.

Merikotkakannan romahduksen jälkeen pesimätietoja alettiin kerätä 1960-luvulla yksittäisten tutkijoiden toimesta. WWF:n merikotkatyöryhmän perustamisen jälkeen 1972 pesimätietojen kerääminen muuttui systemaattiseksi. Kolmenkymmenen vuoden aikana on tunnetut merikotkan pesät tarkastettu vuosittain. Tarkastuksen yhteydessä on kerätty tietoja mm. pesimistuloksesta, pesinnän epäonnistumisen syistä, poikasista, pesäpuusta ja -paikasta sekä kerätty kuoriutumattomia munia ja muita näytteitä myrkkyanalyysejä varten.

Merikotkille tarkoitettuja tekopesiä on rakennettu eri puolille Suomea pesien tahattoman häirinnän minimoimiseksi pesinnän herkimmissä vaiheissa. Merikotkien pesimäpiirit ovat verraten pysyviä. Kannan kasvaessa merikotkat perustavat uusia elinpiirejä vuosittain sekä aiemmin merikotkien asuttamille alueille että uusille seuduille.

Haliaeetus -järjestelmän avulla Luonnontieteellisessä keskusmuseossa syötetään ja ylläpidetään merikotkatietoja. Tiedoista kootaan raportteja tutkijoiden sekä suojelu- ja muiden viranomaisten käyttöön.

Ohjelmistotuotantoprojektiryhmän tarkoituksena on suunnitella ja toteuttaa merikotkien pesätarkastusten yhteydessä kerättyjen tietojen tallettamiseen ja käyttöön soveltuva tietokanta sekä käyttöliittymä tietokantaan. Ryhmämme jatkaa järjestelmän suunnittelua ja toteuttamista ohjelmistotuotantoprojektiryhmän Hali tekemän työn [Hal03a] pohjalta.

Pesätarkastusten tiedot on kerätty lomakkeilla. Talletettavat tiedot selviävät lomakkeilta ja asiakasta haastattelemalla. Käyttöliittymä tarjoaa rajapinnan tietokannan tauluihin, tarkistaa syötetyt tiedot ja tallettaa tiedot tietokantaan. Käyttöliittymän kautta tuotetaan säännönmukaisia raportteja tutkijoiden ja viranomaisten käyttöön.

1.2 Tärkeimmät vaatimukset

Järjestelmän on oltava turvallinen niin, että järjestelmään pääsee vain tunnuksella ja salasanalla. Tietokannasta on pystyttävä hakemaan, lisäämään ja muuttamaan tietoa ja lisäksi on pystyttävä tuottamaan lukuisa määrä erilaisia raportteja. Käyttöliittymän tulee olla mahdollisimman selkeä ja helppokäyttöinen ja loppukäyttäjältä ei voida odottaa teknisen arkkitehtuurin tuntemusta eikä muutakaan ”teknistä” osaamista. Lisäksi järjestelmän täytyy toimia oikeellisesti myös silloin kun käyttäjä tekee virheen, esim. niin että järjestelmä ei hyväksy virheellistä syötettä.

Tarkemmin järjestelmän vaatimukset on kuvattu vaatimusdokumentissa.

1.3 Erikoissanasto ja käytetyt lyhenteet

Black-box
Testausmenetelmä, jossa testaus tehdään ohjelmiston spesifikaation perusteella.

CVS
Concurrent Versions System. Versionhallintaohjelmisto, joka on luotu helpottamaan ohjelmistojen versionhallintaa.

Haliaeetus järjestelmä, järjestelmä
Näillä tarkoitetaan koko toteutettavaa järjestelmää, joka sisältää käyttöliittymän, tietokannan ja näiden välillä olevat toiminnallisuudet.

HTML
HyperText Markup Language. World Wide Webin, eli WWW:n julkaisukieli.

HTTP
Hypertext Transfer Protocol. Siirtokäytäntö, eli protokolla, jonka varaan WWW rakentuu. Hypertekstidokumenttien siirtoa verkossa tukeva komentokieli.

HTTPS
HTTP over Secure Sockets Layer. HTTP:n salakirjoitettu versio.

Istunto
Samalta selaimelta tuleva sarja kyselyjä, jotka tapahtuvat määrätyssä ajanjaksossa.

Java
Ohjelmointikieli, jota käytetään tämän projektin toteutuksessa.

J2EE-suunnittelumalli
J2EE-arkkitehtuuriin sovellettava suunnittelumalli, joita Sun on kehittänyt [Mic04].

JDBC-ajuri
Java DataBase Connectivity -ajuri, mahdollistaa tietokantakutsut palvelinsovelmista.

JDBC-yhteys
Hoitaa yhteyden tietokantaan, jonne järjestelmän tiedot on talletettu.

Käyttöliittymä
Se osa järjestelmästä, joka näkyy loppukäyttäjälle ja jolla järjestelmää käytetään.

Käyttötapaus
Käyttötapauksessa kuvataan käyttäjän tavoite jonkin päämäärän saavuttamiseksi, ja mahdollisimman yksityiskohtaiset tiedot tilanteen taustoista (tilatiedot). Käyttötapauksessa ei oteta mitään kantaa tekniseen toteutukseen. Käyttötapaukset ovat olennainen osa käyttöliittymäsuunnittelua.

Luokka
Java-ohjelmointikielessä yksi kokonaisuus, kuten taulu tietokannassa.

Luokkarakenne
Kuvaa järjestelmän teknisen (luokkien väliset suhteet) rakenteen korkealla tasolla.

Mallipohja
Sama kuin template. Dokumentin runko, joka sisältää tietyllä tavalla merkittyjä kohtia, joihin voidaan ohjelmallisesti lisätä vaihtuvia arvoja.

Merikotka
Merikotka (*Haliaeetus albicilla*) on Suomen suurin petolintu. Sen siipien väli on 190–240 cm ja pituus 76–94 cm. Merikotkalla on tasaruskea höyhenpuku ja suorakaiteen muotoiset siivet, jotka harottavat kärjistään. Pyrstö on lyhyt ja kiilamainen ja vanhoilla linnuilla valkea.

Merikotkakanta
Merikotkan biologinen kanta.

Merikotkatyöryhmä
Merikotkatyöryhmä on Torsten Stjernbergin johtama merikotkien tutkimusta ja suojelua tekevä työryhmä Maailman Luonnonsäätiössä (WWF).

Metodi
Java-luokan sisällä oleva aliohjelma, jota voidaan kutsua itse luokasta tai toisesta Java-luokasta.

Olio
Java-luokan ilmentymä.

Pakkaus
Pakkaus on tapa kerätä yhteen toisiinsa jollain tavalla liittyvät ohjelmaluokat.

Pesä
Merikotkan pesä on hyvin kookas risupesä suuren puun latvassa tai kallionkielekkeellä. Pesä voi olla myös vaihtopesä tai tekopesä.

Reviiri
Yhdellä reviirillä elää yksi merikotkapariskunta, jolla saattaa olla useampikin pesä reviirinsä alueella.

Servlet
Java-ohjelmointikielellä kirjoitettu palvelinsovelma, servletti.

Singleton
Suunnittelumalli, joka tarjoaa vain ainokaisen ilmentymän oliosta, kuten kirjassa [Gam95] on kuvattu.

SQL
Structured Query Language. Standardoitu kieli, jolla voidaan määrittää erilaisia tietokantaoperaatioita.

Tarkastus
Pesille tehdään tarkastuskäyntejä muutaman kerran vuodessa, jolloin saadaan tietoa mm. poikasten määrästä ja pesällä vallitsevista olosuhteista.

Template
Sama kuin mallipohja. Dokumentin runko, joka sisältää tietyllä tavalla merkittyjä kohtia, joihin voidaan ohjelmallisesti lisätä vaihtuvia arvoja.

Tietokanta
Jotain käyttötarkoitusta varten laadittu kokoelma toisiinsa liittyviä säilytettäviä tietoja. Tietokannan teknisiä ominaisuuksia ovat mm. tiedon riippumattomuus sitä käsittelevistä ohjelmista, tietojen samanaikainen käyttö, monipuoliset tiedonhakumahdollisuudet, tietojen suojaus, mutkikkaat riippuvuudet tietojen välillä ja automaattinen varmistus ja elpyminen häiriöistä.

Tilatieto
Käyttötapauksissa tilatiedot kertovat kaikki taustatekijät, jotka vaikuttavat käyttäjän tavoitteen saavuttamiseen.

TKTL
Helsingin yliopisto, Tietojenkäsittelytieteen laitos.

White-box
Rakenteellinen testausmenetelmä, eli testaus tehdään ohjelmiston sisäisen rakenteen perusteella.

WWF
WWF (World Wildlife Fund) eli Maailman Luonnonsäätiö on maailmanlaajuinen luonnonsuojelujärjestö, joka työskentelee luonnon monimuotoisuuden suojelemiseksi ja ekologisten toimintojen ylläpitämiseksi.

2 Suunnittelun ja toteutuksen rajoitteet

Tässä luvussa määritellään rajoitukset suunnittelulle ja toteutukselle. Rajoitukset koskevat teknistä toimintaympäristöä ja eräitä standardeja. Lisäksi on huomattava asiakkaan vaatimus siitä, ettei mitään merikotkiin liittyvää kriittistä suojelutietoa vuoda ulkopuolisille esim. dokumenttien kautta.

2.1 Noudatettavat standardit ja tarvittavat ohjelmat

Käyttäjän selaimen ja WWW-palvelimen välinen kommunikointi tapahtuu suojatulla HTTPS-protokollalla [Soc03]. Käyttäjälle näkyvissä HTML-sivuissa käytetään W3C:n [W3C03b] määrittelemää HTML 4.01-spesifikaatiota [W3C03a].

Haliaeetus-järjestelmän käyttöliittymä toimii Microsoft Internet ExplorerIE selaimen versiolla 6.0 tai uudemmalla [Mic03a]. Palvelimen tietokantana on Oracle 9i [Ora03].

2.2 Luokkakirjastot ja ajurit

Templatet on toteutettu FreeMarker-kirjastolla [Fre03], joka on julkaistu GNU General Public License:n [fsf03] alla.

Oracle 9i:n JDBC-ajurista [Mic03f] käytetään toteutusvaiheessa versiota 9.x for Java 1.4.2.

2.3 Ohjelmointikieli ja -tyyli

Ohjelmisto kirjoitetaan Sun Microsystemin kehittämällä Java-ohjelmointikielellä [Mic03d]. Koodin ulkoasussa pyritään noudattamaan Java Code Conventions –spesifikaatiota [Mic03b].

Koodi kommentoidaan niin, siitä voidaan generoida Javadoc-dokumentaatio [Mic03c].

Dokumentointi- ja kommentointikieli on suomi. Koska käyttöliittymä ja tietokanta on toteutettu suomeksi päädyttiin Java-luokissa käyttämään suomea ja englantia kuitenkin niin, että jos luokan nimi on englanniksi on myös muuttujien ja metodien kielenä englanti ja jos luokan nimi on suomeksi on myös muuttujien ja metodien kielenä suomi. Servleteissä luokkien, muuttujien ja metodien kielenä on suomi.

Versionhallintaan käytetään TKTL:n CVS-versionhallintajärjestelmää [CVS03].

3 Järjestelmän yleiskuvaus

Tässä luvussa kuvataan Haliaeetus-ohjelmistoa. Luvussa 4 kuvataan arkkitehtuuria siten, että lukija saa käsityksen koko ohjelmiston rakenteesta. Lisäksi luvun 4 aliluvuissa on tarkempi kuvaus luokista.

Järjestelmän arkkitehtuuri perustuu käyttäjän selaimen, servlet-palvelimen ja tietokantapalvelimen väliseen vuorovaikutukseen (ks. kuva1).

Järjestelmä on toteutettu kolmitasoarkkitehtuurilla, jonka osakokonaisuudet ovat: esityskerroksena toimii sovelluslogiikkakerroksella dynaamisesti generoitava Web-käyttöliittymä, sovelluslogiikan hoitavat Java-luokat ja tietovarastona toimii Oracle 9i-tietokanta.

Taso 1, Esityskerros

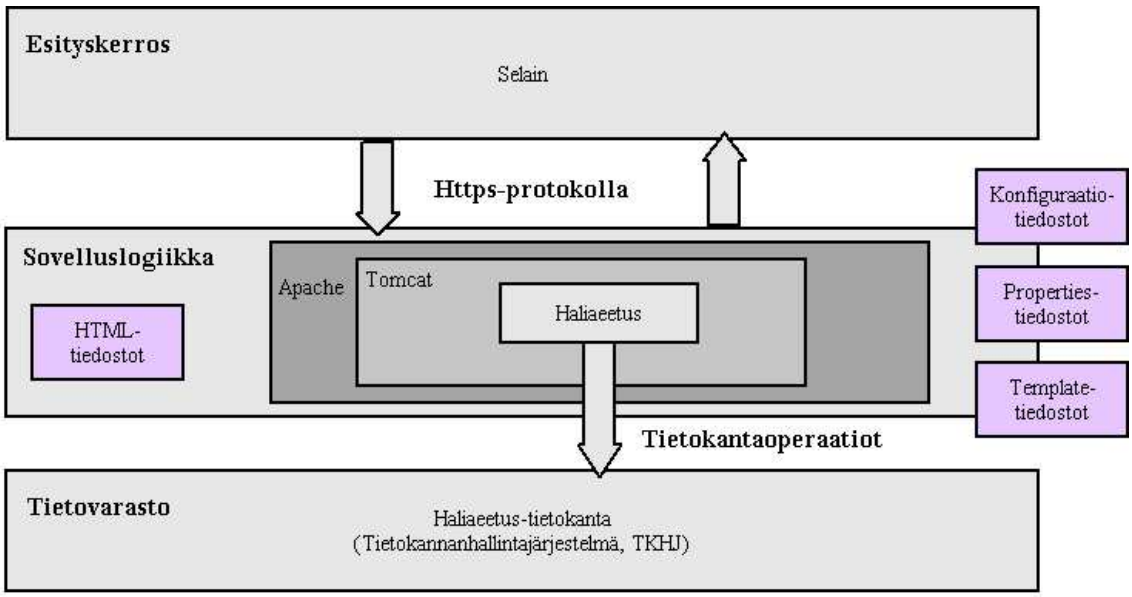
Tasolla 1 oleva selain on vuorovaikutuksessa sekä käyttäjän että tasolla 2 olevan ohjelmiston kanssa https-prokokollan kautta.

Taso 2, Sovelluslogiikka

Palvelinkoneella alkokrunni.cs.helsinki.fi (alias db.cs.helsinki.fi), pyörii Apache WWW-palvelin sekä Tomcat 4.1 servlet engine [Pro03]. Palvelimella sijaitsevat JDBC-ajuri, servletit, Java-luokat, JDBC-tietokantaluokat, Freemarker [Fre03] kirjastoluokat sekä staattiset template-, html- ja properties- tiedostot.

Apache-Tomcat –pari käyttää HTTPS-protokollaa [Soc03] kommunikoidessaan käyttäjän selaimen kanssa ja JDBC-ajuria [Mic03f] kommunikoidessaan Haliaeetus-tietokannan kanssa. Tietokanta on koneella bodbacka.cs.helsinki.fi.

Java-luokat [Mic03e] käyttävät tietokantaa JDBC-tietokantaluokkien avustuksella. Luokat myöskin generoivat dynaamista HTML:ää ja kommunikoivat käyttäjän selaimen kanssa. Servletit saavat tulostamiensa HTML-sivujen rungot template-tiedostoista, ja esitettävät tiedot Haliaeetus-tietokannasta.



Kuva 1: Yleiskuva järjestelmästä

Taso 3, Tietovarasto

Tasolla 3 on Haliaeetus-tietokanta, jossa täytyy olla vastaava JDBC-ajuri, joka tukee JDBC API:a.

4 Arkkitehtuurikuvaus

Tässä luvussa kuvataan ohjelmiston käyttämät tiedostot, niiden keskinäiset suhteet ja sijainti järjestelmässä. Ohjelmisto koostuu viidestä pakkauksesta: haliaeetus.db, haliaeetus.servlet, haliaeetus.helper, haliaeetus.command, haliaeetus.log ja muista tiedostoista, joita käytetään html-sivujen tuottamiseen ja kielitukeen ja asetuksiin. Tietokantaluokat ovat pakkauksessa haliaeetus.db (Luku 4.3) , servletit ovat pakkauksessa haliaeetus.servlet (Luku 4.4), sovelluslogiikkakomponenetit pakkauksessa haliaeetus.command (Luku 4.6), yleiset apuluokat pakkauksessa haliaeetus.helper (Luku 4.5), ja lokitukseen käytettävät luokat ovat pakkauksessa haliaeetus.log (Luku 4.7). Html-sivujen tuottamiseen käytetään mallipohjia eli template-tiedostoja (Luku 4.9.1) ja tyylitiedostoa (Luku 4.9.3). Kielituessa käytetään properties-tiedostoja (Luku 4.9.2).

Pakkauksen haliaeetus.helper luokka HaliProperties ja tietokantaluokat on otettu tipu4:stä [Tip03]. Luokan HaliProperties pohjana on tipu4:n TipuProperties, josta on poistettu joi-

tain metodeja. Myös tietokantaluokkia on hieman muutettu. Ainoastaan luokat DatabaseOperation ja DeleteOperation ovat muuttamattomina tipu4:stä.

4.1 Komponenttien väliset suhteet

Käyttäjän syöttämä informaatio lähetetään selaimen toimintopainikkeista pakkauksen hali.servlet HaliaeetusControllerServlet-servletille. Kaikki käyttöliitymän pyynnöt välitetään keskitetysti servletille, joka autentikoi käyttäjän ja välittää pyynnön eteenpäin sitä käsittelevälle pakkauksen haliaeetus.command komentoluokalle. Komentoluokat käyttävät pakkauksessa hali.db sijaitsevia tietokantaoperaatioihin erikoistuneita luokkia apuna operaatioisssa tietokantaan. Suoritettava toimenpide voi olla haku-, lisäys- tai muokkausoperaatio. Tietokantaoperaatiota varten tarvitsemansa metadatan luokat saavat luomalla ilmentymän Table-luokasta, joka käyttää apunaan tiedostoa tabledata.

Luokat palauttavat tietokantaluokilta saamansa vastauksen käyttäjälle generoimalla selaimelle uuden näytön. Näytön luonnissa luokat käyttävät apuna valmiita mallipohjia sivun ulkoasun määrittelemisessä ja tietojen oikeaan kenttään kohdentamisessa. Kielituki toteutetaan servleteissä, jotka hakevat tarvitsemansa erikieliset tekstit properties–tiedostoista (Luku 4.9.2).

Konfigurointitiedot laitetaan tiedostoon haliaeetus.config (Luku 4.5.1), josta niitä voidaan lukea.

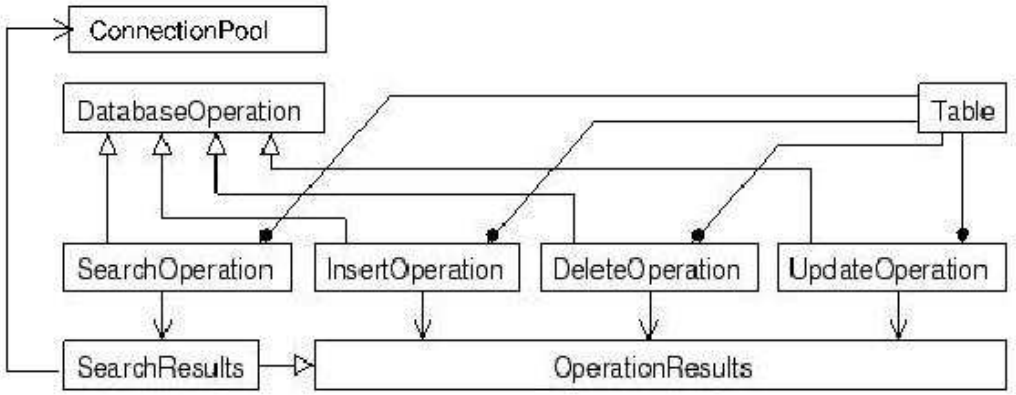
4.2 Luokkien väliset suhteet

Luokkamalli on suunniteltu ”Front Controller Servlet”- ja ”Command and Controller”-suunnittelumallin [Mic04] mukaisesti.

Pakkauksen hali.command luokat voivat käyttää pakkauksen hali.db ja hali.helper palveluja. Komentoluokat suorittavat saamalleen tiedolle tarkistuksia luokan CheckHelper aliluokkien avulla. Luokista otetaan tietokantayhteys pakkauksen hali.db luokan ConnectionPool avulla. Tietokantaluokat huolehtivat operaatioista tietokantaan ja palauttavat operaation tuloksen kutsuvalle luokalle. Tietokantaluokat käyttävät pakkauksen hali.helper luokan HaliProperties palveluja.

4.3 Pakkaus hali.db

Pakkaus **hali.db** otetaan käyttöön sellaisena kun Hali-projektiryhmä sen on koodannut. Täten tämä luku on suoraan lainattu Hali-projektin suunnitteludokumentista [Hal03c]. Ainoastaan *insert*- operaation suhteen saatetaan tehdä jotain muutoksia, jos huomataan että siihen on jäänyt virheitä. On vielä epäselvää kummalla puolella edellisen projektin *insert*-operaatiossa havaitut virheet olivat, *insert*-operaation vai servlettien puolella.



Kuva 2: Tietokantaoperaatioiden luokkarakenne ja tulostyypit

Pakkaus **hali.db** pitää sisällään tietokantaluokat (**kuva**), jotka suorittavat JDBC-yhteyden avulla haku, lisäys, muokkaus tai poisto-operaatioita tietokantaan. Operaatiot on toteutettu luokissa **SearchOperation**, **InsertOperation**, **UpdateOperation** ja **DeleteOperation**, jotka periytyvät yhteisestä abstraktista yliluokasta **DatabaseOperation**.

Operaatiot palauttavat kutsujalleen lisäys-, muokkaus- ja poistotapauksissa totuusarvon, joka indikoi halutun operaation onnistumista. Haku-operaatio palauttaa erillisen **SearchResults**-olion, jolla hakutuloksia voidaan tarkastella lähemmin.

Myös JDBC-yhteys tietokantaan hoidetaan tietokantaluokkien avulla. Yhteyksiä jaetaan yhteysvarannon (*connection pool*) avulla. Luokka **DatabaseOperation** tarjoaa tuen luokan **ConnectionPool** dynaamiselle yhteysjaolle.

4.3.1 ConnectionPool

ConnectionPool eli yhteysvaranto sisältää JDBC-tietokantayhteyksiä. Yhtäaikaisten yhteyksien maksimilukumäärä sekä käytettävä JDBC-ajuri ja tietokannan url ilmoitetaan tiedostossa **haliaeetus.config**, joka sijaitsee palvelinkoneella **alkokrunni.cs.helsinki.fi** kansiossa **/jserv/etc/**.

Konstruktori:

private ConnectionPool(String user, String pw)

> Konstruktori alustaa ja luo **ConnectionPool**-olion. Tarvittavia tietoja ovat muun muassa käytettävä ajuri, tietokannan url ja yhtäaikaisten yhteyksien maksimilukumäärä. Parametreina annettavat käyttäjätunnus ja salasana sijoitetaan luokan muuttujiin, joten kaikki yhteydet tietokantaan luodaan pelkästään näitä tunnuksia käyttäen. Koska servletit saavat käyttää vain luokan staattisia metodeja, konstruktori määritellään näkyvyydeltään yksityiseksi.

Julkiset metodit:

public static Connection requestConnection (String username, String password)

> Metodilla pyydetään tietokantaoperaatioon tarvittava yhteys. Metodia on tarkoitus käyttää siten, että ennen tietokantaoperaatioita servlet pyytää luokalta käyttäjälle yhteyden.
>
> Käyttäjän käyttäjätunnus ja salasana tarkistetaan joka kerta yhteyttä pyydettäessä. Jos tunnukset eivät ole kelvolliset, palautetaan *null*. Parametreja *username* (käyttäjätunnus) ja *password* (salasana) verrataan varannon tiedossa olevaan tietokannan käyttäjätunnukseen ja salasanaan. Kaikkien käyttäjien tulee käyttää samaa tietokantatunnusta kirjautuessaan Haliaeetus-järjestelmään, koska suunnitellussa varannossa on tuki vain yksille tunnuksille.
>
> Suoritettuaan kaikki saman pyynnön aikana tehtävät tietokantaoperaatiot, servletin tulisi palauttaa yhteys varantoon. Yhteys pysyy avoimena, vaikka se luovutetaankin muiden operaatioiden käyttöön.

public static void releaseConnection(Connection con)

Metodi palauttaa aiemmin varatun yhteyden takaisin yhteysvarantoon, jossa se on kaikkien käyttäjien käytettävissä.

public void run()

Säikeen tarvitsema metodi.

Yksityiset metodit:

private synchronized Connection getConnection()

Metodi hankkii yhteyden jollakin seuraavista keinoista:

1. Vapaita yhteyksiä löytyy varannosta: otetaan ensimmäinen vapaa yhteys sieltä käyttöön. Jos saatu yhteys oli ajurin puolesta suljettu, luodaan tilalle uusi yhteys ja yhteyden varausta yritetään uudelleen. Tässä tapauksessa yhteyden pyytäjä joutuu kilpailemaan muiden yhteyttä pyytämään tulleiden käyttäjien kanssa tasavertaisesti.

2. Vapaita yhteyksiä ei ole varannossa: Jos yhteyksien maksimimäärää ei ole vielä ylitetty, luodaan uusi yhteys ja kilpailutetaan se käyttäjien kesken. Jos yhteyksiä on jo maksimimäärä, jäädään odottamaan yhteyden vapautumista.

private void makeBackgroundConnection()

Metodi käynnistää säikeen muodostamaan yhteyttä silloin, kun yhteyksiä ei ole vapaana. Yhteyden muodostus tapahtuu säikeessä, koska yhteyden luominen voi kestää useamman sekunnin. Kun toiminto suoritetaan taustalla, voidaan palvella muita yhteyden pyytäjiä ja vapauttajia rinnalla.

private Connection makeNewConnection()

Metodi luo uuden yhteyden tietokantaan. Jos yhteyden luonti ei onnistunut, palautetaan *null*.

private synchronized void free(Connection connection)

Metodi palauttaa yhteysvarantoon kuuluvan yhteyden. Yhteys siirretään vapaiden yhteyksien joukkoon ja yhteyttä mahdollisesti odottaville ilmoitetaan vapautuneesta yhteydestä.

private synchronized int totalConnections()

Metodi laskee vapaiden ja käytettyjen yhteyksien lukumäärän.

private boolean checkUser(String user, String pw)

Metodi tarkastaa, että käyttäjän tunnus ja salasana vastaavat yhteysvarannon tiedossa olevaa käyttäjätunnusta ja salasanaa.

private synchronized void closeAllConnections()

Metodilla suljetaan kaikki yhteydet, niin vapaat kuin käytössä olevatkin. Metodia käytetään vain **ConnectionPool**-olion tuhoamisen yhteydessä.

private void closeConnections(Set connections)

Metodi sulkee kaikki annetussa yhteysjoukossa (vapaat/käytössä) olevat yhteydet. Metodia käyttää vain **closeAllConnections()**.

Pakkausnäkyvyyden metodit:

protected void finalize()

Lopetusmetodi kutsuu **closeAllConnections()**-metodia. Tällä tavoin varmistetaan, että **ConnectionPool**-olion tuhoamisen yhteydessä kaikki yhteydet tulevat suljetuiksi.

4.3.2 Table

Table-luokan avulla saadaan tietokantataulujen metadata eli nimet, attribuutit ja jokaiseen attribuuttiin liittyvät tiedot tekstitiedostosta **tabledata**. Tiedosto tabledata sijoitetaan toteutusympäristössä palvelinkoneelle **alkokrunni.cs.helsinki.fi** kansioon **/jserv/etc/**. Tiedosto sisältää taulujen tiedot siten, että jokainen tiedoston rivi sisältää yhden taulun tiedot muodossa

taulu;attribuutti:avain:null:type:kommentti;attribuutti:avain:...

missä

- **taulu** on taulun nimi,
- **attribuutti** on taulun sarakkeen eli attribuutin nimi,
- **avain** 'Y' jos attribuutti on taulun pääavain, muutoin 'N',
- **null** 'Y' jos attribuutti voi olla null, muutoin 'N',
- **type** on attribuutin tyyppi (**NUMBER**, **VARCHAR2**, **DATE**..),
- **kommentti** on attribuuttia koskeva kommentti.

Taulun nimen ja eri attribuuttien erottimena on puolipiste ja attribuuttien tietokenttien erottimena kaksoispiste.

Servlet hakee käynnistyessään jokaisen kohdetaulunsa **Table**-olion, ja säilyttää niitä ilmentymämuuttujissaan koko elinkaarensa ajan.

Sisäluokka:

private static class FieldInfo

> **FieldInfo** on **Table**-luokan staattinen sisäluokka, joka toimii yksinkertaisena taulun attribuuttia kuvaavana tietorakenteena.

Sisäluokan muuttujat:

int type

Muuttujan arvona on sarakkeen tyyppi.

String comment

Muuttujaan sijoitetaan sarakkeen kommentti.

boolean key

Muuttujan arvona on **true**, mikäli sarake on taulun pääavain.

boolean nullable

Muuttujan arvona on **true**, mikäli arvo voi olla **NULL**.

Muuttujat:

private String tableName

Muuttujassa on taulun nimi.

private SortedSet attributes

Muuttujassa on taulun attribuuttien nimet (**String**) aakkosjärjestettynä joukkona.

private Map fields

Muuttujassa on taulun attribuuttien tiedot attribuuttien nimien (**String**) osoittamina **Table.FieldInfo**-olioina.

Konstruktori:

private Table(String tableName, SortedSet attributes, Map fields)

Konstruktori luo **Table**-olion sijoittaen parametrit ilmentymämuuttujiinsa. Parametri **tableName** ilmoittaa tietokannan taulun nimen, **attributes** sisältää tietokannan attribuutit järjestyksessä ja **fields** kohdentaa jokaiseen attribuuttiin kyseisen attribuutin muut tiedot (avain, null, type, ja kommentti). Koska konstruktoria saa käyttää vain luokan staattinen metodi, konstruktori on näkyvyydeltään yksityinen.

Julkiset metodit:

public static Table getTable(String tableName)

Metodi palauttaa parametria vastaavan taulun tiedot. Metodin ensimmäisellä kutsukerralla ennen taulutietojen hakua haetaan **tabledata**-tiedostosta taulukuvaukset.

public String getName()

Metodi palauttaa taulun nimen.

public SortedSet getKeys()

Metodi palauttaa taulun attribuuteista ne, jotka ovat taulun pääavaimia. Avain tai avaimet palautetaan järjestettynä joukkona merkkijonoja (**SortedSet**). Merkkijonoilla järjestys on yleensä nouseva.

public SortedSet getAttributes()

Metodi palauttaa taulun attribuuttien nimet järjestettynä joukkona merkkijonoja. Merkkijonoilla järjestys on yleensä nouseva.

public int getType(String attr)

Metodilla voi selvittää parametrina annetun attribuutin tyypin (**NUMBER**, **VARCHAR2**, **DATE**...).

public String getComment(String attr)

Metodi palauttaa parametrina annettuun attribuuttiin liittyvän kommentin.

public boolean isKey(String attr)

Metodilla voidaan selvittää onko parametrina annettu attribuutti taulun avain.

public boolean isNullable(String attr)

Metodi palauttaa tiedon siitä, voiko parametrina annettu attribuutti olla null.

Yksityiset metodit:

private static FieldInfo parseField(StringTokenizer field)

Metodi **loadTabledata()** käyttää tätä metodia apunaan. Metodilla saadaan koottua attribuutin muut tiedot (avain, null, type, ja kommentti) **FieldInfo**-sisäluokan ilmentymään.

private static Map loadTableData()

Metodi lukee tiedostosta tabledata kaikkien tietokannassa käytettyjen taulujen kuvaukset. Tiedosto luetaan rivi riviltä, ja jokaisesta rivistä tehdään **Taulu**-olio.

4.3.3 OperationResults

OperationResults-luokkaa käytetään tietokantaoperaatioiden tulosten tutkimiseen. Tietokantaoperaatio palauttaa **OperationResults**-tyyppisen olion, jossa on operaation totuusarvoinen tulos ja mahdollinen virheilmoitus. Hakutuloksille luodaan oma tarkempi tulosolio, joka on tyyppiä **SearchResults**. Luokka **SearchResults** perii luokan **OperationResults** ominaisuudet.

Konstruktori:

public OperationResults()

Metodi luo tulosolion, jossa virhettä ei ole tapahtunut, eli tulosoliota luotaessa oletetaan tietokantaoperaation onnistuvan.

Julkiset metodit:

public void setErrorMessage(String msg)

Jos tietokantaoperaatio ei jostain syystä onnistunut, tietokantaluokka asettaa tällä metodilla virheilmoituksen tulosolioon, ja operaatio merkitään epäonnistuneeksi.

public boolean succeeded()

Servlet voi tämän metodin avulla selvittää tietokantaoperaation onnistumisen / epäonnistumisen. Jos metodi palauttaa false merkkinä operaation epäonnistumisesta, **getErrorMessage()**-metodia voidaan käyttää virheen tarkempaan tutkimiseen.

public String getErrorMessage()

Tällä metodilla saa kysyttyä operaation epäonnistumisen syyn. Palautettu teksti on tarkoitettu ainoastaan virheen raportointiin.

4.3.4 SearchResults

SearchResults on luokka, jolla hakuoperaation tuloksia voi selata. Luokka periytyy **OperationResults**-luokasta. **SearchResults**-luokka sisältää listan haussa löytyneiden kohdetaulujen riveistä. Servlet pääsee käsiksi hakutuloksiin luokan julkisilla metodeilla. Osa metodeista vaatii, että käyttäjän tarvitsee antaa parametreina myös tietokannan käyttäjätunnus ja salasana (koska nämä metodit puolestaan käyttävät luokkaa **ConnectionPool**).

Konstruktori:

public SearchResults(ResultSet rs, Table t)

Konstruktorin tehtävänä on kutsua yliluokan konstruktoria ja asettaa tietokantaoperaation tuottaman tulosjoukon **ROWID**:t listaan ja operaatiossa ollut kohdetaulu **SearchResults**-olion tiedoiksi.

Julkiset metodit:

public int getRowCount()

Metodi palauttaa tietokantahaussa löytyneiden rivien määrän.

public int getRowNumber()

Metodi palauttaa sen rivin numeron, missä kursori tällä hetkellä on. Palautettu arvo on väliltä [1..n], missä n on tulosjoukon rivien lukumäärä.

public SortedMap getCurrentRow(String username, String password)

Metodi palauttaa tämänhetkisen vastausjoukon rivin järjestettynä **SortedMap**-oliona (attribuutti -> arvo).

public SortedMap getNextRow(String username, String password)

Metodi palauttaa vastausjoukossa nykyistä kohtaa seuraavan rivin, ja kursoria siirretään yksi rivi eteenpäin. Jos kursori on jo viimeisellä rivillä, palautetaan null.

public SortedMap getPreviousRow(String username, String password)

Metodi palauttaa vastausjoukossa nykyistä kohtaa edeltävän rivin, ja kursoria siirretään yksi rivi taaksepäin. Jos kursori on jo ensimmäisellä rivillä, palautetaan null.

public SortedMap getAbsoluteRow(int rowNum, String username, String password)

Metodi palauttaa vastausjoukosta parametrilla **rowNum** määritellyn rivin. Haluttu rivi annetaan väliltä [1..n], missä n on tulosjoukon rivien lukumäärä. (Haussa löytyneiden rivien määrän saa selville luokan metodilla **getRowCount()**).

public SortedMap getAbsoluteRow(String rowId, String username, String password)

Metodi palauttaa vastausjoukosta parametrilla **rowId** määritellyn rivin.

public void removeRow(ROWID rowId)

Metodi poistaa (tietokannasta juuri poistetun) rivin myös SearchResults-olion hakutuloslistasta. Jos riviä ei ole, ei tehdä mitään. Metodia kutsutaan servletistä käsin, jotta juuri poistettu tietue ei näkyisi vastausjoukossa.

public void removeRow(String rowId)

Metodi poistaa (tietokannasta juuri poistetun) rivin myös SearchResults-olion hakutuloslistasta. Metodi kutsuu **removeRow(ROWID rowId)**:tä. Metodia kutsutaan servletistä käsin, jotta juuri poistettu tietue ei näkyisi vastausjoukossa.

Yksityiset metodit:

private void setResultSet(ResultSet rs)

Metodi sijoittaa haussa löydettyjen rivien **ROWID**:t listaan. Tämän jälkeen vastauskursori suljetaan.

private SortedMap getAbsoluteRow(ROWID rowId, String username, String password)

Metodi palauttaa käyttäjän määrittelemän rivin järjestettynä **SortedMap**-oliona. Luokan julkiset rivinhakumetodit käyttävät tätä metodia riviä hakiessaan.

4.3.5 DatabaseOperation

Luokka **DatabaseOperation** on abstrakti yleistys kaikista mahdollisista servlettien tietokantaan kohdistamista operaatioista. Jokainen tietokantaoperaatio on oltava valmisteltavissa siten, ettei joka kerta kyseistä operaatiota tiettyyn tauluun kohdistettaessa tarvitse kyselylauseketta kääntää uudestaan. Siksi jokaisen tietokantaoperaation on toteutettava tietty tietokantaoperaatioiden hallinnalle yhteinen rajapinta.

Jokaisella operaatiolla (poislukien **SearchOperation**) oletetaan olevan yksi tiettyyn tauluun kohdistuva valmisteltu lauseke (**java.sql.PreparedStatement**), jollaisen ilmentymää **DatabaseOperation**-luokka säilyttää jokaista ko. operaation suoritukseen tarjottua tietokantayhteyttä kohti. Näin täytyy tehdä, sillä valmistellut lausekkeet ovat sidottuja siihen yhteyteen, jolla valmistelu on alun perin suoritettu.

Ainoa asia, mitä geneerinen tietokantaoperaatio ei voi määritellä, on SQL-lause, joka määrää mitä kyseinen operaatio käytännössä tekee. Siksi jokaisen erikoistuneen operaatioluokan on toteutettava luokan **DatabaseOperation** abstrakti metodi **getSQLString()**.

Muuttujat:

protected OperationResults result

Operaation tulos talletetaan muuttujaan **result**. Toteuttava tietokantaoperaatio voi missä tahansa vaiheessa tämän muuttujan avulla asettaa operaation tilan epäonnistuneeksi, jolloin se raportoidaan käyttäjälle.

protected Table table

Muuttujaan **table** laitetaan operaation kohdetaulun ilmentymä.

private Map statementMap

Muuttuja statementMap sisältää jokaiselle käytössä olevalle tietokantayhteydelle (**java.sql.Connection**) käännetyt SQL-lauseet (**java.sql.PreparedStatement**).

Konstruktori:

protected DatabaseOperation(Table table)

Konstruktori luo geneerisen tietokantaoperaation, joka kohdistuu parametrin table määrittelemään tietokantatauluun.

Pakkausnäkyvyyden metodit:

protected final PreparedStatement getPreparedStatement(Connection con)

Metodi palauttaa annetulle yhteydelle valmistellun tätä operaatiota vastaavan lausekkeen. Mikäli valmistelua ei juuri tälle yhteydelle vielä ole tehty, se tehdään metodia kutsuttaessa ja talletetaan tämän operaation ilmentymään käytettäväksi juuri tälle yhteydelle tämän operaation tulevia suorituksia varten.

protected int getResultSetType()

Metodi palauttaa operaation vaatiman/salliman vastausjoukon kursorityypin, jonka tulee olla yksi luokan java.sql.ResultSet arvoista

TYPE_FORWARD_ONLY, TYPE_SCROLL_INSENSITIVE tai **TYPE_SCROLL_SENSITIVE**. Oletuksena palautetaan arvo **TYPE_FORWARD_ONLY**.

protected int getResultSetConcurrency()

Metodi palauttaa operaation vaatiman/salliman tietueiden eristyneisyysasteen, jonka tulee olla toinen **java.sql.ResultSet** luokan arvoista **CONCUR_READ_ONLY** tai **CONCUR_UPDATABLE**. Oletuksena palautetaan arvo **CONCUR_READ_ONLY**.

Abstraktit metodit:

protected abstract String getSQLString()

Metodi palauttaa toteuttavan operaation SQL-kielisen lausekkeen, johon on ilmentymää luotaessa luotu valmis parametrilista annetun kohdetaulun avainjoukon avulla.

4.3.6 SearchOperation

Luokka **SearchOperation** perii luokan **DatabaseOperation**. **SearchOperation** suorittaa **SELECT**-lauseella haun tietokannan tauluun annetuilla attribuuttimaskeilla ja muilla hakuehdoilla. Attribuuttimaskivertailut tehdään **LIKE**-operaattoreilla, joten käyttäjät voivat itse syöttää jokerimerkkejä hakukenttiin. Muissa hakuehdoissa voi käyttää mitä tahansa operaattoreita. Tulokset palautetaan erillisessä **SearchResults**-oliossa.

Keskustelu hakuoperaation kanssa on hyvin suoraviivaista. Ensin operaatio konstruoidaan antamalla parametriksi kohdetaulu. Tämän jälkeen koostetaan hakuarvot kartaksi ja muodostetaan mahdollinen muut hakuehdot määrittelevä merkkijono ja kutsutaan niillä metodia **executeSearch()**, joka suorittaa itse operaation. Tuloksena käyttäjä saa erillisen tulosolion **SearchResults**.

Konstruktori:

public SearchOperation(Table table)

Konstruktori luo uuden hakuoperaation, joka kohdistuu parametrin table määrittelemään tietokantatauluun.

Julkiset metodit:

public OperationResults executeSearch(Connection con, Map values, String where)

Metodi suorittaa hakuoperaation kohdetauluun annetulla yhteydellä **con** käyttäen hakuehtoina hakuarvoja **values** ja muita hakuehtoja **where**. Muut hakuehdot määrittelevä merkkijono **where** voi sisältää mitä tahansa **SQL**:n **WHERE**-lauseessa sallittuja hakuehtoja. Kutsu palauttaa haun tuloksena **SearchResults**-olion.

public OperationResults executeSearch(Connection con, Map values, String where, String sortKey)
Metodi suorittaa hakuoperaation kohdetauluun annetulla yhteydellä **con** käyttäen hakuehtoina hakuarvoja **values** ja muita hakuehtoja **where**. Muut hakuehdot määrittelevä merkkijono **where** voi sisältää mitä tahansa **SQL**:n **WHERE**-lauseessa sallittuja hakuehtoja. Tulokset järjestetään kohdetaulun sarakkeen **sortKey** mukaan. Kutsu palauttaa haun tuloksena **SearchResults**-olion.

Yksityiset metodit:

private String escapeQuotes(String s)
Metodi palauttaa parametrinaan saamansa merkkijonon **SQL**-lauseeseen sopivassa muodossa, jossa mahdolliset heittomerkit on kahdennettu, jotta niitä ei tulkittaisi merkkijonoskalaarin erottimiksi.

Pakkausnäkyvyyden metodit:

protected String getSQLString(Map values, String where, String sortKey)

Metodi palauttaa hakulausekkeen muotoa

SELECT ROWID FROM *taulu*
WHERE *attr1* LIKE '*value1*'
AND *attr2* LIKE '*value2*'
...
AND *attrN* LIKE '*valueN*'
AND (*where*)
ORDER BY *sortKey*

Kentät *attr?* ovat kohdetaulun attribuutteja (parametrin *values* avaimia). Ken-

tät *value?* ovat hakuarvoja (parametrin *values* annettua avainta vastaavia arvoja). Jos hakuarvo on *null*, attribuuttia ja arvoa ei lisätä kyselyyn. Merkkijono *where* on muut hakuehdot määrittelevä merkkijono, joka voi sisältää mitä tahansa **SQL**:n **WHERE**-lauseessa sallittuja hakuehtoja, ja se lisätään kyselyyn vain jos se ei ole *null*. Lause **ORDER BY** lisätään vain, mikäli *sortKey* ei ole *null*.

4.3.7 InsertOperation

Luokka **InsertOperation** perii luokan **DatabaseOperation**. Lisäysoperaatiolla voidaan lisätä yksi rivi määrättyyn tietokannan tauluun. Lisäys onnistuu, mikäli taulussa ei vielä ole samoilla avaimilla varustettua tietuetta, taulun eheys- ja avainrajoitteet toteutuvat eikä tiedonsiirtovirheitä tapahdu. Tulokset palautetaan **OperationResults**-oliona.

Keskustelu lisäysoperaation kanssa on hyvin suoraviivaista. Ensin operaatio konstruoidaan antamalla parametriksi kohdetaulu. Tämän jälkeen koostetaan arvot kartaksi ja kutsutaan niillä metodia **executeInsert()**, joka suorittaa itse operaation. Kutsu palauttaa lisäysoperaation tuloksena **OperationResults**-olion.

Konstruktori:

public InsertOperation(Table table)

Konstruktori luo uuden lisäysoperaation, joka kohdistuu parametrin **table** määrittelemään tietokantatauluun.

Julkiset metodit:

public OperationResults executeInsert(Connection con, Map values)

Metodi lisää rivin kohdetauluun arvoilla *values*. Kyseisestä arvokartasta tulee löytyä arvot ainakin kaikille pääavaimille sekä muille '**NOT NULL**' -attribuuteille.

Yksityiset metodit:

private void setParameters(PreparedStatement stmt, Map values)

Metodi kopioi annetut arvot **SQL**-lausekkeeseen. Tätä kutsutaan vain metodin **executeInsert()** sisältä.

private Date StringToDate(String strDate)

Koska käyttäjät ilmoittavat päiväyksen merkkijonona, tarvitaan apumetodi, joka muuntaa merkkijonona annetun päiväyksen **strDate java.sql.Date**-olioksi.

Pakkausnäkyvyyden metodit:

protected String getSQLString()

Metodi palauttaa **SQL**-lauseen muotoa

INSERT INTO ***taulu*** **(*attr1*, *attr2*, ..., *attrN*)**
VALUES (?, ?, ..., ?)

Kentät *attrN* ovat kohdetaulun attribuutteja.

4.3.8 UpdateOperation

Luokka **UpdateOperation** perii luokan **DatabaseOperation**. Päivitysoperaatio muuttaa jonkin jo olemassa olevan rivin attribuuttien arvoja. Operaatio onnistuu, mikäli kohdetaulusta löytyy annetun **ROWID**:n osoittama rivi eikä tiedonsiirtovirheitä tapahdu.

Keskustelu päivitysoperaation kanssa on hyvin suoraviivaista. Ensin operaatio konstruoidaan antamalla parametriksi kohdetaulu. Tämän jälkeen koostetaan arvot kartaksi ja kutsutaan niillä metodia **executeUpdate**, joka saa parametrikseen uusien attribuuttien arvojen lisäksi päivitettävän tietueen **ROWID**:n. Kutsu palauttaa lisäysoperaation tuloksena **OperationResults**-olion. Päivitykset tehdään vain annetuille attribuuteille.

Muuttujat:

private SimpleDateFormat dateFormat

Muuttujaan sijoitetaan tietokannassa käytettävä päiväysten esitysmuoto.

Konstruktori:

public UpdateOperation(Table table)

Konstruktori luo uuden päivitysoperaation, joka kohdistuu parametrin table määrittelemään tietokantatauluun.

Julkiset metodit:

public OperationResults executeUpdate(Connection con, Map values, String rowId)

Metodi suorittaa päivitysoperaation taulun riville *rowId* muuttaen tämän rivin attribuuttien arvot kartan *values* mukaisiksi. Mikäli **Map**-olio *values* ei sisällä jonkin attribuutin arvoa, kyseinen attribuutti jätetään päivittämättä ja näin ollen sen arvo ei muutu.

Yksityiset metodit:

private void setParameters(PreparedStatement stmt, Map values,String rowId)

Metodi kopioi uudet attribuuttien arvot sekä **ROWID**:n kyselylausekkeeseen. Annetut arvot korvaavat kaikki vanhat arvot. Mikäli arvokartasta *values* ei jotain arvoa löydy, kyseiseen kohtaan asetetaan arvo **NULL**. Eheysrajoitteen **NOT NULL** sisältävien kenttien kohdalla tästä aiheutuu väistämättä poikkeus ja toiminnon epäonnistuminen.

Käytännössä kaikkien kenttien päivityksestä ei kuitenkaan synny ongelmaa, sillä päivitysoperaatiota suoritettaessa servletin kutsuolio **javax.servlet.HttpServletRequest** sisältää tietueen kaikkien attribuuttien arvot – myös vanhat arvot niistä kentistä, joita käyttäjä ei ole muuttanut.

Pakkausnäkyvyyden metodit:

protected String getSQLString()

Metodi palauttaa **SQL**-lauseen muotoa

UPDATE *taulu*
SET *attr1* = ?, *attr2* = ?, ..., *attrN* = ?
WHERE ROWID = ?

Kentät *attr?* ovat kohdetaulun attribuutteja.

4.3.9 DeleteOperation

Luokka **DeleteOperation** perii luokan **DatabaseOperation**. Poisto-operaatio poistaa yhden rivin kohdetaulusta. Operaatio onnistuu, mikäli taulusta löytyy annettu rivi eikä tiedonsiirtovirheitä tapahdu.

Keskustelu poisto-operaation kanssa on hyvin suoraviivaista. Ensin operaatio konstruoidaan antamalla parametriksi kohdetaulu. Tämän jälkeen poisto-operaatio käynnistetään kutsumalla metodia **executeDelete()**, joka saa parametrinaan poistettavan rivin **ROWID**:n. Kutsu palauttaa poisto-operaation tuloksena **OperationResults**-instanssin.

Luokka **DeleteOperation** on toteutettu tämän suunnitteludokumentin määritelmän mukaan, mutta sitä ei käytetä ainakaan nyt toteutettavassa versiossa Haliaeetus-järjestelmästä.

Konstruktori:

public DeleteOperation(Table table)

Konstruktori luo uuden poisto-operaation, joka kohdistuu parametrin *table* määrittelemään tietokantatauluun.

Julkiset metodit:

public OperationResults executeDelete(Connection con, String rowId)

Metodi poistaa kohdetaulusta rivin annetulla **ROWID**:llä.

Yksityiset metodit:

private void setParameters(PreparedStatement stmt, String rowId)

Metodi kopioi **ROWID**:n arvon **SQL**-lausekkeeseen.

Pakkausnäkyvyyden metodit:

protected String getSQLString()

Metodi palauttaa **SQL**-lauseen muotoa

DELETE FROM *taulu* WHERE ROWID = ?

4.4 Pakkaus haliaeetus.servlet

Tässä paketissa on vain yksi luokka, **HaliaeetusControllerServlet**, joka on keskitetty komponentti käyttöliittymältä tulevien *GET*- ja *POST*-pyyntöjen käsittelyyn.

Luokka autentikoi käyttäjän, tarkistaa käyttäjän istunnon (session) voimassaolon ja ohjaa pyynnön kutsutulle **CommandDispacher**-luokalle.

4.4.1 HaliaeetusControllerServlet

HaliaeetusControllerServlet-luokka vastaanottaa pyynnöt käyttöliittymältä. **HaliaeetusControllerServlet** perii luokan **javax.servlet.http.HttpServlet**.

Julkiset metodit:

private void processRequest(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException

Prosessoi pyynnön.

Metodi tarkistaa ensiksi, onko käyttäjällä validi sessio. Mikäli ei, niin käyttäjä ohjataan sisäänkirjautumissivulle, muutoin käyttäjälle näytetään käyttöliittymän pyytämä sivu. Metodi luo ilmentymän **CommandDispather**-luokasta paketista **haliaeetus.command** ja delegoi pyynnön eteenpäin käsiteltäväksi.

public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException

Ohjaa *GET*-pyynnön *processRequest*-metodille.

public void doPost (HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException

Ohjaa *POST*-pyynnön *processRequest*-metodille.

public void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException

Ohjaa servletille tulevat suorat pyynnöt *processRequest*-metodille.

private boolean sessionValid(HttpServletRequest request, HttpServletResponse response)

Tarkastaa, onko käyttäjällä validi sessio.

private void translateParamsToAttributes(HttpServletRequest request, HttpServletResponse response)

Muuntaa parametrina annetun pyynnön sisältämät parametrit attribuuteiksi.

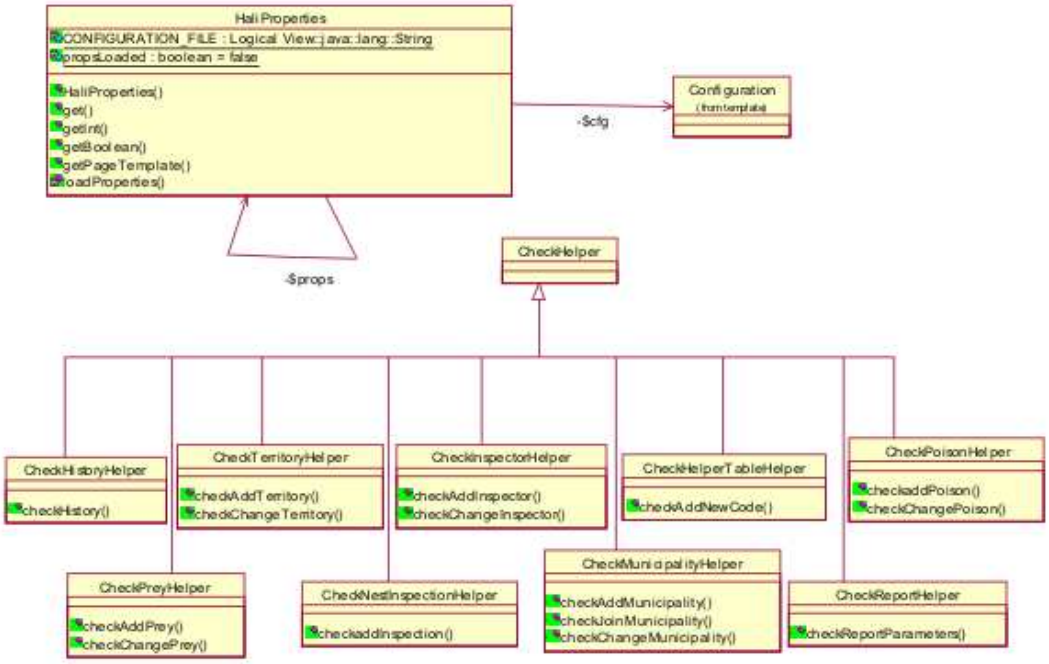
4.5 Pakkaus haliaeetus.helper

Pakkauksen **haliaeetus.helper** luokat suorittavat erilaisia järjestelmän tarvitsemia aputoimintoja.

Paketin **haliaeetus.helper** luokat ja niiden suhteet ilmenevät luokkakaaviosta kuvassa 3.

4.5.1 HaliProperties

Haliaeetus-ohjelmiston asetustiedot löytyvät tiedostosta **haliaeetus.config**. **HaliProperties**-luokan avulla tiedostosta saadaan luettua konfigurointitietoja muille luokille, kuten esimerkiksi käytettävä ajuri JDCB-yhteydelle.



Kuva 3: Luokkakaavio haliaeetus.helper-paketista

HaliProperties perii Javan oman valmiin luokan **java.util.Properties**. Yliluokan metodeja käytetään avain-arvo merkkijonoparien lukemiseen tiedostosta. Kun luokkaa käytetään sen julkisin, staattisten metodein ensimmäisen kerran, asetustiedot luetaan tiedostosta **haliaeetus.config**.

Toteutusympäristössä konfigurointitiedot sisältävä **haliaeetus.config** sijoitetaan palvelinkoneelle **alkokrunni.cs.helsinki.fi** kansioon /jserv/etc/.

Konstruktori:

private HaliProperties()

> Konstruktorin tehtävänä on kutsua yliluokan konstruktoria. Konstruktori määritellään yksityiseksi eli luokasta ei voi luoda ilmentymiä, sillä luokkaa käytetään kirjastoluokan tavoin staattisten metodien avulla.

Julkiset metodit:

public static String get(String key)

Metodi palauttaa parametrilla *key* asetustiedostosta löytyvän arvon. Jos parametria key ei löydy asetustiedostosta, palautetaan null.

public static getInt(String key)

Metodi palauttaa parametrilla *key* asetustiedostosta löytyvän arvon, mutta muutettuna kokonaisluvuksi. Jos annetulla parametrilla ei löytynyt arvoa tiedostosta tai arvoa ei voida tulkita kokonaisluvuksi, palautetaan *-1*.

public static getBoolean(String key)

Metodi palauttaa parametrilla *key* asetustiedostosta löytyvän arvon, mutta muutettuna totuusarvoksi. Jos annetulla parametrilla ei löytynyt arvoa tiedostosta tai arvoa ei voida tulkita totuusarvoksi, palautetaan *false*.

public static Template getPageTemplate(String templateName)

Metodi palauttaa kutsujalleen **Template**-olion, joka vastaa parametrissa määriteltyä templatetiedostoa. Parametrina *TemplateName* annettavassa mallipohjan tiedostonimessä täytyy olla mukana tiedoston ftl-pääte.

Yksityiset metodit:

private static void loadProperties()

Lukee asetustiedot tiedostosta /jserv/etc/haliaeetus.config.

4.5.2 CheckHelper

CheckHelper-luokka tarjoaa sen aliluokille tarvittavat yhteiset apumetodit. Luokassa tulee olemaan pakkausnäkyviä tarkistusmetodeita, joita tarvitaan useammassa kuin yhdessä aliluokassa ja joissa tarkistetaan yksitäisten arvojen oikeellisuuksia. Näiden metodien tarkempi suunnittelu siirtyy toteutuksen yhteyteen.

CheckHelper aliluokkien tehtävänä on tarkastaa **Command**-luokkien parametreina lähettämien arvojen oikeellisuus ja ilmoittaa näille tarkistuksen tuloksesta. Tarkastettavat arvot ovat raja-arvoja, joiden sisällä syötearvon tulee olla. Kullekin tarkastustoimia vaativalle **Command**-luokalle on siis oma aliluokka tarkastusta varten.

4.5.3 NestInspectionHelper

Luokan tehtävänä on tarkastaa **NestInspectionCommand**- ja **RepairNestInformationCommand**- luokkien parametreina lähettämien arvojen oikeellisuus ja ilmoittaa tuloksesta *boolean* arvolla.

Julkinen metodi:

public static boolean checkNestChangeableTable(HttpServletRequest request, Table taulu, int pesa_id, Connection con)

> Tämä metodi tarkistaa vatko pesamuuttuva-taulun tiedot muuttuneet uuden tarkastuksen yhteydessä. Jos muutoksia on tullut palautetaan **true**, muuten palautetaan **false**.

public static boolean checkNest(Properties properties, SimpleSequence sekvenssi, ResourceBundle kieli, Connection con)

> Metodi, jota kutsutaan **NestInspectionCommand** ja **RepairNestInfoCommand**-luokista välittämällä metodille parametreina pyyntö *Properties* ja tietokantayhteys *con*, jota luokka tarvitsee tietokantakyselyitä varten. Jos tarkastuksen tulos on hyväksyttävä, niin metodi palauttaa kutsuneelle luokalle *boolean* arvon *true*, muussa tapauksessa lisätään virheilmoitukset **sekvenssiin** ja palautetaan luokalle arvo *false*.

4.5.4 CheckPoisonHelper

Luokka on **CheckHelper**-luokan aliluokka ja sen tehtävänä on tarkastaa **PoisonCommand**-luokan parametreina lähettämien arvojen oikeellisuus ja ilmoittaa tuloksesta *boo-*

lean arvolla.

Julkiset metodit:

Metodit, jota kutsutaan **PoisonCommand**-luokasta välittämällä metodeille parametreina datamallin *datamap* ja tietokantayhteyden *con*, jota metodit tarvitsevat tietokantakyselyitä varten. Jos tarkastuksen tulos on hyväksyttävä, niin metodit palauttavat kutsuneelle luokalle *boolean* arvon *true*, muussa tapauksessa tehdään datamallin virhekohtiin merkinnät ja palautetaan luokalle arvo *false*.

public boolean checkAddPoison(Map datamap, Connection con)

Metodin tehtävänä on tarkistaa lisättävät myrkkytiedot.

public boolean checkChangePoison(Map datamap, Connection con)

Metodin tehtävänä on tarkistaa myrkkytietoihin tehtävät muutokset.

4.5.5 CheckPreyHelper

Luokka on **CheckHelper**-luokan aliluokka ja sen tehtävänä on tarkastaa **PreyCommand**-luokan parametreina lähettämien arvojen oikeellisuus ja ilmoittaa tuloksesta *boolean* arvolla.

Julkiset metodit:

Metodit, joita kutsutaan **PreyCommand**-luokasta välittämällä metodeille parametreina datamallin *datamap* ja tietokantayhteyden *con*, jota metodit tarvitsevat tietokantakyselyitä varten. Jos tarkastuksen tulos on hyväksyttävä, niin metodit palauttavat kutsuneelle luokalle *boolean* arvon *true*, muussa tapauksessa tehdään datamallin virhekohtiin merkinnät ja palautetaan luokalle arvo *false*.

public boolean checkAddPrey(Map datamap, Connection con)

Metodin tehtävänä on tarkistaa lisättävät saalistiedot.

public boolean checkChangePrey(Map datamap, Connection con)

Metodin tehtävänä on tarkistaa saalistietoihin tehtävät muutokset.

4.5.6 CheckHistoryHelper

Luokka on **CheckHelper**-luokan aliluokka ja sen tehtävänä on tarkastaa **HistoryCommand**-luokan parametreina lähettämien arvojen oikeellisuus ja ilmoittaa tuloksesta *boolean* arvolla.

Julkiset metodit:

Metodit, joita kutsutaan **HistoryCommand**-luokasta välittämällä metodeille parametreina datamallin *datamap* ja tietokantayhteyden *con*, jota metodit tarvitsevat tietokantakyselyitä varten. Jos tarkastuksen tulos on hyväksyttävä, niin metodit palauttavat kutsuneelle luokalle *boolean* arvon *true*, muussa tapauksessa tehdään datamallin virhekohtiin merkinnät ja palautetaan luokalle arvo *false*.

public boolean checkAddHistory(Map datamap, Connection con)

Metodin tehtävänä on tarkistaa lisättävät historiatiedot.

public boolean checkChangeHistory(Map datamap, Connection con)

Metodin tehtävänä on tarkistaa historiatietoihin tehtävät muutokset.

4.5.7 CheckInspectorHelper

Luokka on **CheckHelper**-luokan aliluokka ja sen tehtävänä on tarkastaa **InspectorsCommand**-luokan parametreina lähettämien arvojen oikeellisuus ja ilmoittaa tuloksesta *boolean* arvolla.

Julkiset metodit:

Metodit, joita kutsutaan **InspedtorsCommand**-luokasta välittämällä metodeille parametreina datamallin *datamap* ja tietokantayhteyden *con*, jota metodit tarvitsevat tietokantakyselyitä varten. Jos tarkastuksen tulos on hyväksyttävä, niin metodit palauttavat kutsuneelle luokalle *boolean* arvon *true*, muussa tapauksessa tehdään datamallin virhekohtiin merkinnät ja palautetaan luokalle arvo *false*.

public boolean checkAddInspector(Map datamap, Connection con)

Metodin tehtävänä on tarkistaa onko lisättävässä tarkastajan nimessä tai tunnuksessa jotain vikaa.

public boolean checkChangeInspector(Map datamap, Connection con)

Metodin tehtävänä on tarkistaa tarkastajan tietoihin tehtävät muutokset.

4.5.8 CheckTerritoryHelper

Luokka on **CheckHelper**-luokan aliluokka ja sen tehtävänä on tarkastaa **TerritoryCommand**-luokan parametreina lähettämien arvojen oikeellisuus ja ilmoittaa tuloksesta *boolean* arvolla.

Julkiset metodit:

Metodit, joita kutsutaan **TerritoryCommand**-luokasta välittämällä metodeille parametreina datamallin datamap ja tietokantayhteyden *con*, jota metodit tarvitsevat tietokantakyselyitä varten. Jos tarkastuksen tulos on hyväksyttävä, niin metodit palauttavat kutsuneelle luokalle *boolean* arvon *true*, muussa tapauksessa tehdään datamallin virhekohtiin merkinnät ja palautetaan luokalle arvo *false*.

public boolean checkAddTerritory(Map datamap, Connection con)

Metodin tehtävänä on tarkistaa onko lisättävässä tarkastajan nimessä tai tunnuksessa jotain vikaa.

public boolean checkChangeTerritory(Map datamap, Connection con)

Metodin tehtävänä on tarkistaa tarkastajan tietoihin tehtävät muutokset sallittuja.

4.5.9 CheckReportHelper

Luokka on **CheckHelper**-luokan aliluokka ja sen tehtävänä on tarkastaa **ReportCommand**-luokan parametreina lähettämien arvojen oikeellisuus ja ilmoittaa tuloksesta *boolean* arvolla.

Julkiset metodit:

public boolean checkReportParameters(Map datamap, Connection con)

> Metodi, jota kutsutaan **ReportCommand**-luokasta välittämällä metodille parametreina datamallin *datamap* ja tietokantayhteyden *con*, jota metodi tarvitsee tietokantakyselyitä varten. Metodi tarkistaa voidaanko käyttäjän syöttämillä parametreilla muodostaa raporttia eli onko esimerkiksi vuosiluku olemassa tai löytyykö kannasta kyseistä vuotta. Jos tarkastuksen tulos on hyväksyttävä, niin metodi palauttaa kutsuneelle luokalle *boolean* arvon *true*, muussa tapauksessa tehdään datamallin virhekohtiin merkinnät ja palautetaan luokalle arvo *false*.

4.5.10 Coords

Luokka **Coords** on yhteiskäyttöinen luokka kordinaattien laskentaan. Sen avulla yhtenäiskordinaatit tai astekoordinaatit voidaan laskea toisistaan ja sen avulla saadaan myös muodostettua desimaalikordinaatit. Luokkaa käytetään pesavakio-tauluun lisättävien kordinaattitietojen saamiseksi.

4.5.11 MunicipatyHelper

Luokka tarjoaa kantaan vietävien attribuuttien arvojen tarkistusmetodit luokalle **MunicipalityCommand**.

Konstruktori:

public MunicipalityHelper()

Julkiset metodit:

public static Map checkMunicipalityInsert(Map dataModel, Connection con, ResourceBundle language, String operation)

Tarkistaa voidaanko uusi kunta lisätä kunta-tauluun ja jos voidaan, niin palautetaan kantaan vietävät arvot **Map**-oliossa, muutoin palauttaa arvon *null*. Tämä metodi tarkistaa vain lisäykseen liittyviä seikkoja, kuten onko kuntatunnus käytössä.

public static Map checkMunicipalityValues(Map dataModel, ResourceBundle language, String operation)

Tarkistaa kannan kunta-tauluun vietävät arvot. Jos arvot olivat ok, niin palautetaan ne Map-oliossa, muutoin palautetaan arvo null. Operaatiosta (*insertOperation*, *updateOperation*, *municipalityJoin*) riippuen, virheilmoitukset lisätään operaatiokohtaisiin **Map**-olioihin.

public static String checkDecimal(String des, int integer_length, int decimal_length)

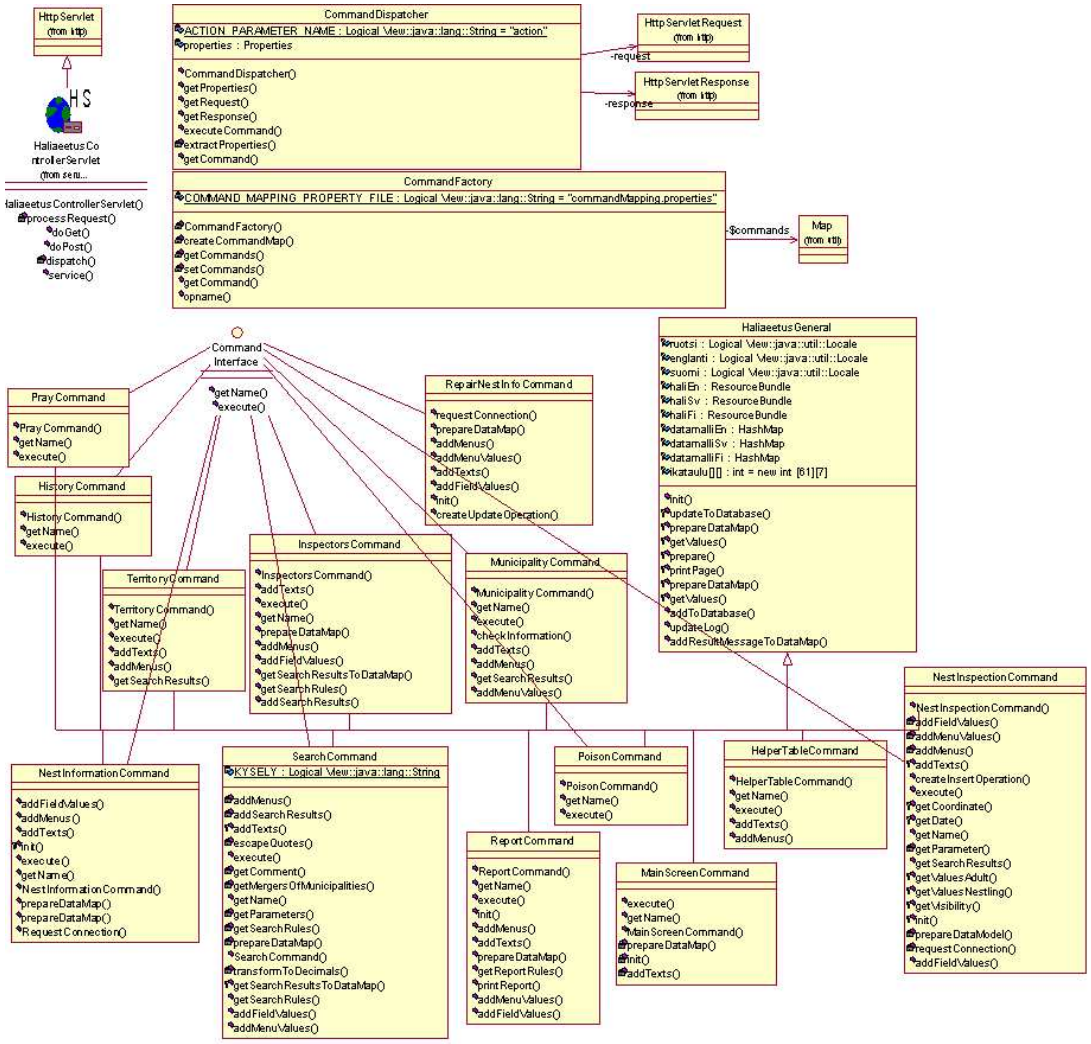
Tarkistaa, että desimaaliluku on halutun lainen. Eli kokonaisosa ja desimaaliosa on annettujen rajojen *integer_length* ja *decimal_length* sisällä. Ja merkkijonona annetun desimaaliluvun kaikki merkit ovat numeroita. Jos pisteen sijasta erotin merkkinä on annetu pilkku, niin muutetaan se pisteeksi. Jos desimaaliluku oli kelvollinen niin palautetaan se takaisin **String**-merkkijonona. Muutoin palautetaan arvo *null*.

4.6 Pakkaus haliaeetus.command

Sovelluskerroksen toiminnallisuus toteutetaan suunnittelumallin mukaisilla **Command**-luokilla.

Paketin **haliaeetus.command** luokat ja niiden suhteet ilmenevät luokkakaaviosta kuvassa 4.

Command-luokat kutsuvat pakkauksen **haliaeetus.db** tietokantaluokkia tietokantaa käsitellessään. Konfigurointitiedostoa lukiessaan ne kutsuvat pakkauksen **haliaeetus.helper** luokkaa **HaliProperties**. Lisäksi **Command**-luokat käyttävät ulkoisia kirjastoja esim. **template**-tiedostojen tulostamiseksi ja kielitukitiedostoja (Luku 4.6.2.1) kielituen toteuttamiseen. Lopuksi ne palauttaa selaimelle generoimansa WWW-sivun.



Kuva 4: Luokkakaavio haliaeetus.command-paketista

Komento-luokan suoritus käynnistyy, kun **CommandFactory**-luokka luo siitä ilmentymän. **Command**-luokka vastaanottaa informaatiota **CommandFactory**-luokalta. Luokka jakaa pyynnöt ja lomakkeelta lähetetyt tiedot oikeille **Command**-luokille. Vastaanotettuaan tiedot **Command**-luokka pyytää **ConnectionPool**-luokalta yhteyttä tietokantaan tietokantaoperaatiota varten.

Saatuaan yhteyden tietokantaan **Command**-luokka lähettää asianomaiselle tietokantaoperaatio-luokalle lomakkeelta saamansa tiedot tietokantaoperaatiota varten. Tietokantaoperaatio voi olla haku (**SearchOperation**), lisäys (**InsertOperation**), muokkaus (**UpdateOperation**) tai poisto (**DeleteOperation**). Command-luokka hakee **Table**-luokan avulla tiedostosta tabledata jokaisen tietokannan kohdetaulunsa kuvauksen eli metadatan (Luku 4.3.2).

Command-luokka saa vastauksena tietokantaoperaatio-luokalta joko **OperationResults**-olion, joka sisältää tiedon operaation onnistumisesta, tai hakuoperaation **SearchResults**-olion, joka sisältää tulosjoukon. Operaation onnistuessa ilman palautettavaa tulosjoukkoa luokka tulostaa näytölle viestin operaation onnistumisesta. Muutoin **Command**-luokka tulostaa mallipohjaa eli **templatea** apunaan käyttäen tulosjoukon tiedot näytölle.

Virhetapauksissa **Command**-luokka tuottaa näytölle virheilmoituksen. Virheilmoitukset saadaan kielitukitiedostoista (Luku 4.6.2.1). **Command**-luokat voivat myös suorittaa tarkistuksia lomakkeelta saamilleen tiedoille, mutta pääsääntöisesti ne käyttävät pakkauksen **haliaeetus.helper** luokkaa Tarkista tietokantaan lisättävien ja muutettavien tietojen oikeellisuuden tarkastamiseksi. Lisäksi luokat voivat tuottaa **template**-tiedostojen avulla uusia näyttöjä (lomakkeita) ottamatta tietokantayhteyttä.

4.6.1 HaliaeetusGeneral

HaliaeetusGeneral-luokka on kaikkien **Command**-luokkien yliluokka, jossa toteutetaan kaikille yhteiset muuttujat ja metodit. Yliluokassa **HaliaeetusGeneral** alustetaan kielituki (Luku 4.6.2.1) datamalliin (Luku 4.6.1), josta jokainen luokka ottaa kopion.

Muuttujat:

protected Template template

Luokan käyttämä mallipohja

protected HashMap datamalliFi

Luokan suomenkieliset staattiset tiedot sisältävä datamalli.

protected HashMap datamalliSv

Luokan ruotsinkieliset staattiset tiedot sisältävä datamalli.

protected HashMap datamalliEn

Luokan englanninkieliset staattiset tiedot sisältävä datamalli.

protected ResourceBundle haliFi

Luokan resurssikimppu, joka sisältää tuen suomen kielelle.

protected ResourceBundle haliSv

Luokan resurssikimppu, joka sisältää tuen ruotsin kielelle.

protected ResourceBundle haliEn

Luokan resurssikimppu, joka sisältää tuen englannin kielelle.

protected Locale suomi

Luokan lokaaliolio, suomi.

protected Locale ruotsi

Luokan lokaaliolio, ruotsi.

protected Locale englanti

Luokanlokaaliolio, englanti.

Julkiset metodit:

public void init()

Tämä metodi kutsutaan jokaisen luokan latauksen yhteydessä. Metodi sijoittaa ilmentymämuuttujille suomi, ruotsi ja englanti arvoksi kieltä vastaavan **Locale**-olion ja luo jokaiselle tuettavalle kielelle (suomi, ruotsi ja englanti) oman resurssikimpun (**ResourceBundle**-olion), jotka sijoitetaan ilmentymämuuttujiin *haliFi*, *haliSv* ja *haliEn*.

protected Connection getConnection(HttpServletRequest request)()

Hakee parametrista request, parametrin sessio avulla käyttäjän tunnuksen ja salasanan. Tunnukselle ja salasanalle haetaan tietokantayhteys. Metodi palauttaa Connection-olion.

protected boolean addToDatabase (Table taulu, Map arvot, Map datamalli, Connection con, ResourceBundle kieli)

Metodi lisää parametrin **Map**-olion arvot tiedot tietokantaan, parametrin taulu määrittelemään tietokantatauluun, luokan **InsertOperation** avulla. Metodi lisää **Map** -olioon datamalli pakkausnäkyvyyden metodit:

protected boolean updateToDatabase (Table taulu, Map arvot, Map datamalli, Connection con, int pesa_id, ResourceBundle kieli)

Metodi päivittää parametrin **Map**-olion arvot tiedot tietokantaan, parametrin taulu määrittelemään tietokantatauluun, luokan **UpdateOperation** avulla. Metodi lisää **Map** -olioon datamalli tiedon päivityksen onnistumisesta.

Mikäli päivitys epäonnistuu, metodi palauttaa *false*.

Map -olion arvot arvojen oikeellisuudet tulisti tarkistuttaa luokalla **haliaeetus.helper.CheckHelper**-luokan oikealla aliluokalla ennen tämän metodin kutsua.

protected void prepareDataModel(Map dataModel,String languageSelection)

Suorittaa alustustoimenpiteitä.

protected void getValues(HttpServletRaquest, Table table)

Hakee parametrista parametrin *table* avulla tarvittavat tiedot ja palauttaa ne **HashMap**-oliona. Jos tietoja ei löydy tai ne sisältävät pelkkiä tyhjiä merkkejä, niitä ei lisätä palautettavaan olioon. Jos mitään tietoja ei löydy palauttaa tyhjän **HashMap**-olion.

protected void printPage(Map dataModel, HttpServletResponse response)

Tulostaa näytön yhdistämällä *dataModel*-muuttujan sisaltamat tiedot Freemarkerin templateen.

protected void addMenus(Connection con, Map dataModel) throws Exception

Lisää kunta-valikon, jossa on vain ne kunnat, joihin on tehty pesätarkastus, datamalliin.

4.6.2 CommandDispatcher

Luokka hoitaa pyyntöjen delegoimisen varsinaisille Komento-luokille.

Konstruktori: **public CommandDispatcher(HttpServletRequest request, HttpServletResponse response)**

Tallettaa parametrit luokkamuuttujiinsa ja kutsuu metodia extractProperties.

Julkiset metodit:

public Properties getProperties()

Metodi palauttaa pyynnön parametrit **Properties**-muodossa.

public HttpServletRequest getRequest()

Metodi palauttaa pyynnön.

public HttpServletResponse getResponse()

Metodi palauttaa vastauksen.

public CommandInterface getCommand()

Lukee pyynnöstä *action* -parametrin ja pyytää **CommandFactory**-luokalta ilmentymän halutusta komentoluokasta.

public void executeCommand()

Kutsuu haluttua varsinaista **Command**-luokkaa.

Yksityiset metodit:

private Properties extractProperties(HttpServletRequest servletrequest)

Luo **Properties**-olion, jonne tallettaa pyynnössä saamansa parametrit.

4.6.3 CommandFactory

Singleton-suunnittelumallin mukainen luokka, joka hoitaa komentojen mappauksen **commandMapping.properties**-konfigurointitiedoston (ks. luku 4.9.2) avulla.

Konstruktori:

private CommandFactory()

Konstruktorin näkyvyydellä (private) estetään ilmentymien luonti.

Julkiset metodit:

public static CommandInterface getCommand(String action)

Palauttaa ilmentymän komentoluokasta, joka vastaa parametriä.

Yksityiset metodit:

private static void createCommandMap()

Metodi lukee **commandMapping.properties**-konfigurointitiedoston (ks. luku 4.9.2) ja muodostaa siitä **Map**-tietorakenteen, jonka tallettaa luokkamuuttujaansa.

private static Map getCommands()

Metodi palauttaa **Map**-tietorakenteen, jossa on **commandMapping.properties**-konfigurointitiedoston arvot.

private static void setCommands(Map commandMap)

Metodi asettaa parametrinaan saamaansa **Map**-tietorakenteen luokkamuuttujaansa arvot.

4.6.4 CommandInterface

Luokka on kaikkien varsinaisten komentoluokkien rajapinta-luokka.

Julkiset metodit:

public String getName()

public void execute(CommandDispatcher dispatcher)

4.6.5 LoginScreenCommand

LoginScreenCommand -luokan tehtävänä on luoda Haliaeetus-järjestelmän sisäänkirjautumissivu ja autentikoida käyttäjä. **LoginScreenCommand** perii luokan **HaliaeetusGeneral** ja toteuttaa luokan **CommandInterface**.

Luokka käyttää **login.ftl-templatea** sivun generoimisessa.

Konstruktori:

LoginScreenCommand()

Julkiset metodit:

public void execute(CommandDispatcher dispatcher)

Tätä metodia kutsutaan **CommandDispather**-luokasta. Metodi kutsuu yliluokan *init*-metodia, joka hoitaa tarvittavat alustukset ja sen jälkeen itse alustaa tarvittavan templaten (*hali.ftl*).

public String getName()

Palauttaa komentoluokan *action*-parametrin nimen.

protected void addTexts(Map dataModel, ResourceBundle language)

Lisää resurssikimpun *language* sisältämät staattiset tekstit ja virheilmoitukset annettuun **Map**-tietorakenteeseen.

protected void prepareDataModel(HttpServletRequest request, HttpServletResponse response)

Alustaa datamallin.

public void generateDisplay(CommandDispatcher dispatcher)

Alustetaan mallipohjaan liittyvät datamallit templaten tarvitsemilla staattisilla teksteillä. Datamallit vastaavat järjestelmän tukemia kieliä.

private void autenticate(CommandDispatcher dispatcher) throws Exception

Hoitaa käyttäjän autentikoimisen ja luo kayttäjälle istunnon (session)

4.6.6 MainScreenCommand

MainScreenCommand -luokan tehtävänä on luoda Haliaeetus-järjestelmän päänäyttö. Päänäytölle (Haku-näyttö) siirrytään suoraan sisäänkirjautumisen jälkeen ja näytön generoimiseksi luokkaa voidaan kutsua myös näyttöjen navigointi-palkin painikkeella ”Haku”. Päänäytöltä voidaan hakea pesien tarkastuksia erilaisilla hakuehdoilla.

Hakutulokset sisältävän näytön generoimiseksi **MainScreenCommand**-luokkaa kutsutaan näytön painikkeella ”Hae Pesät”. Luokka käyttää *mainscreen.ftl*-templatea sivun generoimisessa.

Hakutuloksesta voidaan siirtyä muokkaamaan pesän ja tarkastuksen tietoja ja tekemään uusia tarkastuksi haetuille pesille. Tällöin siirretään kontrolli luokalta eteenpäin ja kutsutaan luokkaa **NestInformationCommand**.

Konstruktori:

MainScreenCommand()

Julkiset metodit:

public void execute(CommandDispatcher dispatcher)

Tätä metodia kutsutaan CommandDispatheristä.

public String getName()

Palauttaa komentoluokan *action*-parametrin. nimen.

Pakkausnäkyvyyden metodit:

protected void addTextsFromResourceBundles()

Alustaa mallipohjaan liittyvät datamallit mainscreen-templaten tarvitsemilla staattisilla teksteillä. Datamallit vastaavat järjestelmän tukemia kieliä.

protected void addTexts(Map dataModel, ResourceBundle language)

Lisää resurssikimpun *language* sisältämät staattiset tekstit ja virheilmoitukset annettuun **Map**-tietorakenteeseen.

void prepareDataModel(Map dataModel, HttpServletRequest request)

Hakee *request*-olion parametrien arvot **Map**-olioihin, jotka lopuksi lisätään **Map**-olioon *dataModel*.

protected ResourceBundle getLanguage(HttpServletRequest request)

Palauttaa kielitukeen tarvittavan resurssikimpun.

protected String makeQuery(Map dataModel, Connection con, HttpServletRequest request)

Rakentaa kyselyn, jolla kannasta haetaan käyttäjän antamien hakuehtojen mukaiset arvot hakutulokseen. Käyttää apunaan privaattia apumetodia **getSearchConditions**.

protected void addSearchResults(Connection con, String kysely, Map dataModel, ResourceBundle language)

Laittaa annettuun **Map**-olioon *dataModel* merkin staattisten hakutuloksiin liittyvien tekstien näyttämisestä selaimella. Suorittaa **SQL**-kyselyn *kysely*

tietokantaan. Kyselyn onnistuessa lisää kyselyn tuloksen annettuun *dataModel*-olioon. Jos haun tulosjoukko on tyhjä, lisää *dataModel*-olioon tiedon, ettei hakuehdoilla löytynyt tietoja. Kyselyn epäonnistuessa lisää *dataModel*-olioon virheilmoituksen ja kyselyn hakuehtoina olleet tiedot.

Yksityiset metodit:

private boolean checkIfNewInspectionCanBeDone(String vuosi, String pesa_id, Connection con)

Tarkistaa voidaanko tehdä uusi tarkastus vuodelle *vuosi* pesälle, jonka pesan id on *pesa_id*. Jos voidaan niin palautetaan totuusarvo *true*.

private void escapeQuotes(String s)

Muuntaa merkkijonon s yksinkertaiset lainausmerkit kahdennettuun muotoon **SQL**-kyselyä varten. Mikäli syöte on esim. *j'aime Oracle*, palauttaa *j''aime Oracle*.

private String getMunicipalityJoins(String kunta)

Kuntaliitoksen mukaan haetaan kannasta (parametri *kunta*)ne pesa/reviiri -kunnat, jotka ovat joskus kuuluneet kysytyn kunnan alueelle.

private String getSearchConditions(Connection con, Map arvoSolmu)

Hankkii hakuoperaatiota varten käyttäjän syöttämät hakuehdot, jotka on talletettu **Map**-olioon *arvoSolmu*, ja palauttaa ne SQL-kyselyn **WHERE**-osaan sijoitettavana **String**-merkkijonona.

private void getParameters(String nimi, HttpServletRequest request)

Palauttaa pyynnön *request* parametrin *nimi* arvon **String**-oliona. Jos parametria ei ole olemassa tai se sisältää pelkkiä tyhjiä merkkejä, palauttaa tyhjän **String**-olion.

private String changeToLatLon(String dd, String mm, String ss)

Yhdistää astekoordinaattien aste (*dd*), minuutti (*mm*) ja sekuntiosan (*ss*) astekoordinaattien pituus- tai leveysosaksi.

4.6.7 NestInspectionCommand

Luokan **NestInspectionCommand** tehtävänä on lisätä tietokantaan tiedot pesästä, jota ei ole aikaisemmin tallennettu tietokantaan, sekä tiedot jo olemassa olevan pesän uudesta tarkastuksesta.

Luokka **extends HaliaeetusGeneral** ja **implements CommandInterface**

Luokkaa kutsutaan näytöltä UusiPesä ja näytöltä UusiTarkastus tietojen tallentamista varten.

Luokka **NestInspectionCommand** lisää pesän tiedot tietokantaan seuraavasti: Aluksi luokka tallettaa HTTPS-pyynnön parametreina välitetyt lisättävät arvot paikalliseen muuttujaan. Sitten se tarkastaa lisättävät arvot. Jos tietokantaan yritetään lisätä virheellisiä arvoja, luokka välittää *template*-tiedoston *Pesa.ftl* ja virheilmoitukset takaisin sille pyynnön välittäneelle **NestInformationCommand luokalle** ja lopettaa suorituksensa. Muuten se pyytää luokalta **ConnectionPool** tietokantayhteyttä. Tämän jälkeen luokka suorittaa jokaista luomaansa **Table**-ilmentymää kohti lisäysoperaation, kerättyään lisättävät arvot **HasMap**-olioon.

Jokainen erillinen lisäysoperaatio palauttaa tiedon tietokantaoperaation onnistumisesta. Tietojen kirjaamispäivä ja muuttamispäivä päivitetään automaattisesti kyseessä olevaksi päiväksi. Kun luokka on suorittanut kaikki erilliset lisäysoperaatiot, se vapauttaa tietokantayhteyden. Mikäli tietojen lisääminen tietokantaan epäonnistui, luokka kutsuu **NestInformationCommand** luokkaa ja välittää tälle virheilmoitukset. Vastaan ottava luokka lisää virheilmoitukset tai onnistumisilmoitukset templetelle ja tuottaa uudestaan sivun jolla on käyttäjän lisäämät arvot ja annetut virheilmoitukset.

Luokka kerää lomakkeelle syötetyt tiedot (jotka se saa käyttöönsä CommandDispatherin välittämistä request tai properties tiedostoista) **HaShMap**-olioihin ja lähettää ne tarkastettavaksi **CheckNest()**-metodille. Jos arvoissa esiintyy virheitä, tuottetaan Uusi pesä –näyttö (kuten yllä selitettiin kutsumalla **NestInformationCommand** luokkaa), jolle tulevat käyttäjän jo syöttämät tiedot sekä virheilmoitus.
Mikäli lisättävät tiedot ovat oikein, luokka lisää tiedot tietokantaan ja kutsuu **NestInfor-**

mationCommand-luokkaa.

Konstruktori:

public void NestInspectionCommand()

Julkiset metodit:

public void execute(CommandDispatcher dispatcher)

Tätä metodia kutsutaan CommandDispatheristä. Se sisältää luokan kaiken toiminnallisuuden, joka on jaettu privaatteihin metodeihin. Luokka suorittaa hieman eri operaatioita riippuen siitä saako se **ActionParametrina** arvon **NewNestCheck** (käyttötapaus uusi tarkastus) vai **NewInspection** (uusi vuositarkastus).

public String getName()

Palauttaa komentoluokan nimen

Yksityiset metodit:

private void getTableData()

Metodi hakee taulujen kuvaukset **TableData** tiedastosta ja luo **Table**-luokan ilmentymät.

private void getPropertyValues()

Tämä metodi hakee arvot kutsussa saamastaan **Properties**-tiedostosta ja vie ne oikeisiin Table-muuttujiin. Metodi käyttää luokan **HaliaeetusGeneral** metodia **getValues(Request request, Table table)**, joka lisää taulujen kuvauksia vastaaviin **HashMap**-olioihin kutsussa **templatelta** välitetyistä arvoista ne jotka ovat samannimisiä kuin taulun muuttujat. Sen lisäksi metodi lisää luokkien kuvauksiin muita arvoja, joita ei saada **getValues**-avulla suoraan tai joita pitää käsitellä.

private void addValuesToTables()

Metodi lisää kannasta noudettavat arvot **HashMap**eihin, sekä lisää vielä joitakin vakiotietoja tietokantaan talletusta varten.

private String getProperties(String nimi)

Palauttaa kutsussa annetun **Properties**-tiedoston parametrin **nimi** arvon String-oliona. Jos parametria ei ole olemassa tai se sisältää pelkkiä tyhjiä merkkejä, palauttaa tyhjä String-olion.

private String getKordinate(String tyyppi)

Hakee kutsussa annetusta **Properties**-tiedostosta koordinaatteihin tarvittavat tiedot ja palauttaa ne merkkijonona. Jos tietoja ei löydy tai ne sisältävät pelkkiä tyhjiä merkkejä, palauttaa tyhjän String-olion.

private HashMap getNestlingValues(Properties properties, Table taulu, int i)

Hakee kutsussa annetusta **Properties**-tiedostosta parametrien **taulu** ja **i** avulla tietokantatauluun poikanen tarvittavat tiedot ja palauttaa ne **HashMap**-oliona. Jos tietoja ei löydy tai ne sisältävät pelkkiä tyhjiä merkkejä, niitä ei lisätä palautettavaan olioon. Jos mitään tietoja ei löydy palauttaa tyhjän **HashMap**-olion.

private void generateDisplay(Map datamap

Tällä metodilla ei tässä luokassa tehdä mitään, mutta implementointi vaatii metodin toteuttamista.

4.6.8 RepearNestInfoCommand

RepairNestInformationCommand tehtävänä on korjata tietokantaan lisättyjen tarkastusten tietoja.

Haku-näytöillä on Korjaa Pesän tietoja -linkki, joka kutsuu luokkaa **RepairNestInformationCommand**. Näiltä näytöiltä kutsuttaessa luokka **RepairNestInformationCommand** ottaa vastaan **Dispatherin**, tutkii mitkä tiedot ovat muuttuneet ja vie korjatut tiedot kantaan.

RepairNestInformationCommand korjaa tarkastuksen tiedot tietokantaan seuraavasti: Aluksi luokka tarkastaa lisättävät arvot. Jos tietokantaan yritetään lisätä virheellisiä arvoja, luokka tuottaa *template*-tiedoston *Pesa.ftl* virheilmoitusten kera kutsumalla **NestInformationCommand**-luokkaa, joka tuottaa teplaten mukaisen sivun. Tämän jälkeen se lopettaa suorituksensa.

RepearNestInfoCommandpyytää luokalta **ConnectionPool** tietokantayhteyttä. Tämän jälkeen luokka tarkastaa taulu kerrallaa onko jokin arvo muuttunut. Muuttuneet arvot päivitettään tauluihin yksi kerrallaan. Aina sitä mukaa kun jonkin arvon todetaan muuttuneen.

Jokainen erillinen päivitys-operaatio palauttaa tiedon tietokantaoperaation onnistumisesta. Tietojen muuttamispäiväksi päivitetään automaattisesti kyseessä olevaksi päiväksi. Kun luokka on suorittanut kaikki erilliset päivitysoperaatiot, se vapauttaa tietokantayhteyden. Mikäli tietojen päivittäminen tietokantaan epäonnistui, luokka tuottaa *template*-tiedoston *Pesa.ftl* virheilmoitusten kanssa kutsumalla **NestInformationCommand**-luokkaa.

Luokka välittää lomakkeelta saadut arvot **CheckNestInspectionHelper()**-metodilleen. Jos arvoissa esiintyy virheitä, tuotetaan uudestaan korjaus –näyttö, jolla on käyttäjän jo syöttämät tiedot sekä virheilmoitus.

Mikäli lisättävät tiedot ovat oikein, luokka päivittää tiedot tietokantaan ja kutsuu **NestInformationCommand**-luokkaa, joka tuottaa Pesätiedot-näytön juuri lisätyillä arvoilla.

Konstruktori:

public void RepearNestInfoCommand()

Julkiset metodit:

public void execute(CommandDispatcher dispatcher)

> Tätä metodia kutsutaan CommandDispatheristä. Metodi sisältää kaiken luokan toiminnallisuuden (toimii runkona). Yksittäiset toiminnot on pilkottu private-metodeiksi, joita tästä metodista kutsutaan.

public String getName()

> Palauttaa komentoluokan nimen

.

Yksityiset metodit:

private void getTableData()

Metodi hakee taulujen kuvaukset **TableData** tiedastosta ja luo **Table**-luokan ilmentymät.

private boolean checkTable(Table taulu, String taulu_nimi, String suoritettava_kysely)

Tämä metodi tarkistaa ovatko parametrina annettavan taulun tiedot muuttuneet. Kyseisen taulun tiedot haetaan kannasta ja niitä verrataan kutsussa saatuihin arvoihin. Jos tiedot ovat muuttuneet päivitetään vanhan taulun arvon päälle uusi arvo.Jos muutoksia ei ole tapahtunut siirrytäään seuraavan arvon tarkastamiseen.

private void checkYearAndTerritory()

Tarkastetaan erikseen onko kunnassa tai reviirissä tapahtunut muutoksia, koska nämä arvot ovat kannassa tallessa ID-numeroilla ja lomakkeelta tulevat arvot ovat tunnuksellisessa String muodossa.

private void generateDisplay(Map datamap)

Tällä metodilla ei tässä luokassa tehdä mitään, mutta implementointi vaatii metodin toteuttamista.

4.6.9 NestInformationCommand

NestInformation tehtävänä on tuottaa Pesätiedot-näyttö.

Näytöiltä Haku, Myrkyt, Saaliit, Historia, Pesätiedot, Uusi pesä ja Pesä Tarkastus kutsuttaessa tai luokkaa NestInformationCommand kutsumalla, luokka tuottaa Pesätiedot-näytön. Näyttö tuotetaan template-tiedoston Pesa.ftl avulla. Luokka sijoittaa näytölle valitun pesän valitun vuoden tarkastustiedot. Mikäli luokan kutsussa ei ole määritelty mitään vuotta, sijoitetaan näytölle uusimman tarkastuksen tiedot. Luokka hakee näytölle sijoitettavat tiedot tietokannasta HTTPS-pyynnön parametreina saamiensa pesä- ja vuositiedon

perusteella. Jos mitään tietoja ei välitetty Luokka hakee näytölle ainoastaan seuraavan vapaan pesa_id:n ja muuten tuottaa tyhjän lomakkeen.

Kostruktori:

public void NestInformationCommand(Template temp, int pesa_id)

Julkiset metodit:

public void execute(CommandDispatcher dispatcher)

Tätä metodia kutsutaan CommandDispatheristä.

public String getName()

Palauttaa komentoluokan *action*-parametrin nimen

.

Yksityiset metodit:

private void addMenus(Connection con, Map datamalli)

Metodi tekee jokaista alasvetovalikkoa kohden **SQL**-kyselyn tietokantaan. Kyselyn onnistuessa metodi lisää alasvetovalikoiden sisällön parametrina annettuun **Map**-olioon datamalli. Kyselyn epäonnistuessa metodi lisää virheilmoituksen parametrina annettuun **Map**-olioon datamalli.

private void addFieldValues (Connection con, Map datamalli)

Metodi laittaa parametrina annettuun **Map**-olioon datamalli merkin staattisten hakutuloksiin liittyvien tekstien näyttämisestä selaimella. Metodi muodostaa tarvittavan **SQL**-kyselyn tietokantaan luokan **SearchOperation** avulla. Kyselyn onnistuessa metodi lisää kyselyn tuloksen parametrina annettuun

Map-olioon datamalli. Jos haun tulosjoukko on tyhjä, metodi lisää **Map**-olioon datamalli tiedon, ettei hakuehdoilla löytynyt tietoja. Kyselyn epäonnistuessa metodi lisää **Map**-olioon datamalli virheilmoituksen ja kyselyn hakuehtoina olleet tiedot.

private void addTexts (Map datamalli, ResourceBundle kieli)
Lisää tekstit template tiedostoon.

Private void prepareDataMap(Logical View::java::lang::String String, Map datamap) Suoraittaa alustus toimenpiteitä.

Private void RequestConnection (Logical View::java::lang::String String, Logical View::java::lang::String String) Pyytää yhteyttä ConnectionPool-Luokalta.

4.6.10 PoisonCommand

Myrkky-luokan tehtävänä on tarkistaa käyttäjän syöttämät myrkkytiedot analysoiduista näytteistä ja viedä ne tietokantaan.

Luokkaä kutsutaan Pesätiedot-näytöltä. Lisäksi Saaliit- ja Historia-näytöiltä on myös mahdollisuus kutsua Myrkky-luokkaä.

Luokkakäyttää Myrkyt-templatea Myrkyt-näytön (Kuva s. 71) generoimisessa.

Konstruktori:

public void PoisonCommand()

Julkiset metodit:

public void execute(CommandDispatcher dispatcher)

Tätä metodia kutsutaan CommandDispatheristä.

public String getName()

Palauttaa komentoluokan *action*-parametrin nimen

.

Yksityiset metodit:

private void addMenus(Connection con, Map datamalli)

Metodi tekee jokaista alasvetovalikkoa kohden **SQL**-kyselyn tietokantaan. Kyselyn onnistuessa metodi lisää alasvetovalikoiden sisällön parametrina annettuun **Map**-olioon datamalli. Kyselyn epäonnistuessa metodi lisää virheilmoituksen parametrina annettuun **Map**-olioon datamalli.

private void addFieldValues (Connection con, Map datamalli)

Metodi laittaa parametrina annettuun **Map**-olioon datamalli merkin staattisten hakutuloksiin liittyvien tekstien näyttämisestä selaimella. Metodi muodostaa tarvittavan **SQL**-kyselyn tietokantaan luokan **SearchOperation** avulla. Kyselyn onnistuessa metodi lisää kyselyn tuloksen parametrina annettuun **Map**-olioon datamalli. Jos haun tulosjoukko on tyhjä, metodi lisää **Map**-olioon datamalli tiedon, ettei hakuehdoilla löytynyt tietoja. Kyselyn epäonnistuessa metodi lisää **Map**-olioon datamalli virheilmoituksen ja kyselyn hakuehtoina olleet tiedot.

private void addTexts (Map datamalli, ResourceBundle kieli)
Lisää tekstit template tiedostoon.

Private void prepareDataMap(Logical View::java::lang::String String, Map datamap) Suoraittaa alustus toimenpiteitä.

Private void RequestConnection (Logical View::java::lang::String String, Logical View::java::lang::String String) Pyytää yhteyttä ConnectionPool-Luokalta.

4.6.11 PrayCommand

PrayCommand-luokan tehtävänä on tarkistaa käyttäjän syöttämät tiedot kerätyistä saalisnäytteistä ja viedä ne tietokantaan. Luokkaä kutsutaan Pesätiedot-näytöltä. Lisäksi Myrkyt- ja Historia-näytöiltä on myös mahdollisuus kutsua Saaliit-luokkaä. Luokka käyttää Saaliit-templatea Saaliit-näytön (Kuvat s. 72 ja 72) generoimisessa.

Luokka tekee ensin tietokantahaun, jolla selvitetään, onko tietokannassa valittuun pesään ja vuoteen liittyviä saalistietoja. Jos on, ne tulostetaan näytölle. Jos kyseiselle vuodelle ei ole kirjattu saalistietoja, tulostetaan perusnäyttö.
Jos käyttäjä lisää tai muuttaa tietoja, tätä metodia kutsutaan uudestaan. Metodi kerää lomakkeelle syötetyt tiedot Map-olioon ja lähettää sen tarkastettavaksi luokalle CheckPrayHelper(). Jos arvoissa esiintyy virheitä, generoidaan Saaliit-näyttö, jossa on käyttäjän jo syöttämät tiedot sekä virheilmoitus.
Mikäli lisättävät/päivitettävät tiedot ovat oikein, luokka lisää/päivittää tiedot kantaan addToDatabase()- tai UpdateToDatabase()-metodilla ja generoi Saaliit-näytön, jossa on käyttäjän syöttämät uudet tiedot sekä mahdollisesti jo olleet vanhat tiedot. Lisäksi ilmoitetaan lisäys/ päivitysoperaation onnistumisesta.

Konstruktori:

public void PrayCommand()

Julkiset metodit:

public void execute(CommandDispatcher dispatcher)

Tätä metodia kutsutaan CommandDispatheristä.

public String getName()

Palauttaa komentoluokan *action*-parametrin nimen

.

Yksityiset metodit:

private void addMenus(Connection con, Map datamalli)

Metodi tekee jokaista alasvetovalikkoa kohden **SQL**-kyselyn tietokantaan. Kyselyn onnistuessa metodi lisää alasvetovalikoiden sisällön parametrina annettuun **Map**-olioon datamalli. Kyselyn epäonnistuessa metodi lisää virheilmoituksen parametrina annettuun **Map**-olioon datamalli.

private void addFieldValues (Connection con, Map datamalli)

Metodi laittaa parametrina annettuun **Map**-olioon datamalli merkin staattisten hakutuloksiin liittyvien tekstien näyttämisestä selaimella. Metodi muodostaa tarvittavan **SQL**-kyselyn tietokantaan luokan **SearchOperation** avulla. Kyselyn onnistuessa metodi lisää kyselyn tuloksen parametrina annettuun **Map**-olioon datamalli. Jos haun tulosjoukko on tyhjä, metodi lisää **Map**-olioon datamalli tiedon, ettei hakuehdoilla löytynyt tietoja. Kyselyn epäonnistuessa metodi lisää **Map**-olioon datamalli virheilmoituksen ja kyselyn hakuehtoina olleet tiedot.

private void addTexts (Map datamalli, ResourceBundle kieli)
Lisää tekstit template tiedostoon.

Private void prepareDataMap(Logical View::java::lang::String String, Map datamap) Suoraittaa alustus toimenpiteitä.

Private void RequestConnection (Logical View::java::lang::String String, Logical View::java::lang::String String) Pyytää yhteyttä ConnectionPool-Luokalta.

4.6.12 HistoryCommand

HistoryCommand-luokka tallettaa tarkistusten jälkeen tietokantaan lähinnä tietoja pesistä ennen varsinaisen seurannan alkamista vuonna 1972. Uudempaakin historiatietoa on myös mahdollista lisätä. Näytöllä voidaan lisäksi päivittää tietyn pesän historiatietoja. Luokkaa kutsutaan Pesätiedot-sivulta. Lisäksi Myrkyt- ja Saaliit –näytöiltä on myös mahdollisuus kutsua Historia-luokkaä. Luokka käyttää Historia-templatea Historia-näytön (Kuva s. 69) generoimisessa.

Luokka tekee ensiksi tietokantahaun, jolla selvitetään, onko tietokannassa valittuun pesään liittyviä historiatietoja. Jos on, ne tulostetaan näytölle. Jos kyseiselle vuodelle ei ole kirjattu historiatietoja, tulostetaan perusnäyttö.
Jos käyttäjä lisää tai muuttaa tietoja, tätä metodia kutsutaan uudestaan.
Metodi kerää lomakkeelle syötetyt tiedot Map-olioon ja lähettää sen tarkastettavaksi luokalle CheckHistoryHelper. Jos arvoissa esiintyy virheitä, generoidaan Historia-näyttö, jossa on käyttäjän jo syöttämät tiedot sekä virheilmoitus.
Mikäli lisättävät tiedot ovat oikein, luokka lisää tiedot kantaan addToDatabase()- tai updateToDatabase()-

metodilla ja generoi Historia-näytön, jossa on käyttäjän syöttämät uudet tiedot sekä mahdollisesti jo olleet vanhat tiedot. Lisäksi ilmoitetaan lisäys/ päivitysoperaation onnistumisesta.

Konstruktori:

public void HistoryCommand()

Julkiset metodit:

public void execute(CommandDispatcher dispatcher)

Tätä metodia kutsutaan CommandDispatheristä.

public String getName()

Palauttaa komentoluokan *action*-parametrin nimen

.

Yksityiset metodit:

private void addMenus(Connection con, Map datamalli)

Metodi tekee jokaista alasvetovalikkoa kohden **SQL**-kyselyn tietokantaan. Kyselyn onnistuessa metodi lisää alasvetovalikoiden sisällön parametrina annettuun **Map**-olioon datamalli. Kyselyn epäonnistuessa metodi lisää virheilmoituksen parametrina annettuun **Map**-olioon datamalli.

private void addFieldValues (Connection con, Map datamalli)

Metodi laittaa parametrina annettuun **Map**-olioon datamalli merkin staattisten hakutuloksiin liittyvien tekstien näyttämisestä selaimella. Metodi muodostaa tarvittavan **SQL**-kyselyn tietokantaan luokan **SearchOperation** avulla. Kyselyn onnistuessa metodi lisää kyselyn tuloksen parametrina annettuun

Map-olioon datamalli. Jos haun tulosjoukko on tyhjä, metodi lisää **Map**-olioon datamalli tiedon, ettei hakuehdoilla löytynyt tietoja. Kyselyn epäonnistuessa metodi lisää **Map**-olioon datamalli virheilmoituksen ja kyselyn hakuehtoina olleet tiedot.

private void addTexts (Map datamalli, ResourceBundle kieli)
Lisää tekstit template tiedostoon.

Private void prepareDataMap(Logical View::java::lang::String String, Map datamap) Suoraittaa alustus toimenpiteitä.

Private void RequestConnection (Logical View::java::lang::String String, Logical View::java::lang::String String) Pyytää yhteyttä ConnectionPool-Luokalta.

4.6.13 TerritoryCommand

TerritoryCommand-luokka generoi Reviirit –näytön (Kuva s. 73) ja suorittaa tietojen lisäämisen ja päivittämisen tietokantatauluun REVIIRI. Luokkaa Reviiri kutsutaan näyttöjen navigointi-palkin painikkeella Reviirit sekä Reviirit–näytön painikkeilla Lisää. Luokkakäyttää Reviirit-templatea Reviirit-näytön generoimisessa.

Luokka kerää lomakkeelle syötetyt tiedot Map-olioon ja selvittää, onko kyseessä pelkästään näytön Reviirit generoiminen vai liittyykö kyselyyn lisäksi lisäysoperaatio. Mikäli metodi pelkästään generoi näytön Reviirit, metodi kutsuu metodiaan pritpage().
Mikäli metodi generoi näytön Reviirit ja tekee lisäyksen tietokantaan, metodi kutsuu metodejaan addToDatabase() ja tulostaSivu(). Ennen metodin addToDatabase() kutsumista luokkapyytää tietokantayhteyden, joka vapautetaan lopussa.

Konstruktori:

public void TerritoryCommand()

Julkiset metodit:

public void execute(CommandDispatcher dispatcher)

Tätä metodia kutsutaan CommandDispatheristä.

public String getName()

Palauttaa komentoluokan *action*-parametrin nimen

.

Yksityiset metodit:

private void addMenus(Connection con, Map datamalli)

Metodi tekee jokaista alasvetovalikkoa kohden **SQL**-kyselyn tietokantaan. Kyselyn onnistuessa metodi lisää alasvetovalikoiden sisällön parametrina annettuun **Map**-olioon datamalli. Kyselyn epäonnistuessa metodi lisää virheilmoituksen parametrina annettuun **Map**-olioon datamalli.

private void addFieldValues (Connection con, Map datamalli)

Metodi laittaa parametrina annettuun **Map**-olioon datamalli merkin staattisten hakutuloksiin liittyvien tekstien näyttämisestä selaimella. Metodi muodostaa tarvittavan **SQL**-kyselyn tietokantaan luokan **SearchOperation** avulla. Kyselyn onnistuessa metodi lisää kyselyn tuloksen parametrina annettuun **Map**-olioon datamalli. Jos haun tulosjoukko on tyhjä, metodi lisää **Map**-olioon datamalli tiedon, ettei hakuehdoilla löytynyt tietoja. Kyselyn epäonnistuessa metodi lisää **Map**-olioon datamalli virheilmoituksen ja kyselyn hakuehtoina olleet tiedot.

private void addTexts (Map datamalli, ResourceBundle kieli)
Lisää tekstit template tiedostoon.

Private void prepareDataMap(Logical View::java::lang::String String, Map datamap) Suoraittaa alustus toimenpiteitä.

Private void RequestConnection (Logical View::java::lang::String String, Logical View::java::lang::String String) Pyytää yhteyttä ConnectionPool-Luokalta.

4.6.14 MunicipatyCommand

Luokan MunicipatyCommand avulla voidaan ylläpitää tietokannan **KUNTA**-taulua. **KUNTA**-tauluun voidaan lisätä uusia kuntia, kuntien tietoja voidaan muuttaa ja kunnille voidaan tehdä kuntaliitoksia.

Luokka MunicipalityCommand tuottaa Kunnat-näytön template-tiedoston *municipality.ftl* avulla. Luokka lisää uuden kunnan tietokantaan seuraavasti: Aluksi luokka tallettaa parametreina välitetyt lisättävät arvot **Map**-olioon. Sitten se tarkastaa lisättävät arvot. Jos tietokantaan yritetään lisätä virheellisiä arvoja, luokkatuottaa template-tiedoston *municipality.ftl* avulla näytön virheilmoitusten kanssa ja lopettaa suorituksensa. Muutoin luokka suorittaa lisäysoperaation.

Lisäysoperaatio palauttaa tiedon tietokantaoperaation onnistumisesta. Tietojen kirjaamispäivä päivitetään automaattisesti kyseessä olevaksi päiväksi. Kun luokka on suorittanut lisäysoperaation, se vapauttaa tietokantayhteyden ja tuottaa template-tiedoston *municipality.ftl* avulla Kunta-näytön, jolla on tieto operaation onnistumisesta.

Kuntien tietoja voidaan hakea kuntatunnuksen, suuralueen/eiden, ympäristökeskuksen tai kunnan sijaintitietojen perusteella. Kun painetaan ”Hae” -painiketta luokka suorittaa hakuoperaation seuraavasti: Aluksi luokka tallettaa parametreina välitetyt hakuehdot arvot **Map**-olioon ja pyytää tietokantayhteyden. Luokka koostaa hakuehtojen mukaisen kyselyn. Kun luokka on suorittanut hakuoperaation, se vapauttaa tietokantayhteyden ja tuottaa template-tiedoston *mainscreen.ftl* ja tulosjoukon avulla Kunta-näytön, jolla on hakua vastaavat kunnat.

Yksittäisen kunnan tietoja voidaan muuttaa editoimalla hakutuloksen rivin kenttiä, jossa on muutettavan kunnan tiedot. Kun painetaan editoidun rivin ”Muuta”-painiketta **MunicipalityCommand** luokka suorittaa päivitysoperaation Kunta-tauluun seuraavasti: Aluksi luokka tallettaa parametreina välitetyt päivitettävät arvot **Map**-olioon ja tarkastaa niiden oikeellisuuden ja pyytää tietokantayhteyden. Kun luokka on suorittanut päivitysoperaation, se vapauttaa tietokantayhteyden ja tuottaa template-tiedoston *mainscreen.ftl* avulla Kunta-näytön.

Kunnille voidaan tehdä myös kuntaliitos. Liitokseen haluttavat kunnat valitaan *checkbox*-valintakenttien avulla ja painetaan ”Kuntaliitos”-painiketta. **MunicipalityCommand** luokkaa kutsutaan ja se suorittaa tämän kuntaliitosoperaation seuraavasti: Aluksi luokka tallettaa parametreina välitetyt lisättävät arvot **Map**-olioon. Sitten se tarkastaa lisättävät/päivitettävät arvot ja pyytää tietokantayhteyden. Kun kuntaliitos on tehty, luokka vapauttaa tietokantayhteyn ja tuottaa template-tiedoston *mainscreen.ftl* avulla Kunta-näytön.

Luokka tuottaa Kunta-näytön myös, kun kutsu tapahtuu miltä tahansa näytöltä Kunnat-linkin kautta.

Konstruktori:

public void MunicipatyCommand()

Julkiset metodit:

public void execute(CommandDispatcher dispatcher)

Tätä metodia kutsutaan CommandDispatheristä.

public String getName()

Palauttaa komentoluokan *action*-parametrin nimen.

Pakkausnäkyvyyden metodit:

protected void addTextsFromResourceBundles()

Alustaa mallipohjaan liittyvät datamallit municipality-templaten tarvitsemilla staattisilla teksteillä. Datamallit vastaavat järjestelmän tukemia kieliä.

protected void addTexts(Map dataModel, ResourceBundle language)

Lisää resurssikimpun *language* sisältämät staattiset tekstit ja virheilmoitukset annettuun **Map**-tietorakenteeseen.

void prepareDataModel(Map dataModel, HttpServletRequest request)

Hakee *request*-olion parametrien arvot **Map**-olioihin, jotka lopuksi lisätään **Map**-olioon *dataModel*.

protected ResourceBundle getLanguage(HttpServletRequest request)

Palauttaa kielitukeen tarvittavan resurssikimpun.

protected void addMenus(Connection con, Map dataModel, ResourceBundle langu-ge)

Lisää **Map**-olioon *dataModel* uuden **Map**-olion *valikko* , johon kerätään html-sivulle tarvittavien valikkojen arvojoukot.

protected String makeQuery(Map dataModel, Connection con, HttpServletRequest request)

Rakentaa kyselyn, jolla kannasta haetaan käyttäjän antamien hakuehtojen mukaiset arvot hakutulokseen. Käyttää apunaan privaattia apumetodia **get-SearchConditions**.

Yksityiset metodit:

private boolean upadateMunicipality(Map dataModel, Connection con, Resource-Bundle language)

Muuttaa kunnan tietoja tietokannassa päivittämällä **KUNTA**-taulun rivejä. Jos Vantaa on liittynyt kunnan a kanssa, niin että kuntaliitoksen nimeksi on valittu Vantaa, niin kannassa ovat rivit:

KUNTA_ID	KUNTA_TUNNUS	KUNTA_LIITOS
1	Vantaa	Vantaa
2	a	a
3	a	Vantaa

Jos nyt Vantaan tietoja haluttaisiin päivittää, niin päivitykset on tehtävä riveille 1 ja 3. Metodi palauttaa totuusarvon *true*, jos päivitys onnistui.

private boolean municipalityJoin(Map dataModel, Table kunta, Connection con, ResourceBundle language)

Tekee kuntaliitoksen, jossa **KUNTA**-taulun rivejä päivitetään ja uusia rivejä lisätään tarpeen mukaan.

Esimerkiksi jos kunta vantaa on tehnyt aiemmin liitoksen kunnan a kanssa ja Espoo on tehnyt liitoksen kunnan b kanssa, niin kannassa on rivit:

KUNTA_ID	**KUNTA_TUNNUS**	**KUNTA_LIITOS**
1	Vantaa	Vantaa
2	a	a
3	a	Vantaa
4	Espoo	Espoo
5	b	b
6	b	Espoo

Jos nyt Vantaa ja Espoo päättäisivät tehdä kuntaliitoksen, jonka nimeksi tulisi Vantaa, niin kantaan lisättäisiin rivit :

7	Espoo	Vantaa
8	b	Vantaa

Ja rivien 1 ja 3 koordinaattitietoja joudutaan päivittämään. Metodi palauttaa totuusarvon *true*, jos kuntaliitos onnistui.

private String getParameter(String name, HttpServletRequest request)

Palauttaa pyynnön *request* parametrin *name* arvon **String**-oliona. Jos parametria ei ole olemassa tai se sisältää pelkkiä tyhjiä merkkejä, palauttaa tyhjän **String**-olion.

private String getSearchConditions(Connection con, Map hakuSolmu)

Hankkii hakuoperaatiota varten käyttäjän syöttämät hakuehdot, jotka on talletettu **Map**-olioon *hakuSolmu*, ja palauttaa ne SQL-kyselyn **WHERE**-osaan sijoitettavana **String**-merkkijonona.

private void addSearchResults(Connection con, String kysely, Map dataModel, ResourceBundle language, HttpServletRequest request)

Laittaa annettuun **Map**-olioon *dataModel* merkin staattisten hakutuloksiin liittyvien tekstien näyttämisestä selaimella. Suorittaa **SQL**-kyselyn *kysely* tietokantaan. Kyselyn onnistuessa lisää kyselyn tuloksen annettuun *dataModel*-olioon. Jos haun tulosjoukko on tyhjä, lisää *dataModel*-olioon tiedon,

ettei hakuehdoilla löytynyt tietoja. Kyselyn epäonnistuessa lisää *dataModel*-olioon virheilmoituksen ja kyselyn hakuehtoina olleet tiedot.

private void escapeQuotes(String s)

Muuntaa merkkijonon s yksinkertaiset lainausmerkit kahdennettuun muotoon **SQL**-kyselyä varten. Mikäli syöte on esim. *j'aime Oracle*, palauttaa *j''aime Oracle*.

4.6.15 InspectorCommand

InspectorCommand-luokan tehtävänä on tarkastaa käyttäjän syöttämät tiedot merikotkien rengastajista ja pesimätietojen kerääjistä ja tallentaa ne tietokantaan. Luokkaa kutsutaan Tarkastajat-näytöltä (Kuva s.73) ja näyttöjen yläosan navigointipalkista painikkeella Tarkastajat. Luokka käyttää Tarkastajat-templatea Tarkastajat-näytön generoimisessa.

Luokka kerää lomakkeelle syötetyt tiedot Map-olioon ja lähettää sen tarkastettavaksi luokan Tarkista metodille tarkistaTarkastajat(). Jos arvoissa esiintyy virheitä, generoidaan Tarkastajat-näyttö, jossa on käyttäjän jo syöttämät tiedot sekä virheilmoitus.
Mikäli lisättävät tiedot ovat oikein, luokka lisää tiedot kantaan käyttämällä addToDatabase()-metodia ja generoi Tarkastajat-näytön, jossa on käyttäjän syöttämät tiedot ja ilmoitus lisäysoperaation onnistumisesta.

Konstruktori:

public void InspectorCommand()

Julkiset metodit:

public void execute(CommandDispatcher dispatcher)

Tätä metodia kutsutaan CommandDispatheristä.

public String getName()

Palauttaa komentoluokan *action*-parametrin nimen

.

Yksityiset metodit:

private void addMenus(Connection con, Map datamalli)

Metodi tekee jokaista alasvetovalikkoa kohden **SQL**-kyselyn tietokantaan. Kyselyn onnistuessa metodi lisää alasvetovalikoiden sisällön parametrina annettuun **Map**-olioon datamalli. Kyselyn epäonnistuessa metodi lisää virheilmoituksen parametrina annettuun **Map**-olioon datamalli.

private void addFieldValues (Connection con, Map datamalli)

Metodi laittaa parametrina annettuun **Map**-olioon datamalli merkin staattisten hakutuloksiin liittyvien tekstien näyttämisestä selaimella. Metodi muodostaa tarvittavan **SQL**-kyselyn tietokantaan luokan **SearchOperation** avulla. Kyselyn onnistuessa metodi lisää kyselyn tuloksen parametrina annettuun **Map**-olioon datamalli. Jos haun tulosjoukko on tyhjä, metodi lisää **Map**-olioon datamalli tiedon, ettei hakuehdoilla löytynyt tietoja. Kyselyn epäonnistuessa metodi lisää **Map**-olioon datamalli virheilmoituksen ja kyselyn hakuehtoina olleet tiedot.

private void addTexts (Map datamalli, ResourceBundle kieli)
Lisää tekstit template tiedostoon.

Private void prepareDataMap(Logical View::java::lang::String String, Map datamap) Suoraittaa alustus toimenpiteitä.

Private void RequestConnection (Logical View::java::lang::String String, Logical View::java::lang::String String) Pyytää yhteyttä ConnectionPool-Luokalta.

4.6.16 HelperTableCommand

HelperTableCommand-luokka generoi Aputaulujen ylläpito -näytön ja suorittaa koodien ja selityksien lisäämisen ja päivittämisen tietokantatauluun **APUTAULU**. Luokka **HelperTable** kutsutaan näyttöjen yläosan navigointi-palkin painikkeella ”Aputaulut” sekä Aputaulujen ylläpito –näytön painikkeilla ”Hae attribuutit”, ”Hae koodit”,”Lisää uusi koodi” ja ”Muuta selitettä”. Luokka käyttää *aputaulut*-templatea näytön generoimisessa.

Luokka kerää lomakkeelle syötetyt tiedot **Map**-olioon ja operaatiosta riippuen, suorittaa kannasta haun, tekee kantaan lisäyksen tai päivitysoperaation. Ja lopuksi generoi Aputaulujen ylläpito -näytön ilmoituksineen.

Konstruktori:

public void HelperTableCommand()

Julkiset metodit:

public void execute(CommandDispatcher dispatcher)

Tätä metodia kutsutaan CommandDispatheristä.

public String getName()

Palauttaa komentoluokan *action*-parametrin nimen.

Pakkausnäkyvyyden metodit:

protected ResourceBundle getLanguage(HttpServletRequest request)

Palauttaa kielitukeen tarvittavan resurssikimpun.

protected void prepareDataModel(Map dataModel, HttpServletRequest request)

Hakee *request*-olion parametrien arvot **Map**-olioihin, jotka lopuksi lisätään **Map**-olioon *dataModel*.

protected void addTextsFromResourceBundles()

Alustetaan mallipohjaan liittyvät datamallit template-tiedoston *municipality.ftl* tarvitsemilla staattisilla teksteillä. Datamallit vastaavat järjestelmän tukemia kieliä.

protected void addTexts(Map dataModel, ResourceBundle language)

Lisää resurssikimpun *language* sisältämät staattiset tekstit ja virheilmoitukset annettuun <Map>-olioon *dataModel.*

protected void addMenus(Connection con, Map dataModel)

Lisää **Map**-olioon *dataModel* uuden **Map**-olion *valikko* , johon kerätään html-sivulle tarvittavien valikkojen arvojoukot.

protected String makeQuery(Map dataModel, Connection con, HttpServletRequest request)

Rakentaa kyselyn, jolla kannasta haetaan käyttäjän antamien hakuehtojen mukaiset arvot hakutulokseen.

protected void addSearchResults(Connection con, String kysely, Map dataModel, ResourceBundle language, HttpServletRequest request)

Laittaa annettuun **Map**-olioon *dataModel* merkin staattisten hakutuloksiin liittyvien tekstien näyttämisestä selaimella. Suorittaa **SQL**-kyselyn *kysely* tietokantaan. Kyselyn onnistuessa lisää kyselyn tuloksen annettuun *dataModel*-olioon. Jos haun tulosjoukko on tyhjä, lisää *dataModel*-olioon tiedon, ettei hakuehdoilla löytynyt tietoja. Kyselyn epäonnistuessa lisää *dataModel*-olioon virheilmoituksen ja kyselyn hakuehtoina olleet tiedot.

Yksityiset metodit:

private Map checkInsert(Map dataModel, Connection con, Table table, ResourceBundle language)

Tarkistaa uuden koodin arvot. Jos arvot ovat sallittuja niin palauttaa **Map**-olion *values*, joka sisältää kantaan vietävät arvot. Muutoin palauttaa *null.*

private Map checkComment(Map dataModel, ResourceBundle language)

Tarkistaa voidaanko annetun aputaulun koodin selitettä päivittää. Jos voidaan niin palautetaan päivitystä varten tarvittavat tietokanta-monikon attribuuttien arvot **Map**-olisssa *values.*

private boolean updateComment(Map dataModel, Map values, Connection con, HttpServletRequest request)

Päivittää koodin uuden selitteen kantaan.

private String getParameter(String name, HttpServletRequest request)

Palauttaa pyynnön *request* parametrin *name* arvon **String**-oliona. Jos parametri on *null* tai se sisältää tyhjän merkkijonon, palauttaa tyhjän **String**-olion.

4.6.17 ReportCommand

RaportCommand-luokka generoi halutun raportin ja tallettaa sen tiedostoon.

Luokkaa kutsutaan Raportit-näytöltä. Luokka käyttää Raportit-templatea Raportit-näytön generoimisessa.

Konstruktori:

public void RaportCommand()

public void execute(CommandDispatcher dispatcher)

Tätä metodia kutsutaan CommandDispatheristä.

public String getName()

Palauttaa komentoluokan *action*-parametrin nimen

.

public void generateDisplay(CommandDispatcher dispatcher)

Alustetaan mallipohjaan liittyvät datamallit templaten tarvitsemilla staattisilla teksteillä. Datamallit vastaavat järjestelmän tukemia kieliä.

private void addMenus(Connection con, Map datamalli)

Metodi tekee jokaista alasvetovalikkoa kohden **SQL**-kyselyn tietokantaan. Kyselyn onnistuessa metodi lisää alasvetovalikoiden sisällön parametrina annettuun **Map**-olioon datamalli. Kyselyn epäonnistuessa metodi lisää virheilmoituksen parametrina annettuun **Map**-olioon datamalli.

private void addFieldValues (Connection con, Map datamalli)

Metodi laittaa parametrina annettuun **Map**-olioon datamalli merkin staattisten hakutuloksiin liittyvien tekstien näyttämisestä selaimella. Metodi muodostaa tarvittavan **SQL**-kyselyn tietokantaan luokan **SearchOperation** avulla. Kyselyn onnistuessa metodi lisää kyselyn tuloksen parametrina annettuun **Map**-olioon datamalli. Jos haun tulosjoukko on tyhjä, metodi lisää **Map**-olioon datamalli tiedon, ettei hakuehdoilla löytynyt tietoja. Kyselyn epäonnistuessa metodi lisää **Map**-olioon datamalli virheilmoituksen ja kyselyn hakuehtoina olleet tiedot.

private void addTexts (Map dataModel, ResourceBundle language, String languageChoice)

Lisää tekstit template tiedostoon.

void prepareDataModel(HttpServletRequest request, HttpServletResponse response, boolean message, int amount) throws Exception

Suorittaa alustus toimenpiteitä.

public void printNewNestPDF()

Tulostaa uusi pesä -raportin

public void printOldNestPDF(ResultSet rs, String name, String year, Connection con, String territory_id)

Tulostaa vanha pesä -raportin

private void fillValuesForSecondPage(PdfContentByte cb, ResultSet rs) throws Exception

Tulostaa kannasta haetut arvot vanha pesä -raportin toiselle sivulle.

private void fillValuesForThirdPage(Connection con, PdfContentByte cb, ResultSet rs, String territory_id)

Tulostaa kannasta haetut arvot vanha pesä -raportin kolmannelle sivulle.

private ResultSet getBestNestResult(Connection con, String territory_id)

Suorittaa kyselyn, jolla haetaan tietyn reviirin paras pesimistulos kannasta

private ResultSet getNestsInTerritory(Connection con, String territory_id)

Suorittaa kyselyn, jolla haetaan tietyn reviirin kaikki pesät

private int getSearchResult(Connection con, CommandDispatcher dispatcher)

Tulostaa vanha pesä raportin valmiiseen pdf-pohjaan

private void fillValuesForFirstPage(PdfContentByte cb, ResultSet rs) throws Exception

Tulostaa kannasta haetut arvot vanha pesä -raportin ensimmäiselle sivulle.

4.7 Pakkaus haliaeetus.log

Tässä paketissa on kaksi luokkaa, **LoggingRulesInitializer** ja **HaliaeetusLogger**, jotka yhdessä muodostavat keskitetyn komponentin lokitiedoston käsittelyyn.

Luokkia tehdessä on käytetty hyväksi kirjaa **JavaServlets** (Docendo, 1.painos 2003), jonka ovat tehneet Tero Ahonen, Tapio Hämeen-Anttila ja Kim Åstrand.

4.7.1 LoggingRulesInitializer

Tämä lokiasetusten alustajana toimiva luokka toteuttaa **ServletContextListener**-rajapinnan.

LoggingRulesInitializer lukee **web.xml** tiedostoon määritellyn, kontekstia varten olevan alustus parametrin **loggingRulesFile**, joka kertoo lokia varten tarvittavan asetustiedoston **haliaeetuslog.properties** (ks. luku 4.9.2) sovelluksen kontekstin hakemistossa.

Sovelluksen käynnistyessä tämä kuuntelijaluokka lukee asetustiedoston ja luo tämän avulla **Properties**-objektin sekä tallentaa sen kontekstiin HaliaeetusLogger lokin alustaja luokkaa varten.

Julkiset metodit:

public void contextInitialized(ServletContextEvent event)

> Sovelluksen käynnistyessä lukee asetustiedoston **haliaeetuslog.properties** ja luo sen avulla properties-objektin jonka sijoittaa kontekstiin.

public void contextDestroyed(ServletContextEvent event)

> Sovelluksen sammuessa tallentaa asetukset kontekstista takaisin asetustiedostoon.

4.7.2 HaliaeetusLogger

Luokka toteuttaa Haliaeetusjärjestelmän järjestelmälokin. Toteutuksessa käytetään hyväksi <tt>java.util.logging</tt> pakettia. Luokka tarjoaa metodit lokikirjauksia varten.

Sovelluksen käynnistyessä lukee **LoggingRulesInitializer**-kuuntelijan kontekstiin viemästä **Properties**-objektista kolme parametria *LogName*, joka on **Haliaeetus**-järjestelmälokin

nimi, *logFile*, joka kertoo lokitiedoston sijainnin sovelluksen kontekstin hakemistossa, sekä *logLevel*, joka sisältää tiedon lokitustasosta. Näitä parametreja tarvitaan **java.util.logging.Logger**-objektin luomista varten.

Julkiset metodit:

public void contextInitialized(ServletContextEvent event)

Metodi suoritetaan sovelluksen käynnistyksessä. Luo ensin lokitiedoston. Hakee kontekstista **Properties**-objektista lokin nimen, lokitiedoston sijainnin ja lokitustason ja vie kontekstiin näillä parametreilla luodun lokin *systemlog*. Tämän jälkeen luo **Logger**-objektin, jolle annetaan parametriksi konteksitissa oleva järjestelmäloki *systemlog*.

public void contextDestroyed(ServletContextEvent event)

Metodi suoritetaan sovelluksen alasajon aikana. Poistaa Haliaeetus-järjestelmälokin kontekstista (*systemlog*).

public void setLogLevel(int level, Logger log)

Metodi muuttaa parametrina saadun lokin *log* lokitustason parametrina saadun tason *level* mukaiseksi.

public static void severeLogMessage(String severe)

Kirjoittaa parametrina annetun virheilmoituksen *severe* lokiin. Virheilmoituksen taso (**java.util.logging.Level**) on **SEVERE**.

public static void fineLogMessage(String fine)

Kirjoittaa parametrina annetun virheilmoituksen *fine* lokiin. Virheilmoituksen taso (**java.util.logging.Level**) on **FINE**.

public static void finerLogMessage(String finer)

Kirjoittaa parametrina annetun virheilmoituksen *finer* lokiin. Virheilmoituksen taso (**java.util.logging.Level**) on **FINER**.

public static void finestLogMessage(String finest)

Kirjoittaa parametrina annetun virheilmoituksen *finest* lokiin. Virheilmoituksen taso (**java.util.logging.Level**) on **FINEST**.

public static void infoLogMessage(String info)

Kirjoittaa parametrina annetun virheilmoituksen *info* lokiin. Virheilmoituksen taso (**java.util.logging.Level**) on **INFO**.

public static void logP(Level level, String sourceClass, String sourceMethod, String msg, Throwable thrown)

Kirjaa lokiin lokitiedon, jonka taso (**java.util.logging.Level**) on *level*, lähdeluokka on *sourceClass*, lähdemetodi on *sourceMethod* viesti on *msg* ja virheilmoitus *thrown*.

public static void logP(Level level, String sourceClass, String sourceMethod, String msg)

Kirjaa lokiin lokitiedon, jonka taso (**java.util.logging.Level**) on *level*, lähdeluokka on *sourceClass*, lähdemetodi on *sourceMethod* viesti on *msg*.

4.8 Sekvenssikaaviot

Haliaeetus-järjestelään liittyvät sekvenssikaaviot on esitetty liitteessä A.

4.9 Muut tiedostot

Järjestelmä käyttää mallipohjia (template-tiedostot) dynaamisten html-sivujen tuottamiseen. Lisäksi järjestelmässä on yksi staattinen html-sivu, jolta käyttäjä kirjautuu järjestelmään. Järjestelmän kielituki toteutetaan Javan properties-tiedostoilla. Luokka Tarkista käyttää myös properties-tiedostoja apunaan.

4.9.1 Template-tiedostot

Servlet-luokkien tuottamat tulosteet muunnetaan käyttäjän ymmärtämään HTML-muotoon FreeMarker-mallipohjien eli template-tiedostojen (.ftl) avulla. Jokaista näyttöä kohden on yksi mallipohja paitsi näytöt Uusi pesä, Pesätiedot ja Uusi tarkastus käyttävät samaa mallipohjaa. Lisäksi on mallipohja sivujen yläosan linkkejä varten.

Template-tiedostot sisältävät tavallista HTML-koodia sekä FreeMarker-komentoja, joiden avulla on toteutettu mm. ehtorakenteet ja luettelomuotoisen tiedon esitys. Template-tiedostojen avulla servletti voi generoida käyttöliittymän näytölle valmiita lomakepohjia, listata list-operaatiolla dynaamisesti tulosjoukkoja näytölle ja yhdistellä eri tavoin näytöllä esitettäviä osia include-komennoilla. Template-tiedostot vastaavat käyttöliittymän eri näyttöjä ja niiden avulla voidaan esim. if-lauseita käyttäen generoida näytön virheilmoituksia.

HTML-lomakkeissa käytetyt kenttien ja FreeMarker-muuttujien nimet vastaavat (soveltuvilta osin) tietokannan taulujen ja sarakkeiden nimiä (esim. muodossa taulu.attribuutti). Template-tiedostot sijaitsevat palvelinkoneella bodbacka.cs.helsinki.fi.

Olennaisin asia templateissa on datamalli, jonka Servletti luo. Datamalli on normaali puutietorakenne. Datamallista template voi suoraan sijoittaa html:n sekaan muuttujia. Tässä luvussa keskitytään datamallin rakenteeseen, koska itse ftl-tiedostot ovat melko triviaaleja html-kieltä ja perusohjelmointia tuntevalle, ja tiedostojen perusrakenteen pystyy näin ollen päättelemään näyttöjen kuvista kappaleessa 5. Tässä dokumentissa datamalli kuvataan seuraavasti puuna:

```
(juuri)
+- muuttuja
+- solmu
| +-a
| | +-arvo
| +-b
+-sekvenssi
| +-0
| +-1
| +-n
```

Mallissa jokainen puun normaali solmu voi toimia tyhjänä solmuna (=hakemisto), muuttujana (sisältää arvon), tai olla molempia. Sekvenssityyppiset solmut taas toimivat kuin taulukko. Esimerkissä siis sekvenssi toimii kuin taulukko, ja viittaus sekvenssi[1] viittaa sekvenssin toiseen muuttujaan/alkioon. Sekvenssin alkiot voivat toimia myös hakemistorakenteina, ja yhdessä alkiossa voi olla monta muuttujaa. Tavallisiin muuttujiin viitataan templatessa esim solmu.a.arvo. Haliaeetus-ratkaisussa kaikki näyttöjen staattiset tekstit haetaan kielituen takia tiedostosta. Tämä aiheuttaa sen, että datamallipuista tulee isoja ja tällä esitystavalla graafisesti korkeita (itse puuhan on melko matala).

navi.ftl `navi.ftl` on navigointipalkin template, joka näkyy jokaisen näytön ylälaidassa (esim. kuva reffig:utvp). Se sisällytetään muihin templateihin include-lauseella. Solmu `teksti` sisältää alimuuttujinaan kaikki palkin staattiset tekstit. Muuttuja `kutsuja` taas kertoo, mikä servletti on kutsunut sivua, eli mille näytölle palkki kulloinkin tulee. Tällä tavalla oikea sivu voidaan korostaa palkista eri värillä. (Liite B:navi.ftl)

login.ftl `login.ftl` tuottaa sisäänkirjautumissivun. Solmun `teksti` alimuuttujat sisältävät näytön staattiset tekstit. Jos solmun `ilmoitus` arvo on true, `syy`-sekvenssiin on ladattu virheilmoitukset muuttujiin `ilmoitus.syy[n].teksti`. Muuttuja `arvo.tunnus` säilöö syötetyn käyttäjätunnuksen virhetilanteissa, jotta käyttäjätunnus-kenttä ei turhaan tyhjene. (Liite I: login.ftl)

haku.ftl Hakusivun staattiset tekstit ladataan datamallin solmun `teksti` alimuuttujista. Myös solmulla `tulos` on alimuuttujina tekstejä, jotka toimivat hakutuloksen sarakkeiden otsikkoina. Haun tulosjoukko voi olla kooltaan 0-n riviä. Muuttuja `tulos.joukko` on sekvenssi, jossa jokaisella muuttujalla (0-n) on omat neljä saraketta: esim. `tulos.joukko[2].numer` `tulos.joukko[2].nimi`,`tulos.joukko[2].reviiri`,`tulos.joukko[2].kunta`. Tulosjoukko listataan näytölle list-komennolla, jos solmu `tulos="true"`. Näytöllä on lisäksi valikkoja, joihin täytyy hakea vaihtoehdot tietokannasta. Nämä sekvenssit ovat solmun `valikko` alla. Aputaulu-vaihtoehtojen lisäksi kaikilla valikkojen sekvensseillä on alimuuttuja valittu, joka kertoo tarvittaessa valitun arvon haun jälkeen. Ilmoitukset (esim. ”valituilla hakuehdoilla ei löytynyt pesiä”) ovat solmun `ilmoitus` alla. Solmun `arvo` alimuuttujina on kaikki käyttäjän tekstikenttiin syöttämät arvot, jotka on näytettävä myös haun jälkeen. Muuta-solmussa on muuttujia, joita NestInformationCommand luokka ja pesa.ftl tarvitsee, kun kontrolli siirtyy hakusivulta eteenpäin NestInformationCommand luokalle. (Liite D: mainscreen.ftl)

pesa.ftl `Pesä.ftl` toimii näyttöjen Uusi pesä, Uusi tarkastus ja Pesätiedot pohjana. Datamallin muuttuja kutsuja sisältää tiedon kutsuvasta servletistä. Tämän tiedon perusteella template laittaa oikeat tiedot sivuille if-lauseiden avulla. Datamallin solmun teksti alla on kaikki näytön staattiset tekstit. Suurin osa teksteistä ladataan jokaiselle kolmelle sivulle, mutta esim. kenttien disablointi vaihtelee sivujen välillä, käyttötapauksesta riippuen. Datamallin solmun `arvo` alla on kaikki tekstikenttiin tietokannasta ladattavat arvot. Muuttujat on luotava Servletissä, vaikka niissä ei olisi mitään tietoa (arvoksi annetaan siis tyhjä merkkijono esim. arvo.lahetetty.nayte_i=””). Näin toimittaessa templaten

ei tarvitse huolehtia muuttujien olemassaolosta. Riittää kun muuttuja tulostetaan näytölle, oli sillä sitten oikeaa sisältöä tai ei. Näitä muuttujia tarvitaan näytöissä Uusi tarkastus ja Pesätiedot. Solmun valikko alla on kaikki alasvatovalikoihin ladattavat aputauluarvot. Nämä ladataan kaikille kolmelle näytölle list-lauseiden avulla. Jokaisella valikkosekvenssin alkiolla on lisäksi muuttujat valittu, joka kertoo valittuna olevan vaihtoehdon näytöillä Uusi tarkastus ja Pesätiedot ja teksti, joka sisältää itse valikkoon ladattavan arvon. Solmun ilmoitus arvo kertoo, onko esim. lisäysoperaatiossa ollut virheitä. Jos on, virheet talletetaan sekvenssiin ilmoitus.syy, ja virheilmoitukset tulostetaan näytölle muuttujista ilmoitus.syy[n].teksti. (Liite F: pesa.ftl)

myrkyt.ftl, saaliit.ftl, historia.ftl `myrkyt.ftl`, `saaliit.ftl`, ja `historia.ftl` ovat templatet, joita ei toteutettu tämän projektin puitteissa. Näytöistä on olemassa html-tiedostot, mutta mitään todellista toiminnallisuutta niihin ei ole suunniteltu eikä toteutettu.

reviirit.ftl `reviirit.ftl` on reviirien lisäys- ja hakusivu. Solmussa `teksti` on kaikki näytön staattiset tekstit. Solmussa `navi` on navigointipalkin template, joka tulostetaan sivun ylälaitaan ja jossa linkit on muille sivuille. (Liite H: reviirit.ftl)

municipality.ftl `municipality.ftl` on kuntatietojen ylläpitosivu, jolla voidaan luoda uusia kuntia ja tehdä kuntaliitoksia sekä muokata kunnan tietoja. Solmussa `teksti` on kaikki näytön staattiset tekstit. Solmussa `valikko` ovat suuralueiden ja ympäristökeskusten tietokannan aputaulusta ladattavat koodit. Solmussa `tulos` on kuntien haun tulosrivit. Näitä ovat liitettävien/muutettavien kuntien kuntatunnukset, koordinaatit sekä suuralue ja ympäristökeskus – alasvetovalikoihin tietokannan aputauluista ladattavat koodit. Solmussa `navi` on navigointipalkin template, joka tulostetaan sivun ylälaitaan ja jossa linkit on muille sivuille. (Liite E: municipality.ftl)

tarkastajat.ftl `tarkastajat.ftl` on tarkastajatietojen ylläpitosivu, jolla voidaan lisätä tarkastajia ja muuttaa tarkastajien nimeä. Solmussa `teksti` on kaikki näytön staattiset tekstit. Solmussa `arvo` tekstikenttien alkuarvot. Solmussa `navi` on navigointipalkin template, joka tulostetaan sivun ylälaitaan ja jossa linkit on muille sivuille. (Liite J: tarkastajat.ftl)

aputaulut.ftl `aputaulut.ftl` on aputaulujen ylläpitosivu, jolla voidaan lisätä aputauluun koodeja ja muuttaa koodien selitteitä. Solmussa `teksti` on kaikki näytön staattiset tekstit. Solmussa `muuta` ovat tietokannasta ladattavat tekstikenttien alkuarvot eli kyseiseen attribuuttiin liittyvät ennestään tallennetut koodit ja selitteet. Solmussa `valikko` ovat taulu- ja attribuutti-alasvetovalikoihin ladattavat taulujen ja attribuuttien nimet. Solmussa arvo säilytetään valikon valinnat . Solmussa `navi` navigointipalkin template, joka tulostetaan sivun ylälaitaan ja jossa on linkit muille sivuille. (Liite C: aputaulut.ftl)

raportit.ftl `raportit.ftl` on raporttien tallennussivu. Solmussa `teksti` on kaikki näytön staattiset tekstit, joista raporttien nimet muodostavat sekvenssin (raportti[0], raportti[1],...). Solmussa `valikko` on tietokannan aputaulusta ladattavat suuralueiden ja ympäristökeskusten koodit. Solmussa `ilmoitus` virheilmoitus loppu_ennen_alkua, joka tulostetaan näytölle, mikäli loppuvuosi on alkuvuotta aikaisempi. Solmussa `navi` on navigointipalkin template, joka tulostetaan sivun ylälaitaan ja jossa linkit on muille sivuille. (Liite G: report.ftl)

4.9.2 Properties-tiedostot

Properties-tiedostot ovat tekstitiedostoja, jotka sisältävät avain-arvoparin yhtäläisyysmerkillä erotettuna (esim. language=Kieli). Kukin avain-arvopari on tiedostossa omalla rivillään. Avaimissa ja yhtäläisyysmerkin ympärillä ei saa olla välilyöntimerkkejä, arvoissa sitä vastoin välilyönnit ovat sallittuja. Isot ja pienet kirjaimet ovat merkitseviä.

Kielituen tiedostot Kielituki toteutetaan properties-tiedostojen avulla. Kielitukitiedostoihin sijoitetaan generoitavien html-sivujen staattiset tekstit, virheilmoitukset ja muut ilmoitukset. Tuettavat kielet ovat suomi, ruotsi ja englanti. Kullekin kielelle luodaan oma properties-tiedosto. Avaimet ovat kaikissa properties-tiedostoissa samat, mutta arvona tiedostossa on kyseisen kielen mukainen teksti merkkijonona. Haliaeetus-järjestelmässä avain on sama kuin template-tiedoston muuttuja, johon teksti kohdistetaan. Properties-tiedostot sijoitetaan toteutusvaiheessa palvelinkoneelle alkokrunni.cs.helsinki.fi samassa kansioon kuin servletit.

Haliaeetus-järjestelmässä kielituki käyttää tiedostoja:

haliResources_fi.properties

haliResources_sv.properties

haliResources_en.properties

Tiedostot nimetään seuraavan syntaksin mukaisesti:
haliResources_[ISO-639 kielikoodi].properties.

Alla on esimerkki kolmesta properties-tiedostosta suomeksi, ruotsiksi ja englanniksi. Esimerkeissä avaimet ovat englanniksi, mutta toteutusympäristössä ne ovat suomeksi, jolloin ne vastaavat template-tiedostoissa käytettäviä muuttujia.

Tiedosto haliResources_fi.properties sisältää tuen suomen kielelle. Esimerkki tiedoston sisällöstä:

language=Kieli

compute=Laske

whiteTailedEagle=Merikotka

Tiedosto haliResources_sv.properties sisältää tuen ruotsin kielelle. Esimerkki tiedoston sisällöstä:

language=Språk

compute=Räkna

whiteTailedEagle=Havsörn

Tiedosto haliResources_en.properties sisältää tuen englannin kielelle. Esimerkki tiedoston sisällöstä:

language=Language

compute=Compute
whiteTailedEagle=White-tailed eagle

Ylläpitäjän on huolehdittava kielitiedostoista. Jos uusi käännettävä sana tarvitaan, ylläpitäjän on lisättävä se samalla avaimella ja kielen mukaisella arvolla jokaiseen properties-kielitiedostoon.

Java käyttää Unicode-merkistöä, jolloin kielituki voidaan toteuttaa tarvittaessa kaikilla maailman kielillä. Properties-tiedostojen avulla kielivalikoiman laajentaminen on helppoa; servletteihin tarvitsee tehdä vain vähäisiä lisäyksiä uuden kielen tukea varten.

Komentoluokkien Properties-tiedostot Haliaeetus käyttää komento-luokille konfiguroitavia properties-tiedostoja.

Tiedostossa yhdistyy käyttöliittymän kutsut oikeaan kutsuttavaan komento-luokkaan.

Tiedoston rakenne on seuraavanlainen:

Search=haliaeetus.command.SearchCommand MainScreen=haliaetus.command.MainScreenCommand

Lokin asetustiedosto Haliaeetus-järjestelmän lokia varten tarvitaan asetustiedosto **haliaeetuslog.properties**. Asetustiedostossa sisältää lokituksen tason, lokin nimen, sekä lokitiedoston sijainnin järjestelmän hakemistorakenteessa.

Tiedoston rakenne on seuraavanlainen:

logLevel=0
logName=haliaeetus_systemlog
logFile=WEB-INF/log/haliaeetus_log.txt

4.9.3 Tyylitiedosto

Tyylitiedosto tyyli.css sijoitetaan samaan hakemistoon, kuin template-tiedostot. Tiedosto toimii html-tyylitiedostona, joka kuvaa html-sivujen ulkoasua.

4.9.4 Staattiset html-tiedostot

Järjestelmän aloitussivu index.shtml on ainoa staattinen html-sivu. Sivu on suojattu sivu. Käyttäjän on annettava oma käyttäjätunnus ja salasana jotta päästään Haliaeetus-järjestelmän kirjautumissivulle. Sivu siis kutsuu Hali-servlettiä, jos käyttäjällä oli oikeat tunnukset.

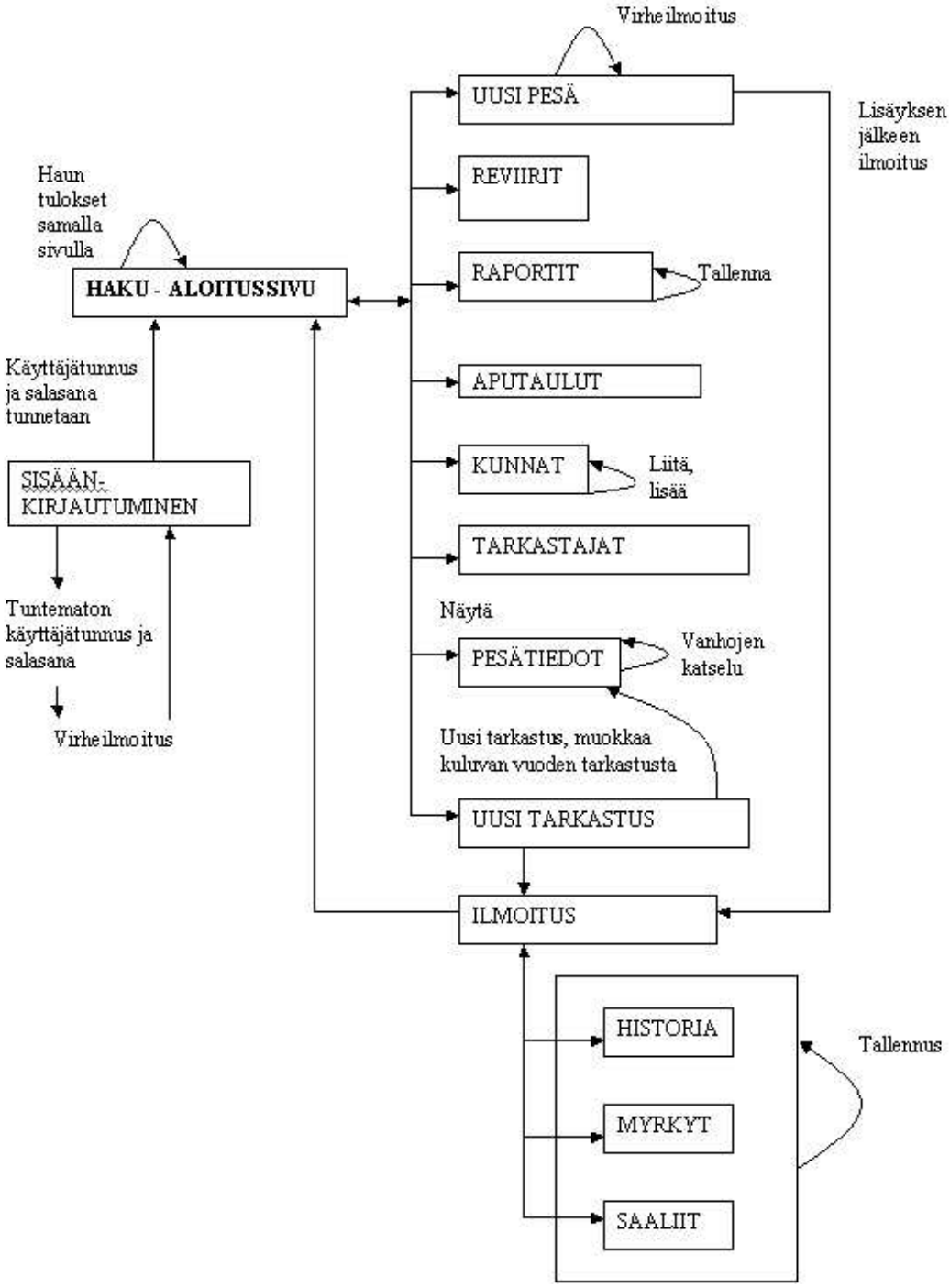
Tiedosto sijaitsee palvelinkoneella alkokrunni.cs.helsinki.fi.

5 Käyttöliittymä

Tässä luvussa kuvataan järjestelmän käyttöliittymä. Suunnittelun lähtökohtana ovat vaatimusdokumentissa [Kot04b]kuvatut käyttötapaukset. Käyttötapaukset on kuvattu vaatimusdokumentissa sanallisesti, ja niiden pohjalta on tehty näyttökuvien sarjat, joissa käyttäjä saavuttaa tavoitteensa.

Käyttöliittymässä on 12 erilaista näyttöä: sisäänkirjautuminen, haku, uusi tarkastus, myrkyt, saaliit, historia, uusi pesä, pesän tietojen muokkaus, raportit, aputaulut, ilmoitussivu, laji, tarkastajat ja kunnat. Näytöt historia, myrkyt ja saaliit toteutetaan tämän projektin puitteissa, vain jos aikaa jää.

Haku-sivu on sisäänkirjautumisen jälkeen ensimmäinen sivu. Navigointipalkista, joka on kaikkien sivujen yläreunassa, pääsee näyttöihin Uusi pesä, Raportit, Aputaulut, Tarkastajat, Laji ja Kunnat. Näistä neljä viimeistä ovat ylläpitonäyttöjä. Uusi pesä-näytöllä lisätään uusi pesä ja Raportit-näytöltä saadaan aikaan erilaisia raportteja. Haku-sivulla tehdyn haun perusteella pääsee jo olemassa olevalle pesälle tekemään uuden tarkastuksen (Uusi tarkastus) tai muuttamaan vanhoja tietoja (Muuta). Näistä näytöistä pääsee myös näyttöihin Myrkyt, Saaliit ja Historia, jotka liittyvät aina tiettyyn pesään. Käyttöliittymä on kuvattu tarkemmin luvussa 5.1.



Kuva 5: Käyttöliittymän näyttöjen väliset suhteet

5.1 Kuvasarjat

Kuvasarjoissa kuvataan näyttö näytöltä, kun käyttäjä pyrkii tavoitteeseensa järjestelmän avulla. Tällä hetkellä näytöt on toteutettu html:llä, eikä niiden koodi sisällä mitään toiminnallisuutta. Projektin edetessä html-koodiin lisätään toiminnalisuus, mutta näyttöjen ulkoasu tulee säilyttämään muotonsa sellaisena kuin ne seuraavissa kuvissa esitetään.

5.1.1 Käyttäjän sisäänkirjautuminen

Sisäänkirjautuminen -sivulla käyttäjä kirjoittaa sivun kenttiin käyttäjätunnuksen ja salasanan sekä painaa Sisään -nappia. KÄyttäjä voi myös vaihtaa etusivulla järjestelmän käyttämää kieltä. (**Kuva 5**).

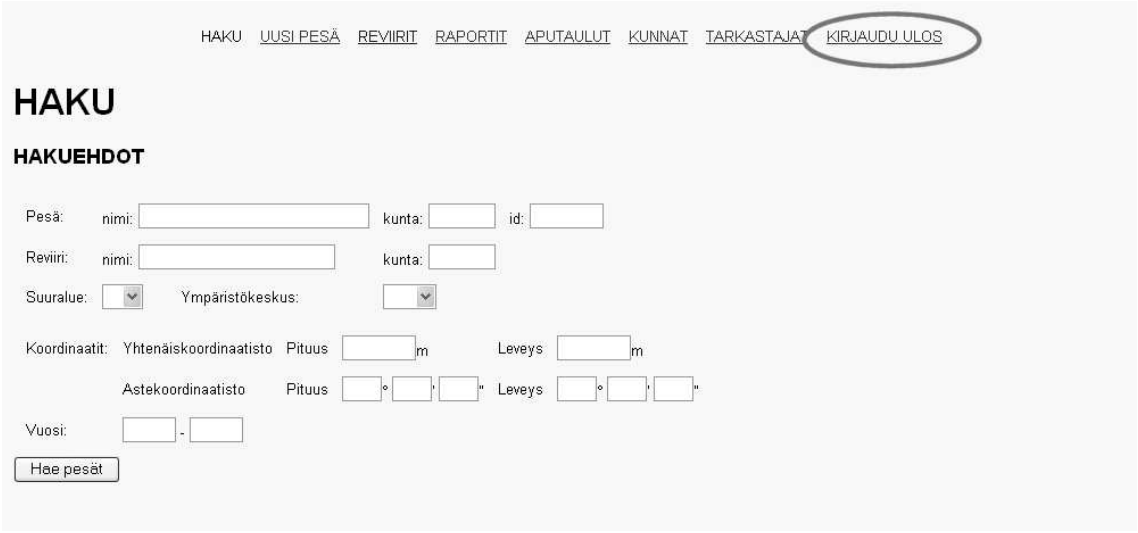
Kuva 6: Sisäänkirjautuminen

Aloitussivuna on Haku-sivu. Käyttäjä kirjautuu ulos Kirjaudu ulos-linkistä (**Kuva 6**). Uloskirjautumisen jälkeen palataan jälleen sivulle Sisäänkirjautuminen, jossa tekstikenttien yläpuolella on kommentti ”Olet kirjautunut ulos järjestelmästä”.

5.1.2 Uuden pesän lisääminen

Kun käyttäjä haluaa lisätä kantaan kokonaan uuden pesän, hän painaa navigointipalkin Uusi pesä-linkkiä. Sivulla on lomake, johon uuden pesän tiedot lisätään. Pakolliset kentät on merkitty tähdellä. Lopuksi käyttäjä painaa sivun alareunassa olevaa Tallenna -nappia (**Kuvat 7,8,9** ja **10**). Sulje -painike ei tallenna mitään tietoa kantaan, vaan käyttäjä palaa järjestelmän aloitussivulle.

Tallenna -painikkeen painamisen jälkeen käyttäjä saa tiedon onnistunesta talletuksesta(**Kuva 11**). Linkeistä Myrkyt, Saaliit ja Historia pääsee tallettamaan lisää tätä pesää



Kuva 7: Uloskirjautuminen

koskevaa tietoa. Kyseiset näytöt esitellään myöhemmin tässä luvussa. Takaisin aloitussivulle -linkki vie käyttäjän takaisin järjestelmän etusivulle.

5.1.3 Aputaulun päivitys

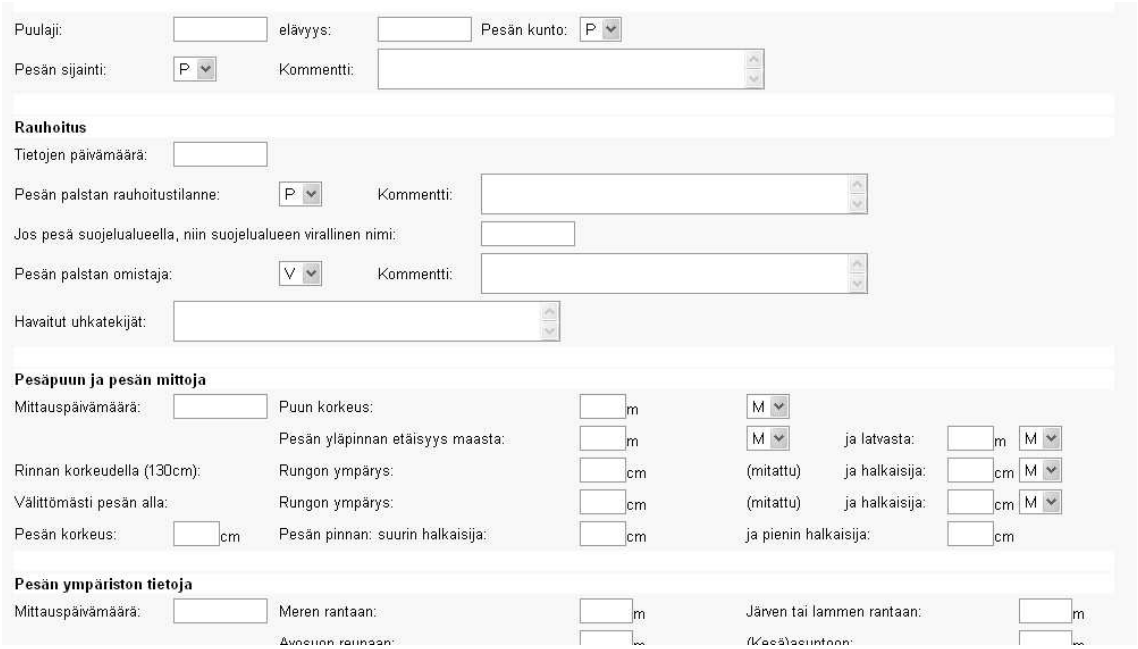
Käyttötapaus 6:n variaatiossa, käyttäjä joutuu lisäämään aputauluun uuden koodin. Tämä johtuu siitä, että haluttua koodia ei järjestelmästä vielä löydy. Käyttäjä menee Aputaulujen ylläpito -sivulle navigointipalkin linkillä Aputaulut. Sivulla käyttäjä valitsee haluamansa taulun alasvetovalikosta (**Kuva 12, 1.**).Tämä aiheuttaa sen, että Attribuuttialasvetovalikkoon ilmestyy kaikki kyseisen taulun aputaulua käyttävät attribuutit. Kun oikea attribuutti valitaan (**Kuva 12, 2.**), alapuolelle ilmestyy lista kyseisen attribuutin selitteistä. Tyhjälle riville lisätään uusi koodi ja selite (**Kuva 12, 3.**), sekä painetaan Lisää-nappia (**Kuva 12, 4.**).

Lisää-napin painamisen jälkeen uusi koodi ja selite ilmestyy listaan. Selitteitä voi korjata tekemällä korjaukset suoraan selite-kenttään ja painamalla kyseisen rivin Muuta-nappia. Tällöin järjestelmä antaa varoituksen siitä, että käyttäjä varmasti tietää mitä on tekemässä. Mahdolliselle uudelle koodille on myös ilmestynyt tyhjä rivi.

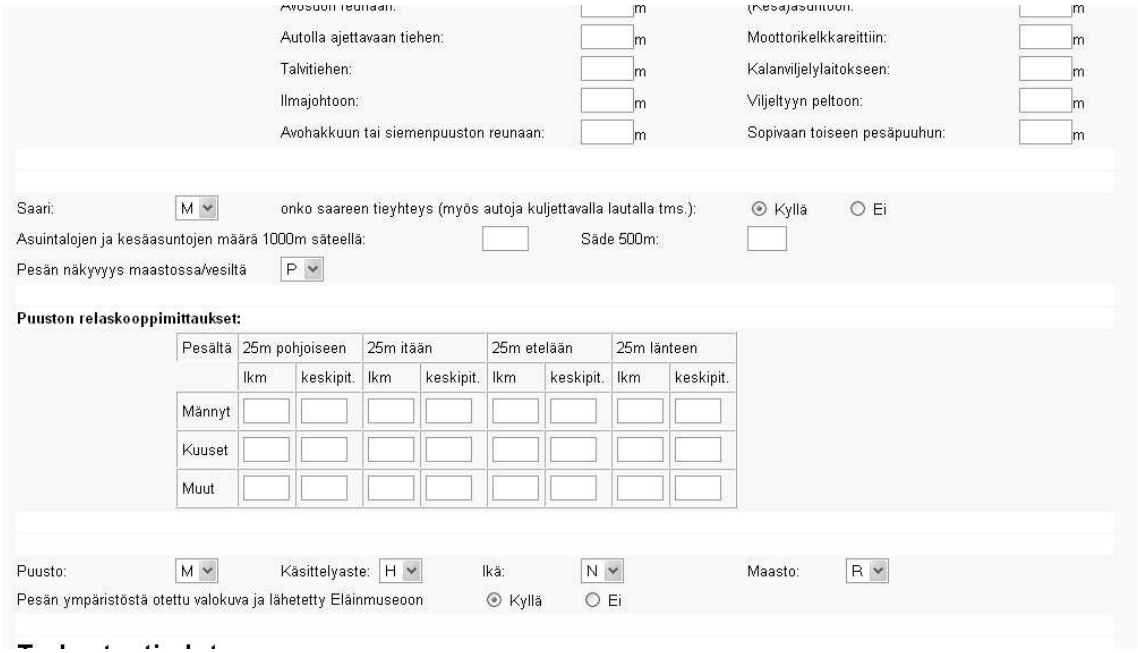
5.1.4 Vanhan pesän tarkastus

Käyttäjä hakee Haku-sivulla haluamansa pesät täyttämällä sopivat hakuehdot (tässä tapauksessa reviirin nimi), ja painamalla Hae Pesät-nappia (**Kuva 14**).

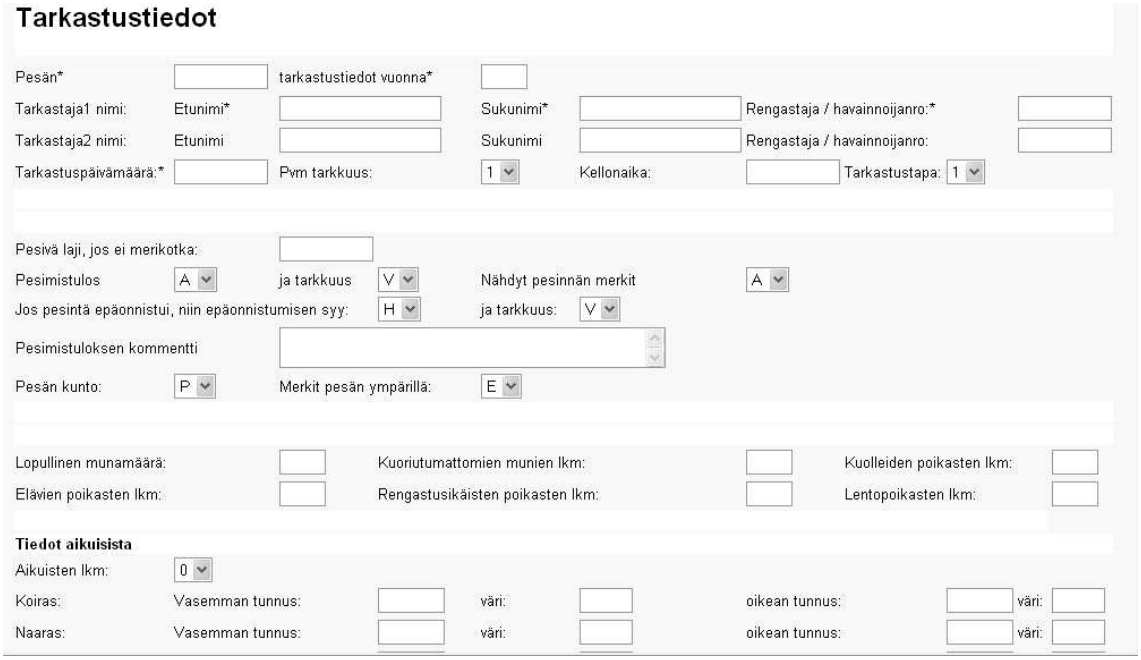
Kuva 8: Uuden pesän lisääminen 1



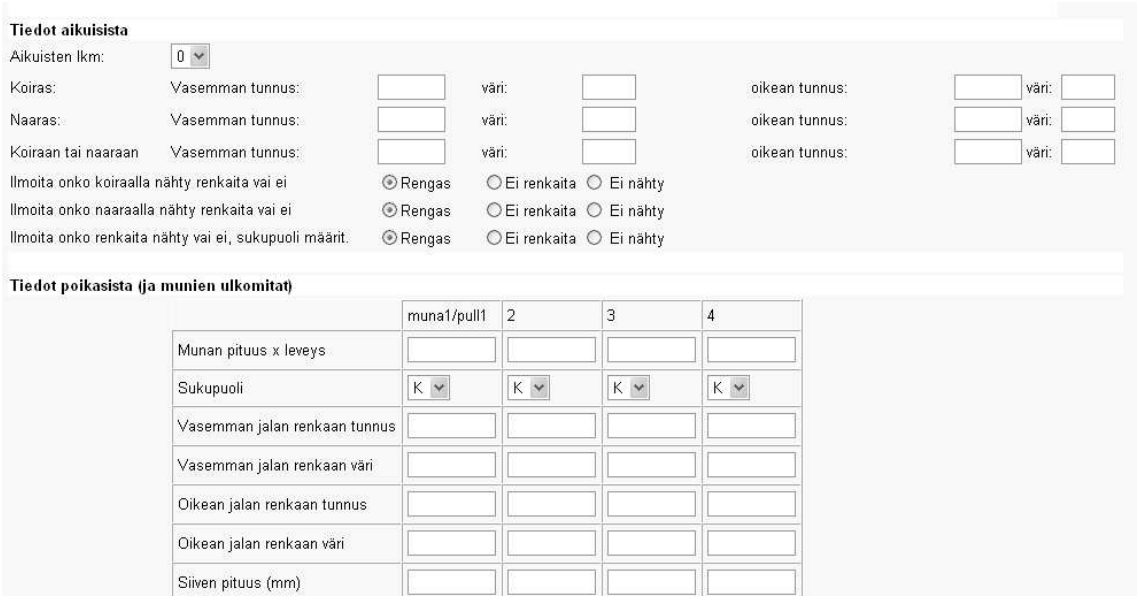
Kuva 9: Uuden pesän lisääminen 2



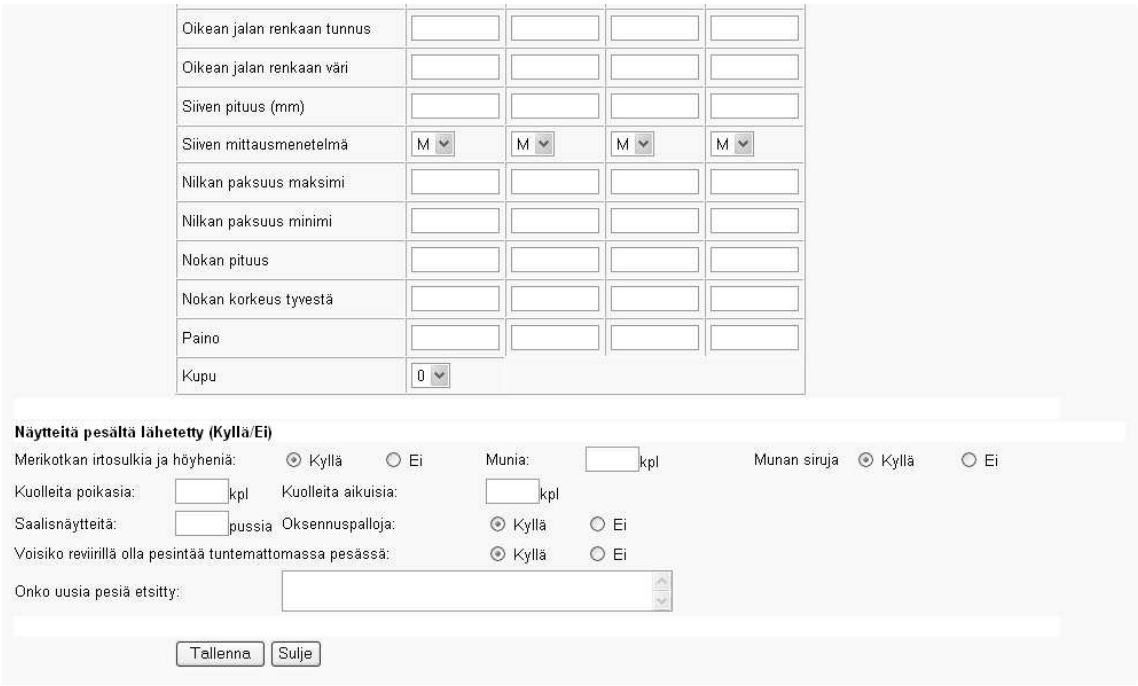
Kuva 10: Uuden pesän lisääminen 3



Kuva 11: Uuden pesän lisääminen 4

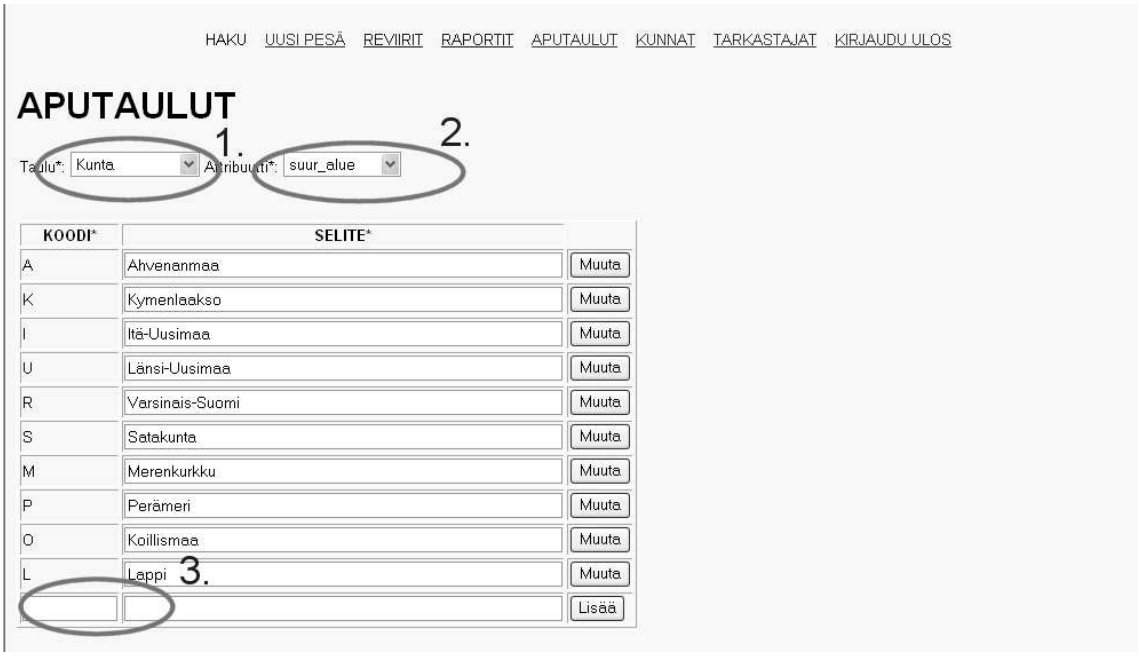


Kuva 12: Uuden pesän lisääminen 5



Kuva 13: Uuden pesän lisääminen 6

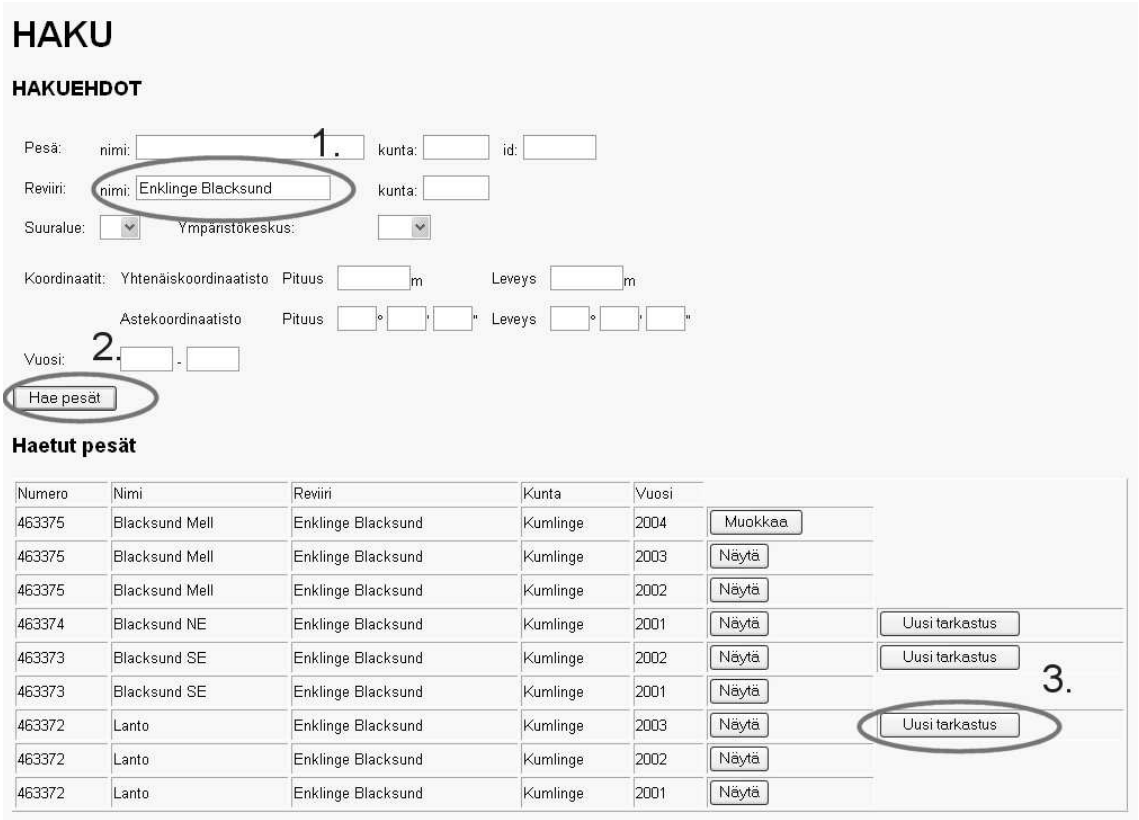
Kuva 14: Ilmoitus onnistuneesta lisäyksestä



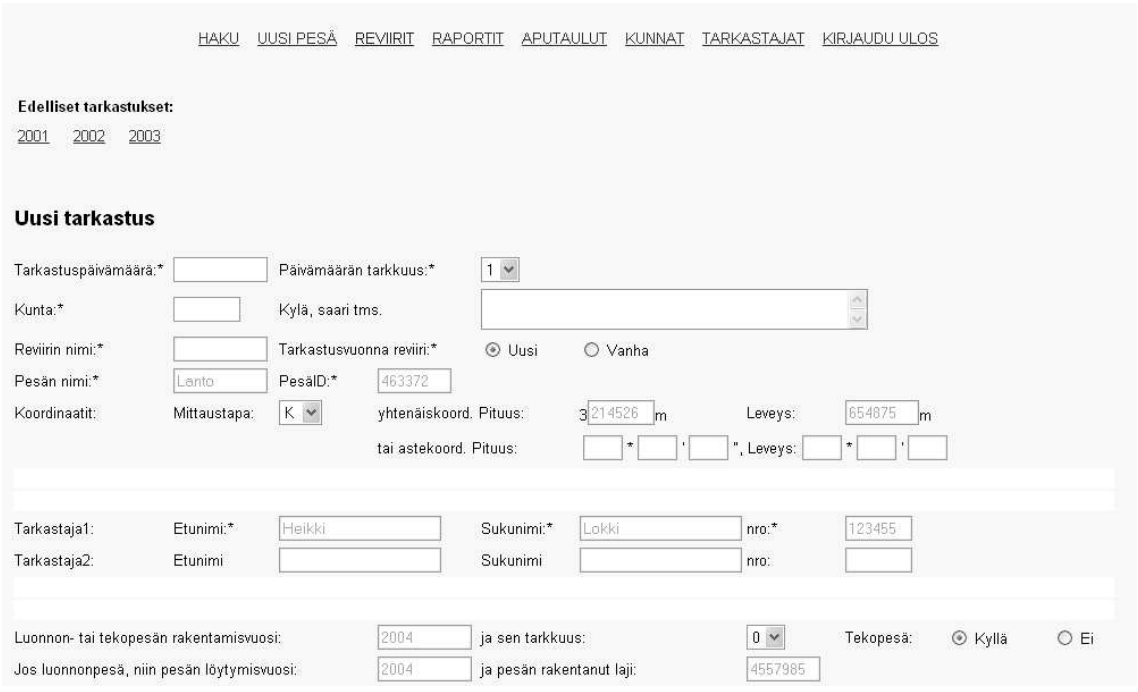
Kuva 15: Aputaulun päivitys

Hakutulokset ilmestyvät listaksi näytön alareunaan. Painamalla Näytä -painiketta, käyttäjä pääsee katselemaan aiempina vuosina pesälle tehtyjä tarkastuksia. Näitä tietoja käyttäjä ei voi muuttaa. Jos pesällä on kuluvana vuonna tehty jo tarkastuksia, käyttäjä voi painaa Muokkaa -painiketta ja jatkaa kyseisen pesän tietojen tallentamista. Mikäli pesälle ei ole kuluvana vuonna tehty vielä yhtään tarkastusta, on kyseessä uusi tarkastus ja käyttäjä painaa Uusi tarkastus -painiketta. Pesävakion ja edellisuvoden tiedot ovat esitäytettynä lomakkeessa.(**Kuva 15**).

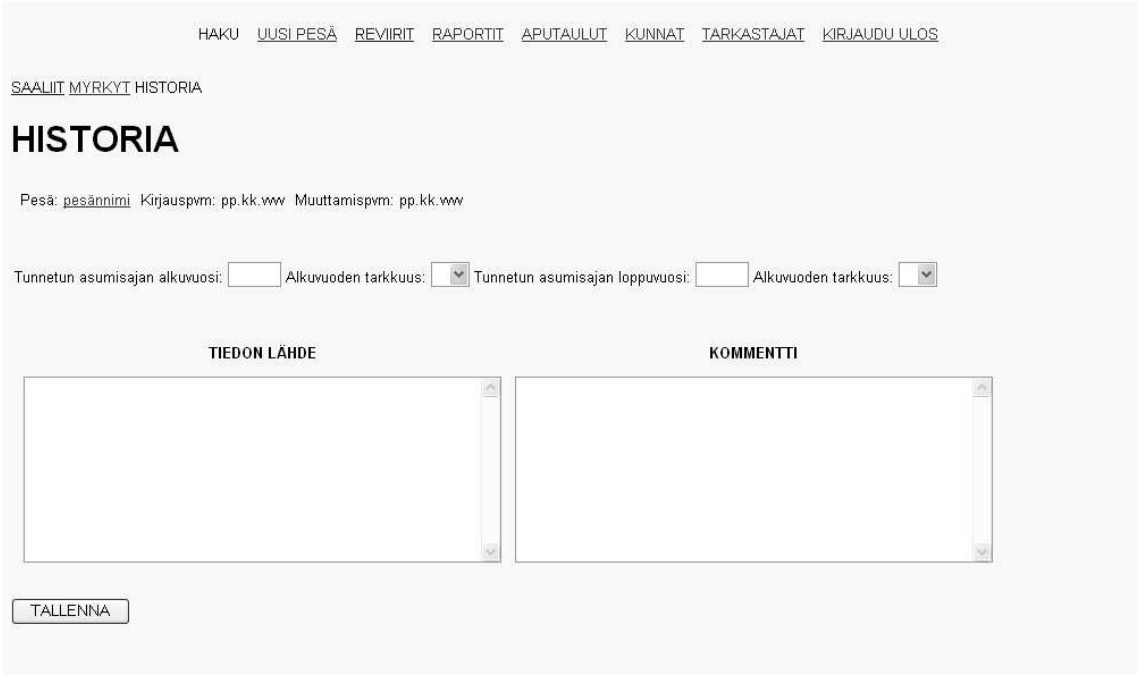
Lomake on pääosin esitäytetty edellisen vuoden tietojen perusteella. Tarkastupäivä on kuitenkin syötettävä. Ja jos muutoksia on tullut, voi vanhoja tietoja muuttaa. Pesän vanhempiin tarkastuksiin pääsee sivun yläosassa olevilla linkeillä.



Kuva 16: Vanhan pesän hakeminen 1



Kuva 17: Uusi tarkastus vanhalle pesälle



Kuva 18: Historia -näyttö

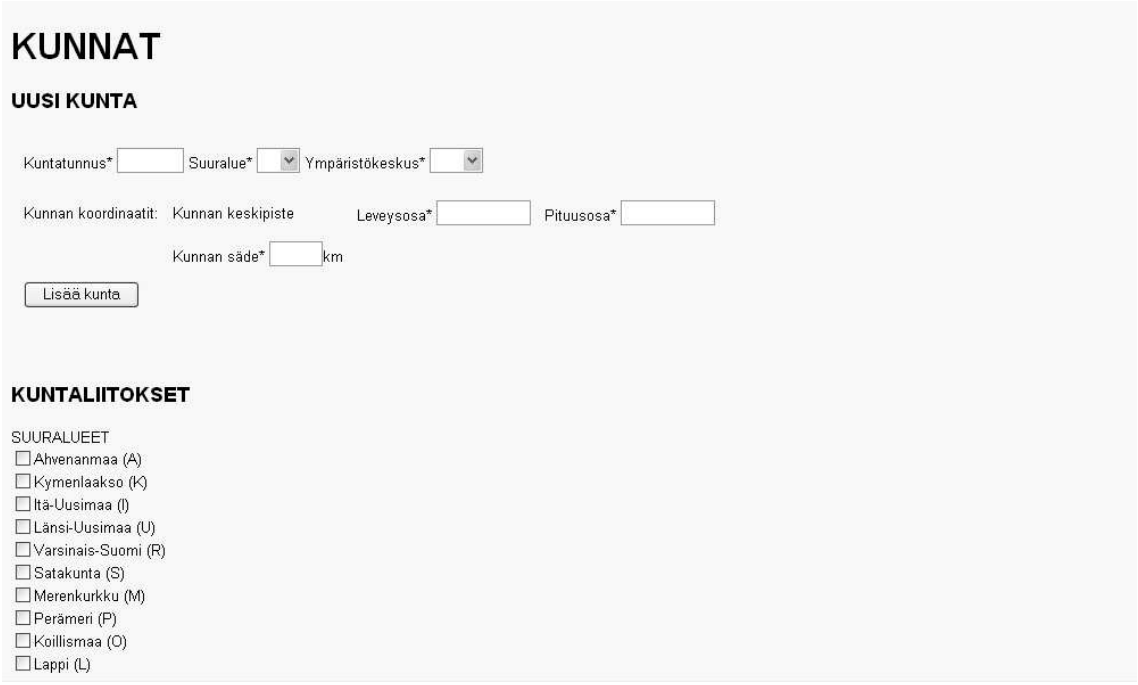
Tarkastustietoja (Näytöllä otsikot Tarkastus, Aikuiset linnut ja Poikaset) ei ole esitäytetty, eli lomakkeen alaosa on tyhjennetty edellisen vuoden tiedoista. Käyttäjä täyttää nämä tiedot ja painaa Tallenna -nappia (**Kuva 17**). Seuraus tästä on sama kuin Uuden pesän lisäämisessä. Muut reviirin pesät lisätään samalla tavalla.

5.1.5 Historia-näyttö

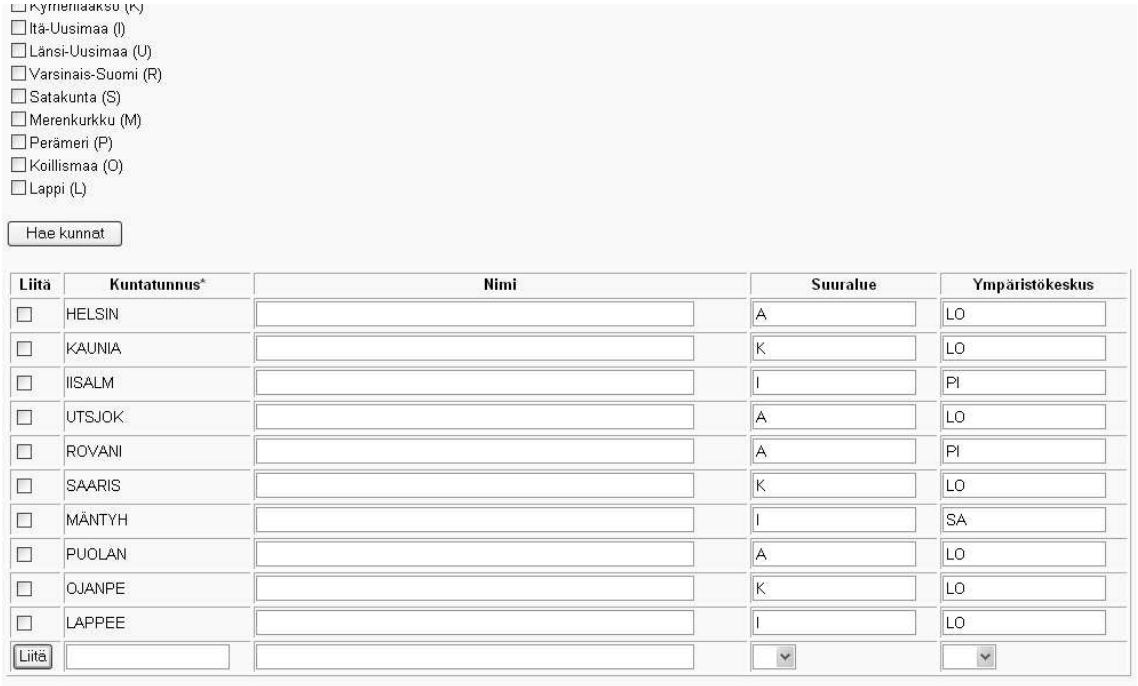
Historia-näytöllä voidaan tallettaa, katsella ja muuttaa tietyn pesän historiatietoja. Historia-sivulle pääsee tietyn pesän Pesätiedot-näytöltä.

5.1.6 Kunnat-näyttö

Kunnat-näytöllä voi lisätä uuden kunnan (**Kuva 24, 1.**). Kuntaliitosta varten on ensin haettavat halutut kunnat tietyltä suuralueelta tai –alueilta ruksimalla suuralueet ja painamalla Hae kunnat-nappia (**Kuva 24, 2.**). Tämän seurauksena näytön alareunaan tulee taulukko kunnista valituilla suuralueilla (**Kuva 24, 3.**). Tästä listasta ruksitaan yhdistettävät kunnat ja kirjoitetaan uuden kunnan tunnus, valitaan suuralue, ympäristökeskus ja painetaan Liitä -nappia (**Kuva 24, 4.**).



Kuva 19: Kunnat 1



Kuva 20: Kunnat 2

Kuva 21: Tarkastajat -näyttö

5.1.7 Myrkyt-näyttö

Myrkyt-näytölle päästään tietyn pesän Pesätiedot-näytön linkiltä. Myrkyt-näytöllä voi katsoa, lisätä ja muuttaa tietyn pesän myrkkytietoja/vuosi (**Kuva 25**).

Kuva 25: Myrkyt-näyttö

5.1.8 Saaliit-näytöt

Saaliit_vanhat-näytölle (**Kuva 26**) päästään kuten Myrkyt- ja Historia-näytöillekin. Saaliit_vanhat-näytöllä voi tarkastella tietyn pesän saalistietoja/vuosi. Koska samalle pesälle voi tehdä useita saalistarkastuksia/vuosi, on saaliille oma ylläpitonäyttö. Saalit_vanhat sivun päivämäärälinkeistä pääsee muuttamaan yhden saalistarkastuksen tietoja, ja Uusi-linkistä pääsee lisäämään uuden tarkastuksen. Saaliit_yllapito-näytöllä voi lisätä ja muuttaa yksittäisen saalistarkastuksen tietoja (**Kuva 27)**.

Kuva 26: Saaliit_vanhat-näyttö

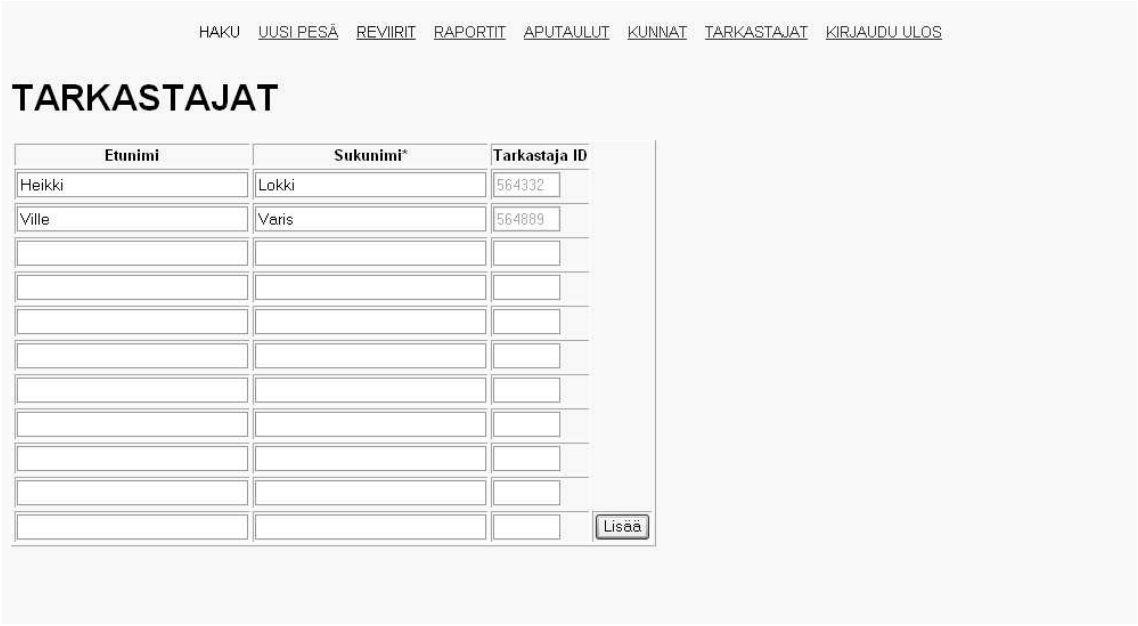
Kuva 27: Saaliit_yllapito-näyttö

5.1.9 Tarkastajat-näyttö

Tarkastajat-näytölle pääsee yläosan navigointipalkista. Sivulla voi lisätä uuden tarkastajan (**Kuva 28, 1.**), tai muuttaa jo olemassa olevien tarkastajien nimiä. Sivulle listautuu käyttäjän syöttämien hakuehtojen perusteella taulukko tietokantaan talletetuista tarkastajista (**Kuva 28, 2.**). Tarkastajan ID:tä ei ole mahdollista muuttaa.

5.1.10 Reviirit-näyttö

Reviirit-näytöllä voidaan lisätä järjestelmään uusi reviiri (**Kuva 29**)



Kuva 22: Tarkastajat -näyttö

Kuva 29: Reviirit-näyttö

6 Testaussuunnitelma

Tässä luvussa kuvataan ohjelmistolle tehtävän testauksen eri vaiheet, joita ovat komponenttitestaus (6.1), integrointitestaus (6.2) ja järjestelmätestaus (6.3). Itse testauksen tarkasta kuvauksesta ja testitapauksista tehdään erillinen dokumentti testivaiheen lopuksi. Testivaihe alkaa toteutusvaiheen loppupuolella.

Vaikka ohjelmointi pyritään tekemään mahdollisimman virheettömästi, virheitä ohjelmiin jää aina, jollei kyse ole aivan triviaalista ohjelmasta. Testauksen tarkoitus on havaita aikaisemmin tuntemattomat virheet. Täydellinen testaus on usein mahdotonta, mutta testauksella pyritään mahdollisimman pienellä työmäärällä löytämään mahdollisimman paljon virheitä. Testauksessa on tärkeää, että se on suunnitelmallista ja järjestelmällistä, ja samat testitapaukset ovat toistettavissa.

6.1 Komponenttitestaus

Komponenttitestauksessa ohjelmiston jokainen luokka ja metodi testataan erikseen. Testaus suoritetaan lausekattavasti. Tämä menetelmä on rakenteellinen, eli white-box mene-

telmä. Rakenteellisessa testauksessa testitapaukset tehdään ohjelmiston rakenteen perusteella. Rakenteelliseen testaukseen kuuluu myös pöytätestaus, eli ohjelman läpikäyminen paperilla. Pöytätestausta tehdään jo suunnittelun ja toteutuksen yhteydessä.

6.2 Integrointitestaus

Integrointitestauksen tarkoituksena on varmistaa eri luokkien yhteensopivuus. Kyseessä on black box-menetelmä, eli testausaineistona ovat ohjelman spesifikaation perusteella valitut syötteet. Syötteet jaetaan ekvivalenssiluokkiin siten, että samassa luokassa olevat syötteet todennäköisesti löytävät samat virheet. Näin testattavien syötteiden lukumäärä pysyy kohtuullisena. Syötteiden valinta on yleensä heuristista.

Ekvivalenssiluokkien lisäksi käytetään raja-arvoanalyysiä, jossa keskitytään rajatapauksiin. Virheet keskittyvät usein juuri syötteiden raja-tapauksiin. Lisäksi näiden systemaattisten menetelmien lisäksi kannattaa käyttää vielä epäformaalia menetelmää, arvaamista. Ainakin kirjoittajan oman kokemuksen perusteella monet isoimmista virheistä löytyvät usein epämääräisellä tavalla, jota voisi kutsua vaistoksi.

6.3 Järjestelmätestaus

Järjestelmätestauksessa testataan koko ohjelmisto. Testauksessa kiinnitetään huomiota mm. toiminnallisuuteen, käyttöliittymään, virhetilanteisiin, turvallisuuteen ja suorituskykyyn. Järjestelmätestauksessa otetaan huomioon vaatimusdokumentissa määritellyt ohjelmiston vaatimukset.

6.4 Testicaset

6.4.1 Käyttäjän tunnistus

Käyttöliittymän yleiskuva

Testattavat asiat	OK/EI?
Sisäänkirjautumislomake aukeaa sivulle saavuttaessa	
Tabulaattori käy läpi täytettävät kentät ja painikkeet järjestyksessä	
Lomakkeessa kaikki vaadittavat kentät	
Kaikki painikkeet löytyvät	

Toimintojen testaus

Testattavat asia	Alkutila	Syöte	Oletettu tulos
Yritetään kirjautua tyhjin kentin	Tyhjät kentät	Painetaan Sisään-painiketta	Virheilmoitus
Yritetään kirjautua väärällä tunnuksella	Väärä tunnus, oikea salasana	Painetaan Sisään-painiketta	Virheilmoitus
Yritetään kirjautua väärällä salasanalla	Oikea tunnus, väärä salasana	Painetaan Sisään-painiketta	Virheilmoitus
Yritetään kirjautua väärällä tunnuksella ja salasanalla	Väärä tunnus, väärä salasana	Painetaan Sisään-painiketta	Virheilmoitus
Yritetään kirjautua oikealla tunnuksella ja salasanalla	Oikea tunnus, oikea salasana	Painetaan Sisään-painiketta	Ilmoitus onnistuneesta sisäänkirjautumisesta ja siirtymä Haku-näytölle

6.4.2 Uuden pesän lisääminen

Käyttöliittymän yleiskuva

Testattavat asiat	OK/EI?
Uusi pesä-lomake aukeaa Uusi Pesä-toiminnon avulla	
Tabulaattori käy läpi täytettävät kentät ja painikkeet järjestyksessä	
Lomakkeessa kaikki vaadittavat kentät	
Kaikki painikkeet löytyvät	
Alasvetovalikoista löytyvät oikeat arvot	

Toimintojen testaus

Testattavat asia	Alkutila	Syöte	Oletettu tulos
Yritetään lisätä tyhjä lomake	Tyhjät kentät	Painetaan Lisää-painiketta	Virheilmoitus
Yritetään lisätä väärän muotoista dataa	Kentissä virhellisiä syötteitä	Painetaan Lisää-painiketta	Virheilmoitus
Yritetään lisätä oikean muotoista, mutta suuruusluokataan vääränlaista dataa	Kentissä virheellisen suuruisia syötteitä	Painetaan Lisää-painiketta	Virheilmoitus
Yritetään lisätä kelvollista dataa	Kentissä oikeanlaiset syötteet	Painetaan Lisää-painiketta	Ilmoitus onnistuneesta lisäyksestä

6.4.3 Vanhan pesän tarkastus

Käyttöliittymän yleiskuva

Testattavat asiat	OK/EI?
Lomake aukeaa Uusi tarkastus-toiminnon avulla	
Tabulaattori käy läpi täytettävät kentät ja painikkeet järjestyksessä	
Lomakkeessa kaikki vaadittavat kentät	
Kaikki painikkeet löytyvät	
Alasvetovalikoista löytyvät oikeat arvot	
Lomakkeen esitäytetyt arvot oikeita	

Toimintojen testaus

Testattavat asia	Alkutila	Syöte	Oletettu tulos
Yritetään lähettää tyhjä lomake	Tyhjät kentät	Painetaan Lähetä-painiketta	Virheilmoitus
Yritetään lähettää väärän muotoista dataa	Kentissä virhellisiä syötteitä	Painetaan Lähetä-painiketta	Virheilmoitus
Yritetään lähettää oikean muotoista, mutta suuruusluokataan vääränlaista dataa	Kentissä virheellisen suuruisia syötteitä	Painetaan Lähetä-painiketta	Virheilmoitus
Yritetään lähettää kelvollista dataa	Kentissä oikeanlaiset syötteet	Painetaan Lähetä-painiketta	Ilmoitus onnistuneesta tallennuksesta

6.4.4 Pesän hakeminen

Käyttöliittymän yleiskuva

Testattavat asiat	OK/EI?
Lomake aukeaa Haku-linkin avulla	
Tabulaattori käy läpi täytettävät kentät ja painikkeet järjestyksessä	
Lomakkeessa kaikki vaadittavat kentät	
Kenttien nimet on kirjoitettu oikein	
Kaikki painikkeet löytyvät	
Alasvetovalikoista löytyvät oikeat arvot	

Toimintojen testaus

Testattava asia	Alkutila	Syöte	Oletettu tulos
Haetaan tyhjin hakuehdoin	Tyhjät kentät	Painetaan Hae pesät-painiketta	Hakutulokseen ei ilmesty yhtään pesää
Haetaan väärin hakuehdoin	Kentissä väärän muotoisia syötteitä	Painetaan Hae Pesät-painiketta	Hakutulokseen ei ilmesty yhtään pesää
Haetaan oikeanlaisin ehdoin - 0 hakutulosta	Kenttiin on syötetty tietokantaa apuna käyttäen sellaiset arvot, että haun tulokseksi tulee 0 pesää	Painetaan Hae Pesät-painiketta	Hakutulokseen ei ilmesty yhtään pesää
Haetaan oikeanlaisin ehdoin - 1 hakutulos	Kenttiin on syötetty tietokantaa apuna käyttäen sellaiset arvot, että haun tulokseksi tulee 1 pesä	Painetaan Hae pesät-painiketta	Hakutulokseen ilmestyy täsmälleen yksi, oikea pesä
Haetaan oikeanlaisin ehdoin - monta hakutulosta	Kenttiin on syötetty tietokantaa apuna käyttäen sellaiset arvot, että haun tulokseksi tulee useita pesiä	Painetaan Hae Pesät-painiketta	Hakutulokseen ilmestyvät oikeat pesät (Laskeva aakkosjärjestys, samannimiset pesät järjestetty laskevaan järjetykseen vuoden mukaan)

6.4.5 Raporttien tuottaminen

Käyttöliittymän yleiskuva

Testattavat asiat	OK/EI?
Lomake aukeaa "Raportit-toiminnon avulla	
Tabulaattori käy läpi täytettävät kentät ja painikkeet järjestyksessä	
Lomakkeessa kaikki vaadittavat kentät	
Kenttien nimet kirjoitettu oikein	
Kaikki painikkeet löytyvät	
Alasvetovalikoista löytyvät oikeat arvot	
Ensimmäinen radionappivalikon raporteista on valittu	

Toimintojen testaus

Testattava asia	Alkutila	Syöte	Oletettu tulos
Yritetään tuottaa kukin raportti muiden lomakkeen kenttien ollessa tyhjiä	Yksi raportti valittu radionapin avulla, muutoin tyhjä lomake	Painetaan Tallenna tiedostoon-painiketta	Virhe vaiko eikö?
Yritetään tuottaa kukin raportti, kun ehtokentissä on vääränlaisia syötteitä	Yksi raportti valittu radionapin avulla, muissa kentissä vääränlaisia syötteitä	Painetaan Tallenna tiedostoon-painiketta	Raportti ilmestyy tiedostoon ja näytölle tulee ilmoitus raportin tuottamisesta
Yritetään tuottaa kukin raportti, kun ehtokenttien syötteet on valittu niin, ettei raporttiin voi sisällyttää yhtään pesää	Yksi raportti valittu radionapin avulla ja muihin kenttiin on syötetty tietokantaa apuna käyttäen sellaiset arvot, että raporttiin ei voi sisällyyttää yhtään pesää	Painetaan Tallenna tiedostoon-painiketta	Raportti ilmestyy tiedostoon ja näytölle tulee ilmoitus raportin tuottamisesta
Yritetään tuottaa kukin raportti, kun ehtokenttien syötteet on valittu niin, että raporttiin sisältyy vain yksi pesä, reviiri, kunta, ympäristökeskus tai suuralue	Yksi raportti on valittu radionapin avulla ja muihin kenttiin on syötetty tietokantaa apuna käyttäen sellaiset arvot, että raporttiin sisältyy yksi pesä, reviiri, kunta, ympäristökeskus tai suuralue	Painetaan Tallenna tiedostoon-painiketta	Raportti ilmestyy tiedostoon ja näytölle tulee ilmoitus raportin tuottamisesta
Yritetään tuottaa kukin raportti siten, että vain alkuvuosi on valittu, vain loppuvuosi on valittu, on valittu pidempi vuosiväli tai on valittu raportin tuottaminen yhdeltä vuoden ajalta	Yksi raportti on valittu radionapin avulla ja vain jälkimmäinen aikaväli-kentistä, ensimmäinen aikaväli-kentistä tai molemmat aikaväli-kentät on täytetty (ensimmäisessä pienempi vuosiluku kuin jälkimmäisessä tai molemmissa sama vuosiluku).	Painetaan Tallenna tiedostoon-painiketta	Raportti ilmestyy tiedostoon ja näytölle tulee ilmoitus raportin tuottamisesta

6.4.6 Aputaulun päivittäminen

Käyttöliittymän yleiskuva

Testattavat asiat	OK/EI?
Lomake aukeaa Aputaulut-toiminnon avulla	
Tabulaattori käy läpi täytettävät kentät ja painikkeet järjestyksessä	
Lomakkeessa kaikki vaadittavat kentät	
Kenttien nimet kirjoitettu oikein	
Kaikki painikkeet löytyvät	
Alasvetovalikoista löytyvät oikeat arvot	
Kun attribuutti on valittu, oikeat koodit ilmetyvät näytölle	

Toimintojen testaus

Testattava asia	Alkutila	Syöte	Oletettu tulos
Yritetään tuottaa kukin raportti muiden lomakkeen kenttien ollessa tyhjiä	Yksi raportti valittu radionapin avulla, muutoin tyhjä lomake	Painetaan Tallenna tiedostoon-painiketta	Virhe vaiko eikö?
Yritetään lisätä tietylle attribuutille tyhjä koodi ja selite	Taulu ja Attribuutti valittuina, lisättävät Koodi- ja Selite-kentät tyhjiä	Painetaan Lisää-painiketta	Virheilmoitus
Yritetään lisätä tietylle attribuutille jo käytössä oleva koodi	Taulu ja Attribuutti valittuina, lisättävässä Koodi-kentässä jo kyseisessä attribuutissa käytössä oleva koodi, lisättävässä Selite-kentässä jokin merkkijono	Painetaan Lisää-painiketta	Virheilmoitus
Yritetään lisätä tietylle attribuutille käyttämätön koodi selitteineen	Taulu ja Attribuutti valittuina, lisättävässä Koodi-kentässä uusi koodi kyseiselle attribuutille, Selite-kentässä jokin merkkijono	Painetaan Lisää-painiketta	Uusi koodi ja selite ilmestyvät ruudulle
Yritetään muuttaa tietyn attribuutin selitettä tyhjäksi	Taulu ja Attribuutti valittuina, muutettava Selite-kenttä tyhjä	Painetaan Muuta-painiketta	Virheilmoitus
Yritetään muuttaa tietyn attribuutin selitettä toiseksi	Taulu ja Attribuutti valittuina, muutettavassa Selite-kentässä jokin merkkijono	Painetaan Muuta-painiketta	Varmistus muutoksesta, ja Kyllä-toiminnon jälkeen muutoksen päivitys listaan

6.4.7 Pesän tietojen muuttaminen

Käyttöliittymän yleiskuva

Testattavat asiat	OK/EI?
Lomake pesätietojen muuttamiseen aukeaa Muuta-toiminnon avulla hakusivulta	
Tabulaattori käy läpi täytettävät kentät ja painikkeet järjestyksessä	
Lomakkeessa kaikki vaadittavat kentät	
Kenttien nimet on kirjoitettu oikein	
Kaikki painikkeet löytyvät	
Alasvetovalikoista löytyvät oikeat arvot	

Toimintojen testaus

Testattavat asia	Alkutila	Syöte	Oletettu tulos
Yritetään päivittää tyhjiksi lomakkeen pakollisia kenttiä	Pakolliset kentät tyhjiä	Painetaan Muuta-painiketta	Virheilmoitus
Yritetään päivittää lomakkeeseen vääränmuotoista dataa	Kentissä virhellisiä syötteitä	Painetaan Muuta-painiketta	Virheilmoitus
Yritetään päivittää lomakkeeseen oikeanmuotoista, mutta suuruusluokataan vääränlaista dataa	Kentissä virheellisen suuruisia syötteitä	Painetaan Muuta-painiketta	Virheilmoitus
Yritetään päivittää pesätietoja kelvollisin syötetin	Kentissä oikeanlaiset syötteet	Painetaan Lisää-painiketta	Ilmoitus onnistuneesta päivityksestä ja siirtymä Pesätiedot-näytölle

6.4.8 Reviiritietojen päivittäminen

Käyttöliittymän yleiskuva

Testattavat asiat	OK/EI?
Lomake aukeaa Reviirit-linkin avulla	
Tabulaattori käy läpi täytettävät kentät ja painikkeet järjestyksessä	
Lomakkeessa kaikki vaadittavat kentät	
Kenttien nimet on kirjoitettu oikein	
Kaikki painikkeet löytyvät	
Alavetovalikoista löytyvät oikeat arvot	

Toimintojen testaus

Testattavat asia	Alkutila	Syöte	Oletettu tulos
Yritetään lisätä uusi reviiri tyhjin kentin	Nimi-, Kunta- ja Kirjauspäivämäärä-kentät tyhjiä	Painetaan Lisää-painiketta	Virheilmoitus
Yritetään lisätä väärän muotoista dataa	Kentissä virheellisiä syötteitä	Painetaan Lisää-painiketta	Virheilmoitus
Yritetään lisätä lomakkeella oikeanmuotoista, mutta suuruusluokataan vääränlaista dataa	Kentissä virheellisen suuruisia syötteitä	Painetaan Lisää-painiketta	Virheilmoitus
Yritetään hakea reviiritietoja tyhjin hakuehdoin	Tyhjät kentät	Painetaan Hae-painiketta	Hakutulokseen ei ilmesty yhtään reviiriä
Yritetään hakea reviiritietoja vääränmuotoisin hakuehdoin	Kentissä virheellisiä syötteet	Painetaan Hae-painiketta	Hakutulokseen ei ilmesty yhtään reviiriä
Yritetään hakea reviiritietoja oikeanlaisin hakuehdoin - 0 tulosta	Kenttiin on syötetty tietokantaa apuna käyttäen sellaiset arvot, että haun tulokseksi tulee 0 reviiriä	Painetaan Hae-painiketta	Hakutulokseen ei ilmesty yhtään reviiriä
Yritetään hakea reviiritietoja oikeanlaisin hakuehdoin - 1 tulos	Kenttiin on syötetty tietokantaa apuna käyttäen sellaiset arvot, että haun tulokseksi tulee tasan 1 reviiri	Painetaan Hae-painiketta	Hakutulokseen ilmestyy tasan 1 reviiri
Yritetään hakea reviiritietoja oikeanlaisin hakuehdoin - monta tulosta	Kenttiin on syötetty tietokantaa apuna käyttäen sellaiset arvot, että haun tulokseksi saadaan useampi reviiri	Painetaan Hae-painiketta	Hakutulokseen ilmestyy monta reviiriä
Haetaan reviiritietoja kullakin hakuehdolla yksitellen ja kaikilla ehtojen kombinaatioilla käyttäen oikeanlaisia hakuehtoja	Kombinaatiota vastaavat hakuehtokentät täytetty, muut hakuehdokentät tyhjiä	Painetaan Hae-painiketta	Hakutulokseen ilmestyvät hakuehtoja vastaavat reviirit
Yritetään päivittää reviiritietoja tyhjin kentin	Nimi-, Kunta- ja Kirjauspäivämäärä-kentät tyhjiä	Painetaan Muuta-painiketta	Virheilmoitus
Yritetään päivittää reviiritietoihin vääränmuotoista dataa	Kentissä virheellisiä syötteitä	Painetaan Muuta-painiketta	Virheilmoitus
Yritetään päivittää reviiritietoihin oikeanmuotoista, mutta suuruusluokaltaan vääränlaista dataa	Kentissä virheellisen suuruisia syötteitä	Painetaan Muuta-painiketta	Virheilmoitus
Yritetään päivittää reviiritietoihin oikeanlaista dataa	Kentissä oikeanlaisia syötteitä	Painetaan Muuta-painiketta	Varmistus muutoksesta ja Kyllä-toiminnon jälkeen päivitys listaan

Lähteet

CVS03 CVS, Cvs - concurrent versions system, 2003. `http://www.cs.helsinki.fi/group/oukki/doc/ohjeita/cvs-manual/`

Fre03 Freemarker, Freemarker 2.x, 2003. `http://www.freemarker.org`

fsf03 fsf, Gnu general public license, version 2, june 1991, 2003. http://www.fsf.org/licenses/gpl.html

Gam95 Gamma, E., *Design Patterns*. Addison-Wesley Pub Co., 1995.

Hal03a Hali, O., Ohjelmistotuotantoprojekti, 2003. http://www.cs.helsinki.fi/group/hali

Hal03b Hali, O., Projektisuunnitelma, 2003. http://www.cs.helsinki.fi/group/hali/dokumentit/projektisuunnitelma.pdf

Hal03c Hali, O., Suunnitteludokumentti, 2003. http://www.cs.helsinki.fi/group/hali/dokumentit/Suunnittelu_v_1_5.pdf

Hal03d Hali, O., Vaatimusdokumentti, 2003. http://www.cs.helsinki.fi/group/hali/dokumentit/Vaatimus_v_1_7.pdf

Kot04a Kotkat, O., Ohjelmistotuotantoprojekti, 2004. http://www.cs.helsinki.fi/group/kotkat

Kot04b Kotkat, O., Vaatimusdokumentti, 2004. http://www.cs.helsinki.fi/group/kotkat/dokumentit/vaatimus_pe_27_2.pdf

Mer03 Merikotka, S. W., Suomen wwf - merikotka, 2003. http://www.wwf.fi/kotimaisetlajit/merikotka.html

Mic03a Microsoft, Internet explorer home page, 2003. http://www.microsoft.com/windows/ie/default.asp

Mic03b Microsystems, S., Code conventions for the java^{TM} programming language, 1999, 2003. http://java.sun.com/docs/codeconv/

Mic03c Microsystems, S., Javadoc tool home page, 2003. http://java.sun.com/j2se/javadoc/

Mic03d Microsystems, S., Java^{TM} 2 platform, standard edition (j2se^{TM})., 2003. http://java.sun.com/j2se/1.4/

Mic03e Microsystems, S., Java^{TM} servlet technology, 2003. http://java.sun.com/products/servlet/

Mic03f Microsystems, S., Jdbc^{TM} technology, 2003. http://java.sun.com/products/jdbc/

Mic04 Microsystems, S., Patterns, 2004. http://java.sun.com/blueprints/patterns/

Ora03 Oracle, Oracle9i database, 2003. http://www.oracle.com/ip/deploy/database/oracle9i/

Pan03 Pandion, O., Pandion, 2003. http://www.cs.helsinki.fi/group/pandion

Pro03 Project, T. A., The apache jserv project, 2003. http://java.apache.org/jserv/

Soc03 Society, T. I., Hypertext transfer protocol - http/1.1, 1999, 2003. http://www.cs.helsinki.fi/group/pandion

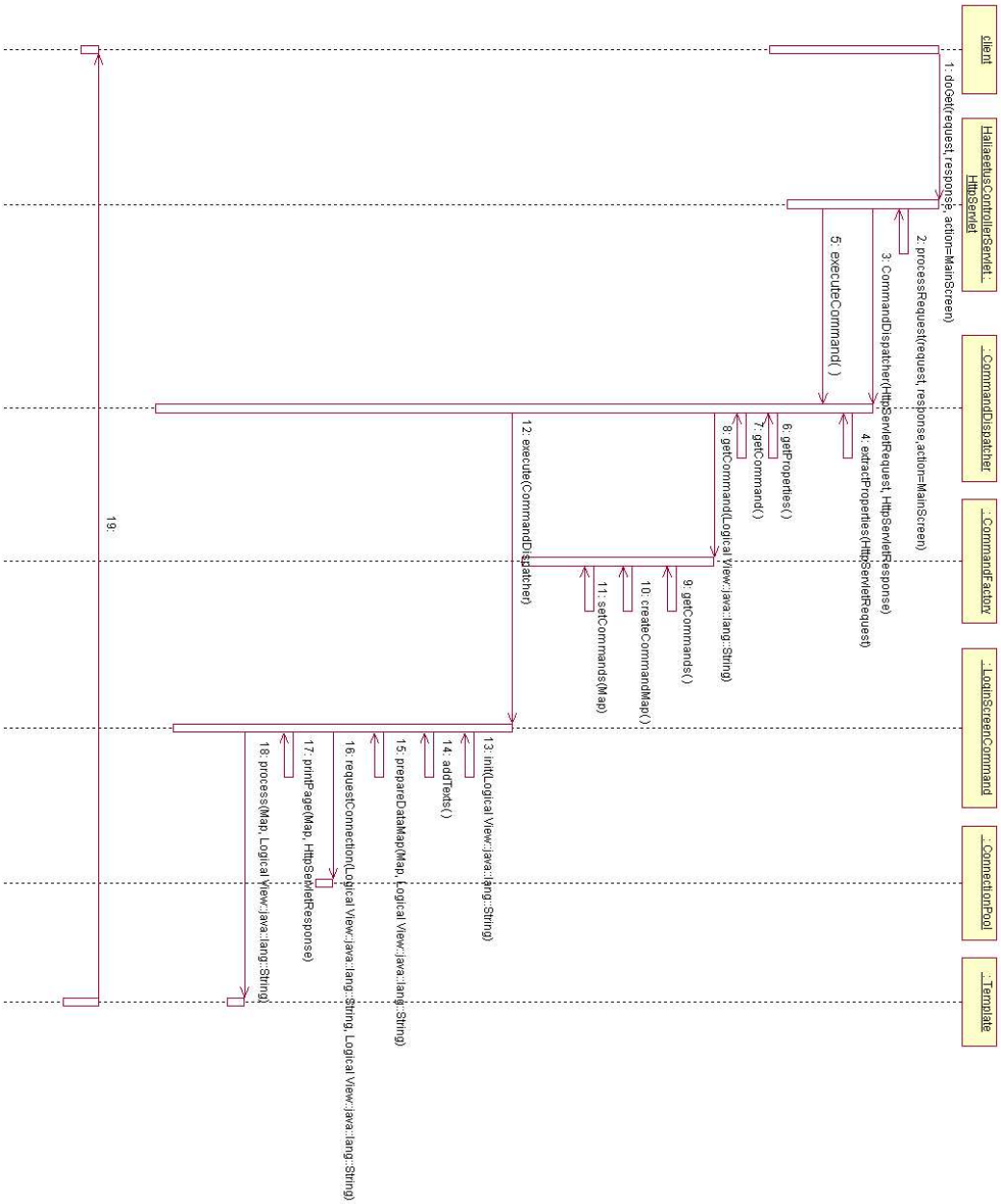
Tip03 Tipu4, O., Ohjelmistotuotantoryhmä, 2003. http://www.cs.helsinki.fi/group/tipu4/

W3C03a W3C, Html 4.01 specification, 1999, 2003. http://www.w3.org/TR/html401/

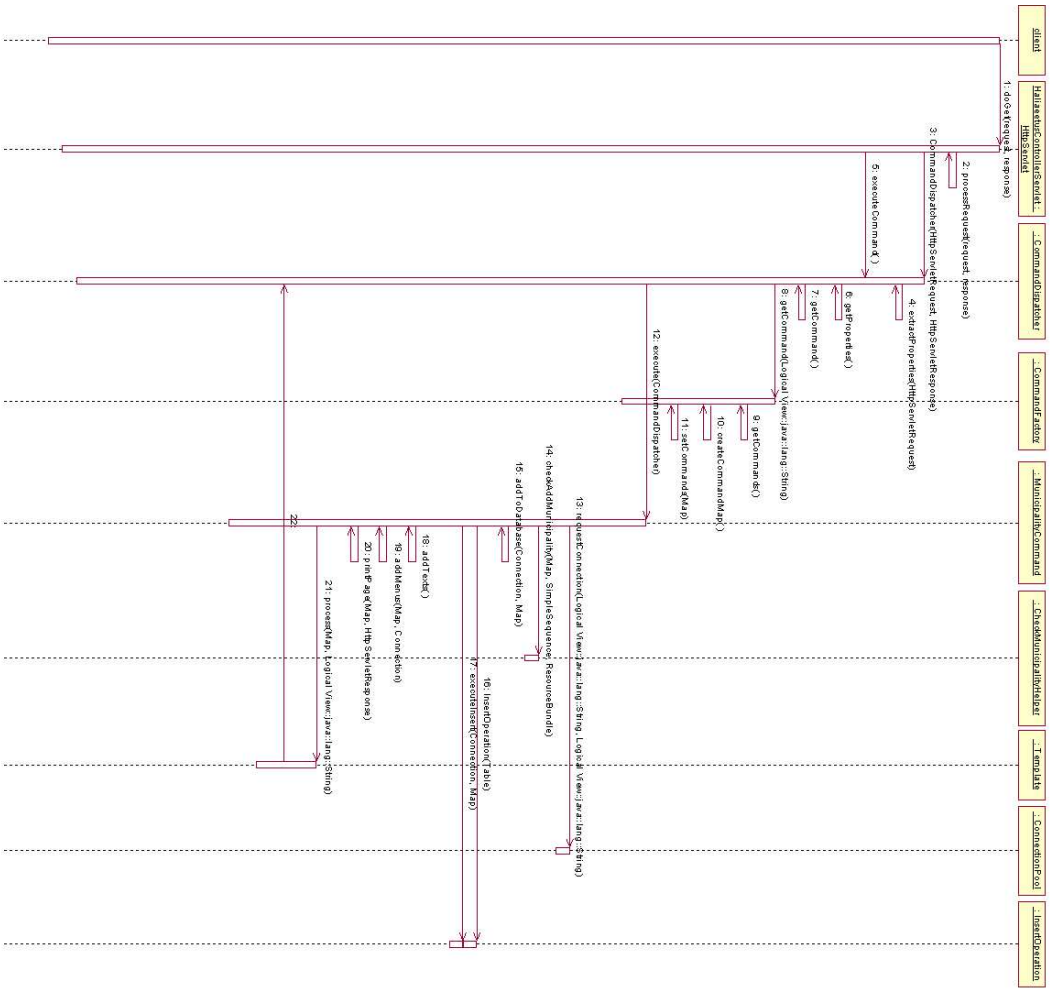
W3C03b W3C, The world wide web consortium, 2003. http://www.w3.org/

WWF04 WWF, S., Suomen wwf, 2004. http://www.wwf.fi

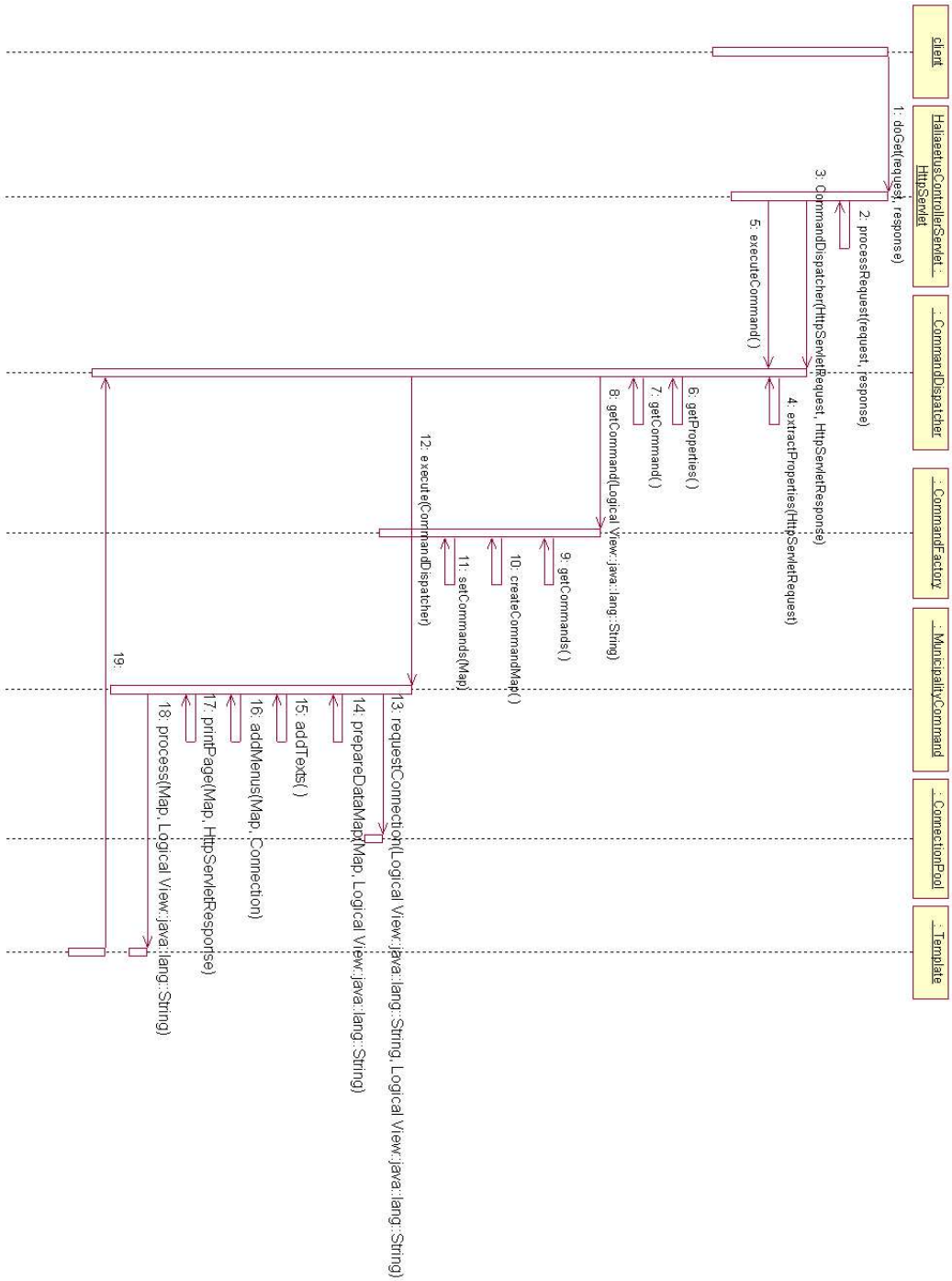
A Sekvenssikaaviot



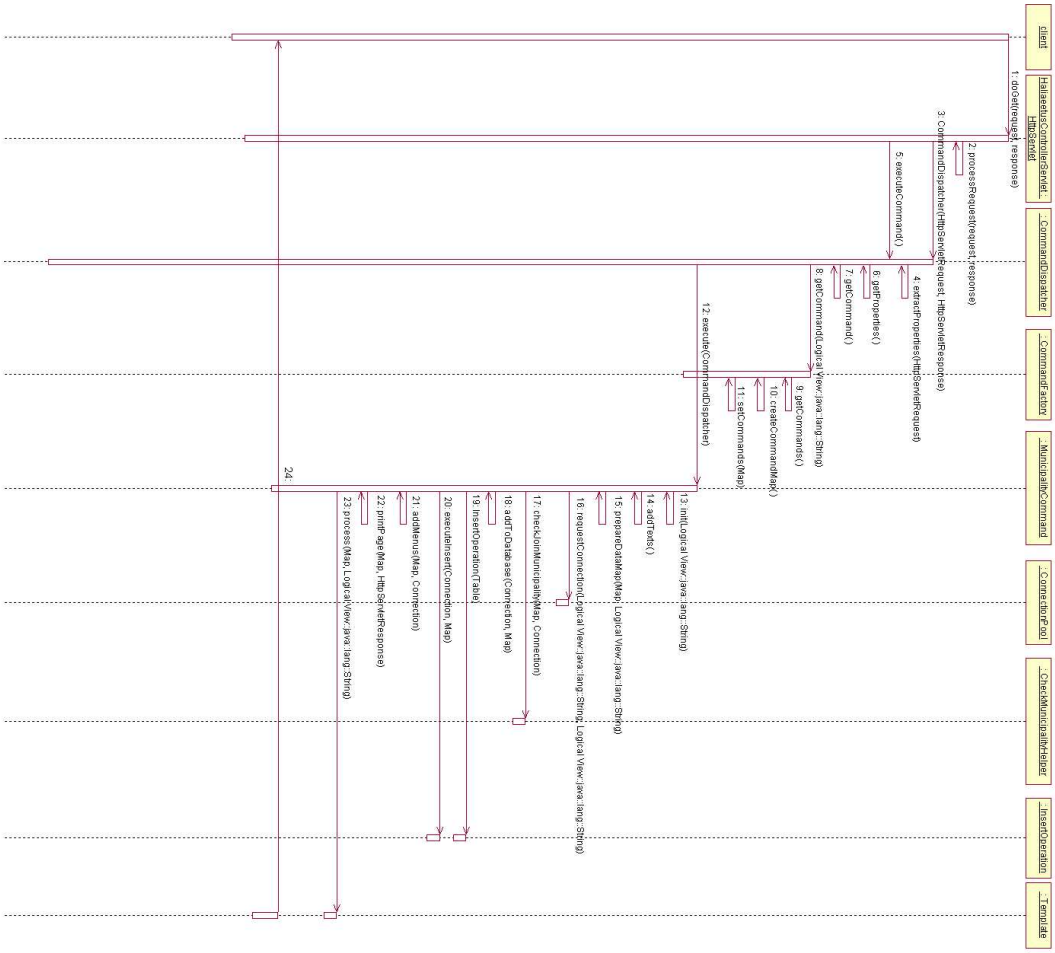
Kuva 23: Järjestelmään sisäänkirjautuminen



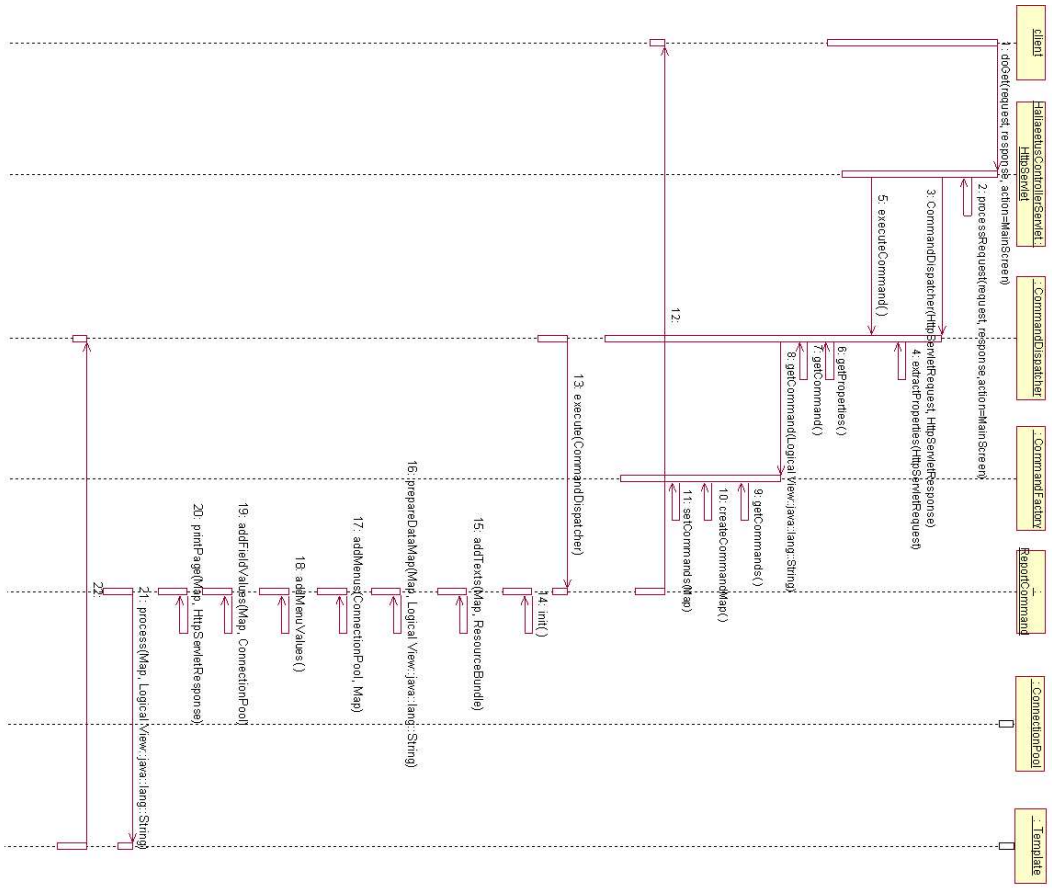
Kuva 24: Uuden kunnan lisääminen



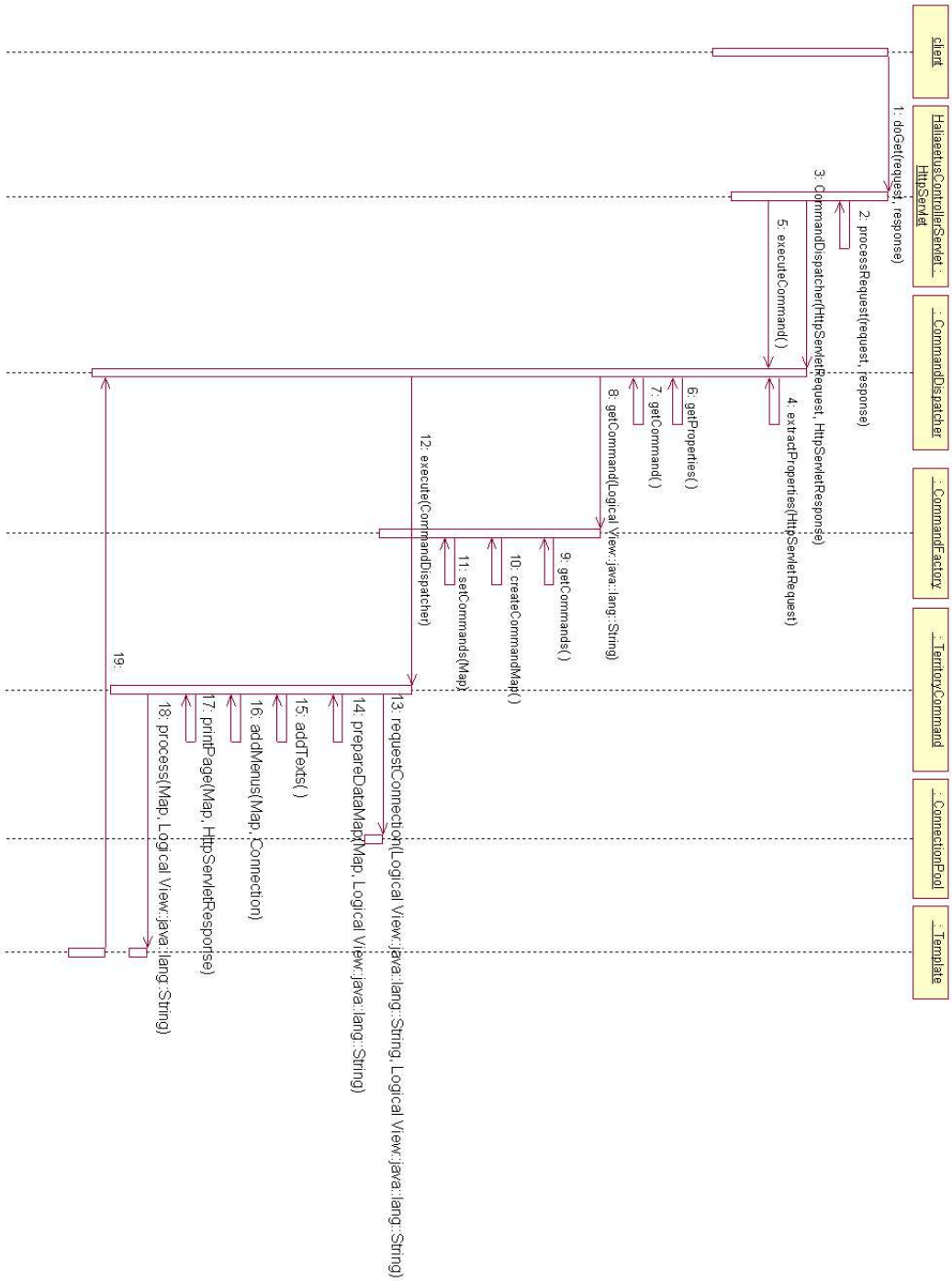
Kuva 25: Kunta-näytön generointi



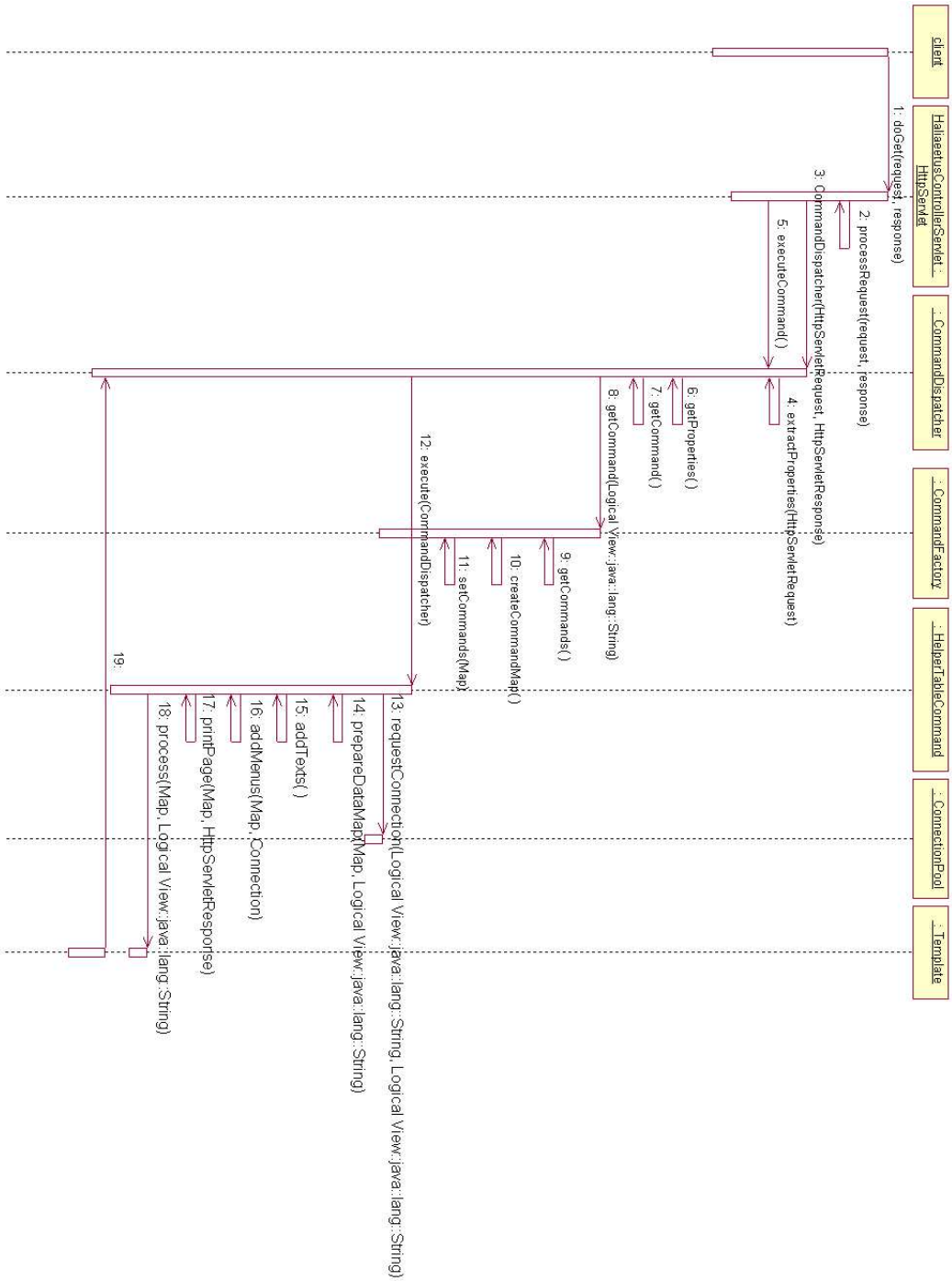
Kuva 26: Kuntaliitoksen tekeminen



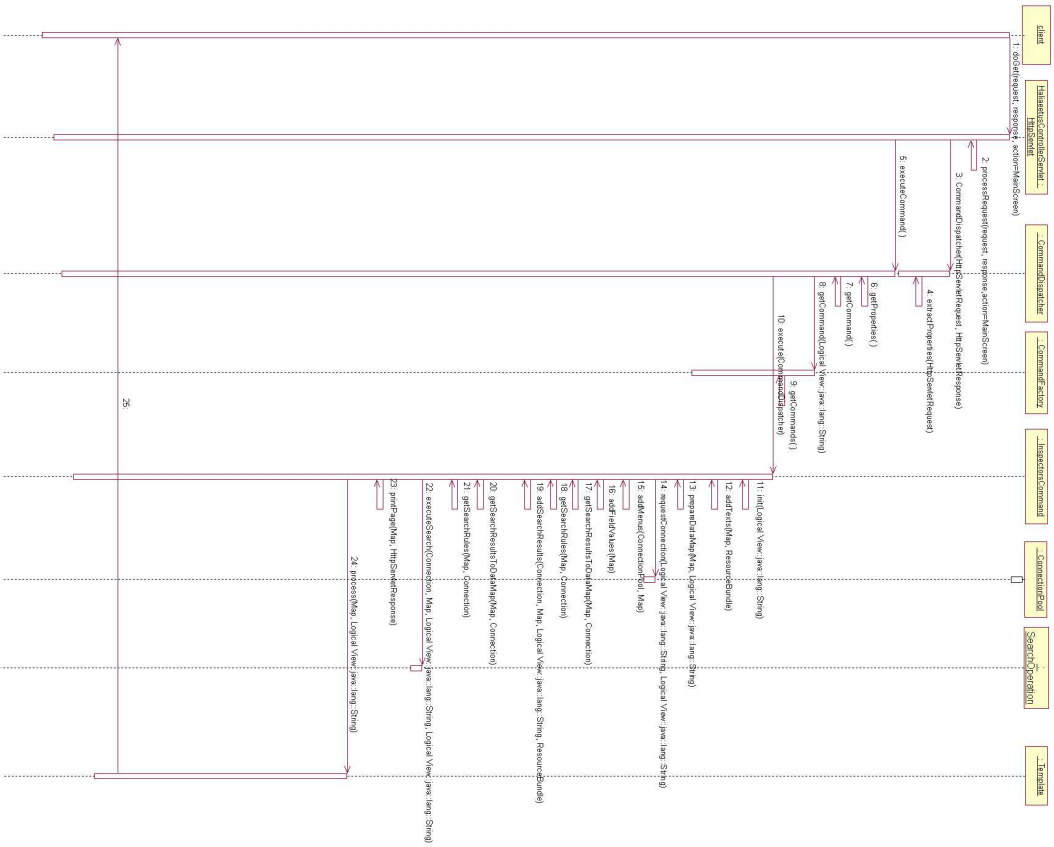
Kuva 27: Raportti-näytön generointi



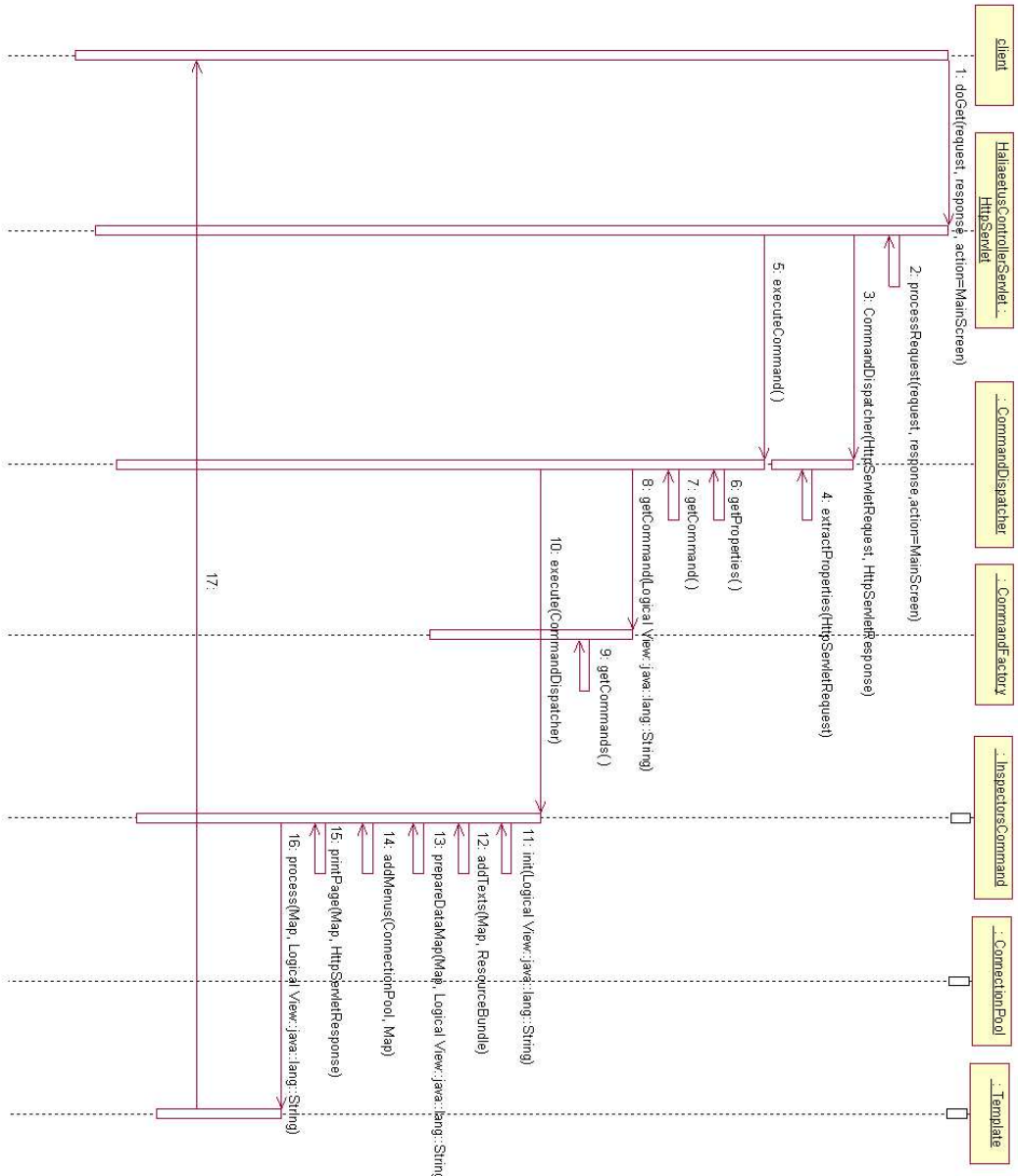
Kuva 28: Reviiri-näytön generointi



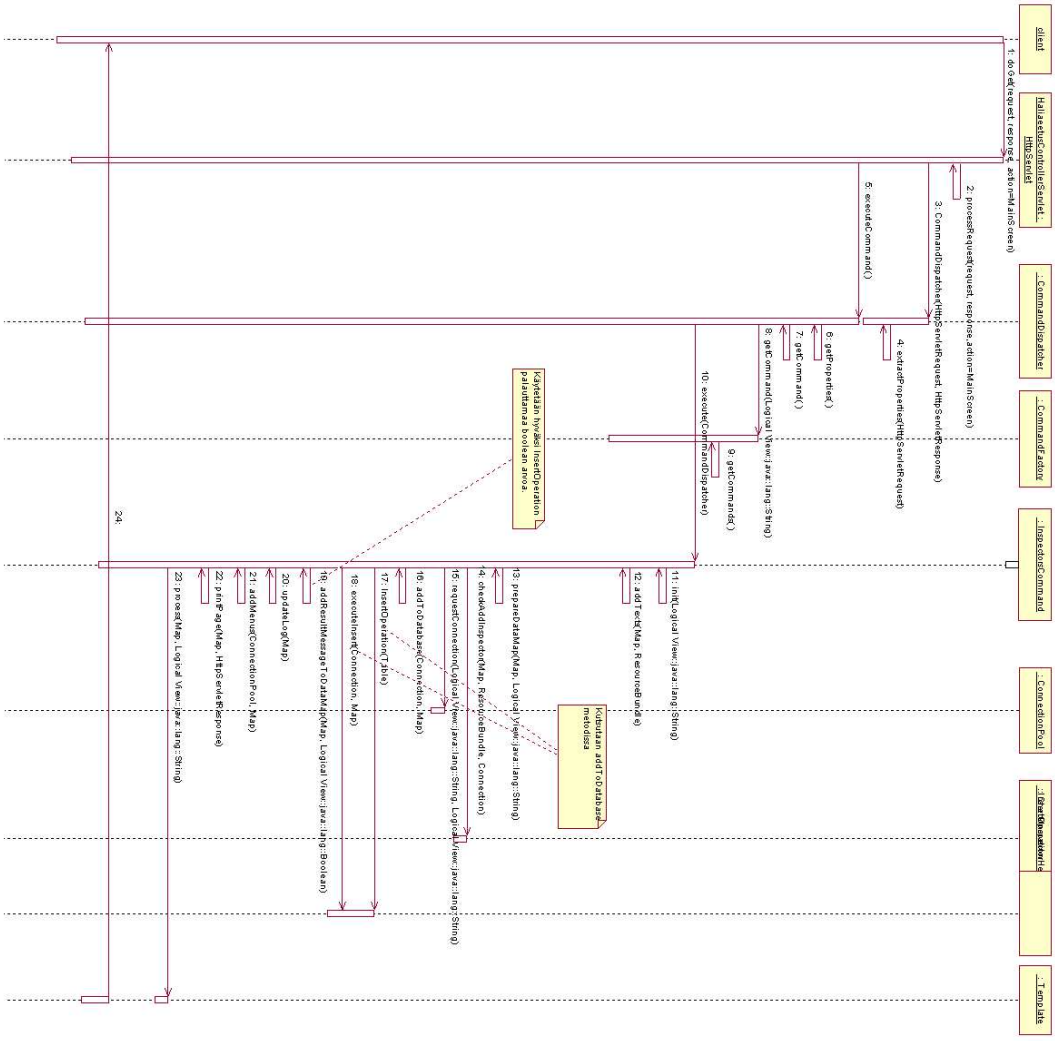
Kuva 29: Aputaulu-näytön generointi



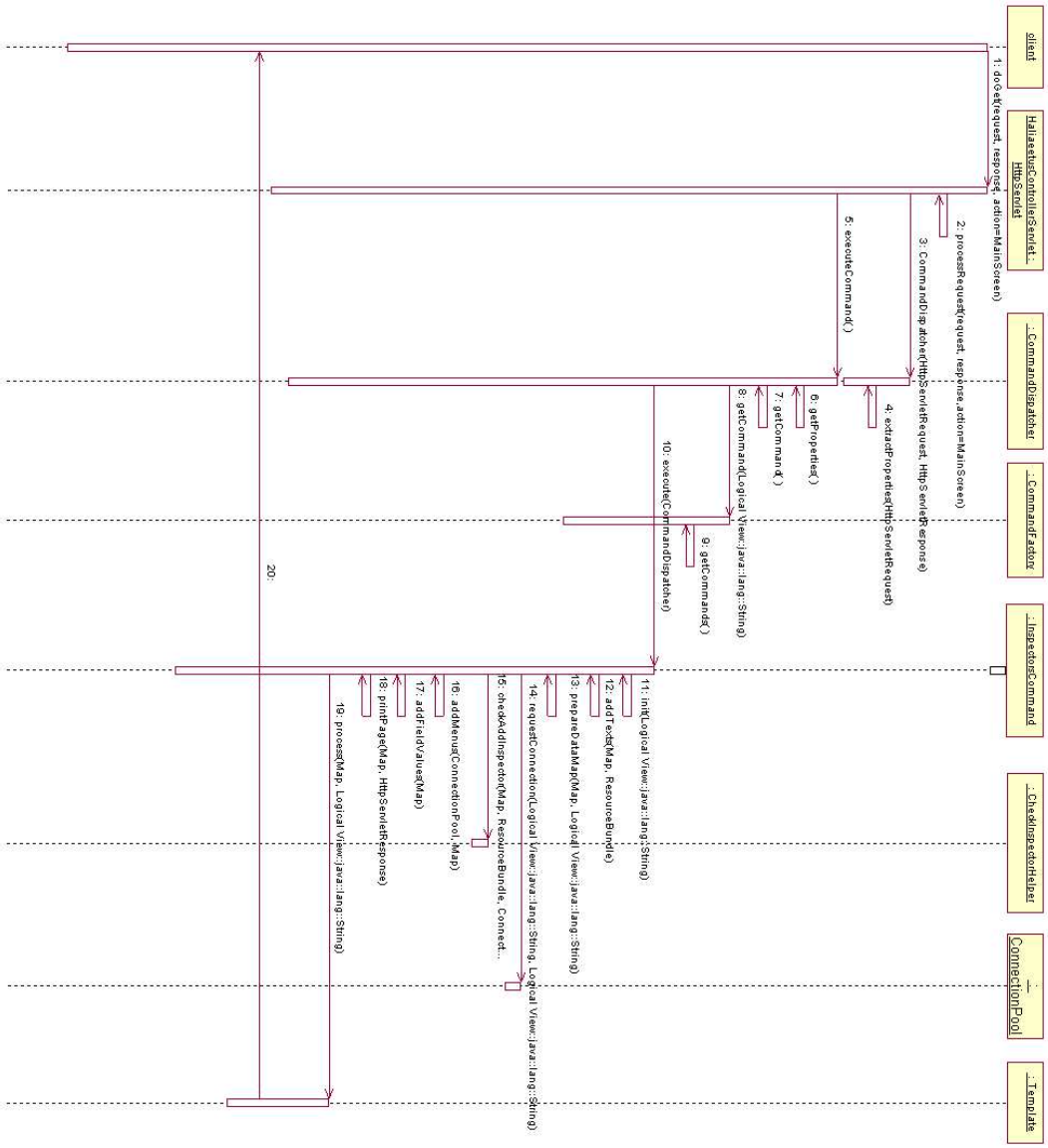
Kuva 30: Tarkastajan haku



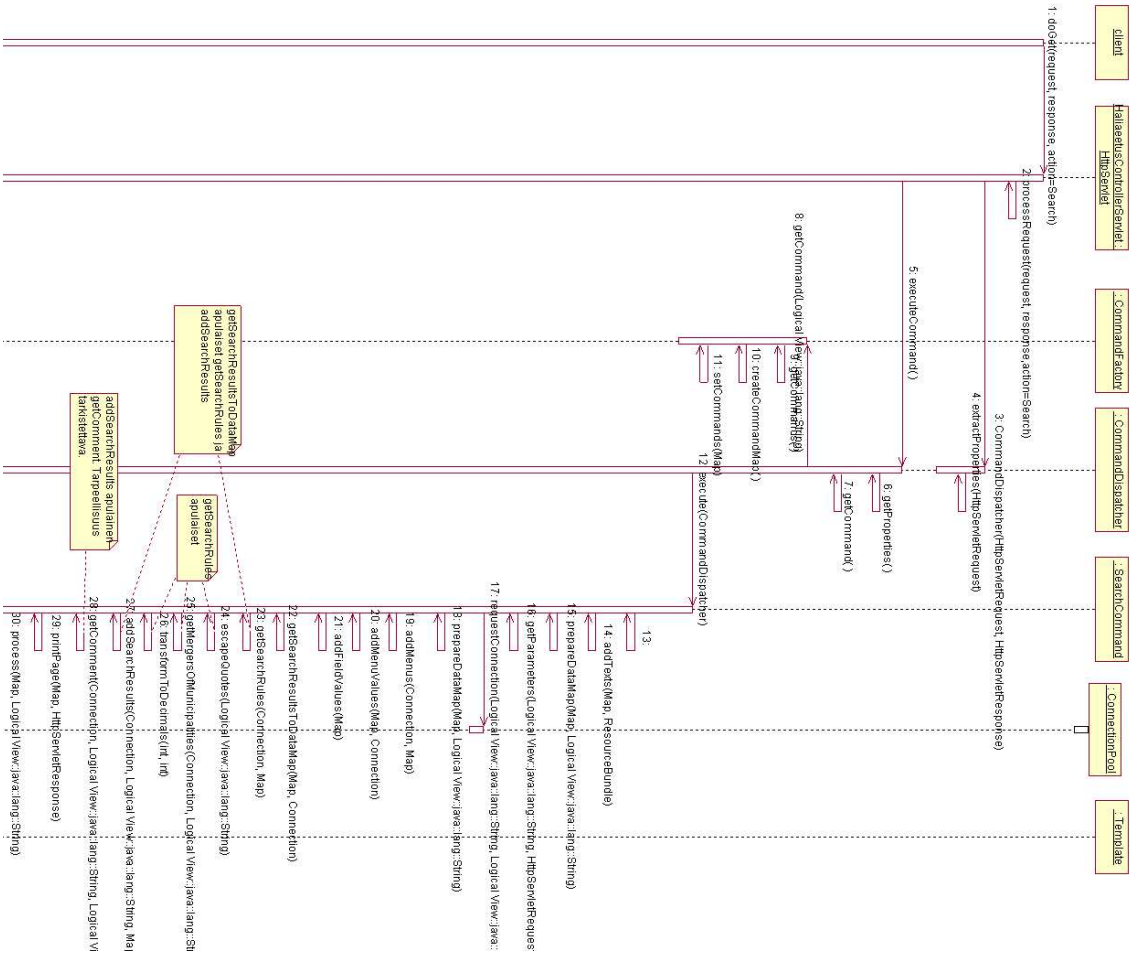
Kuva 31: Tarkastaja-näytön generointi



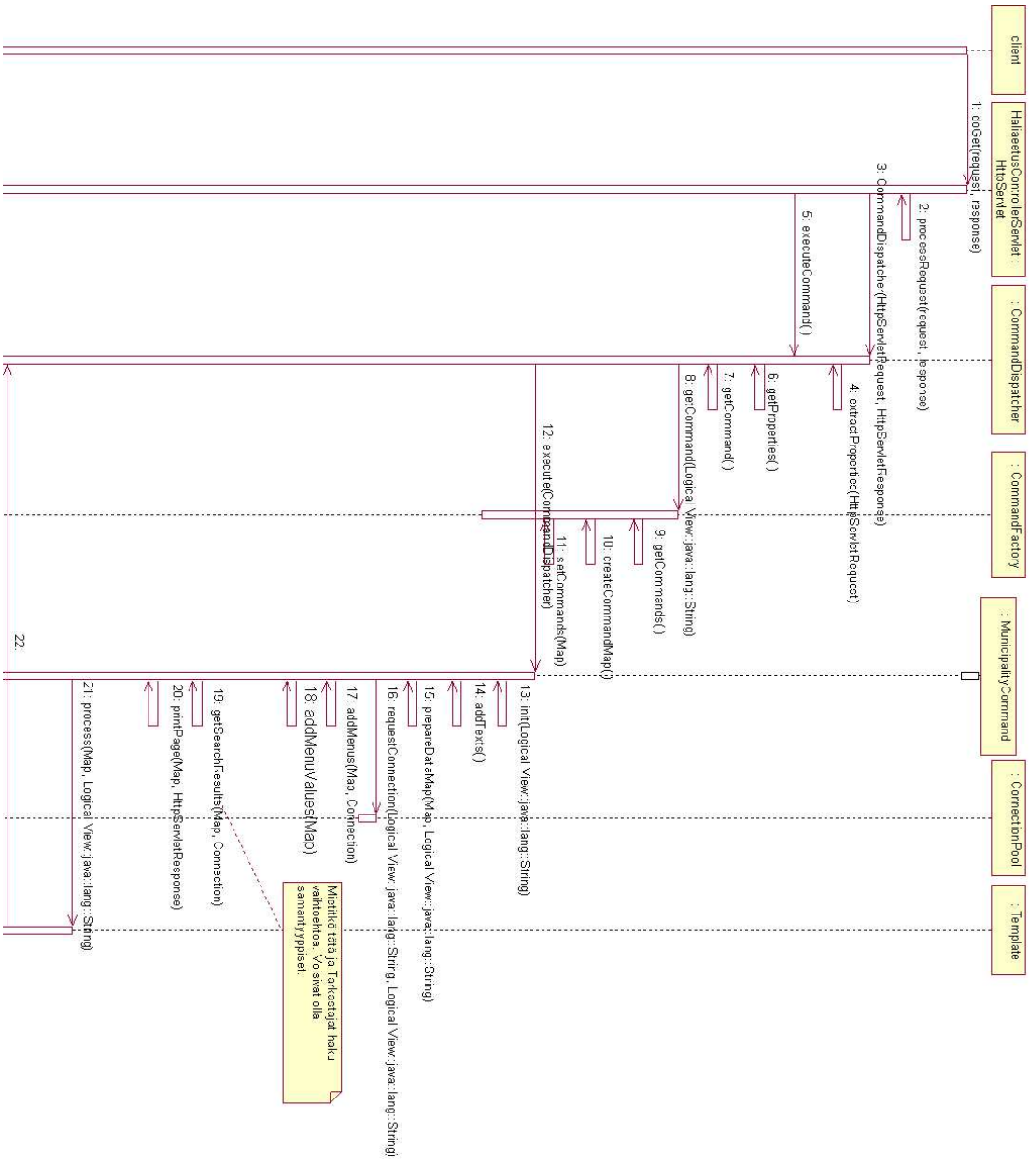
Kuva 32: Tarkastajan onnistunut lisäys



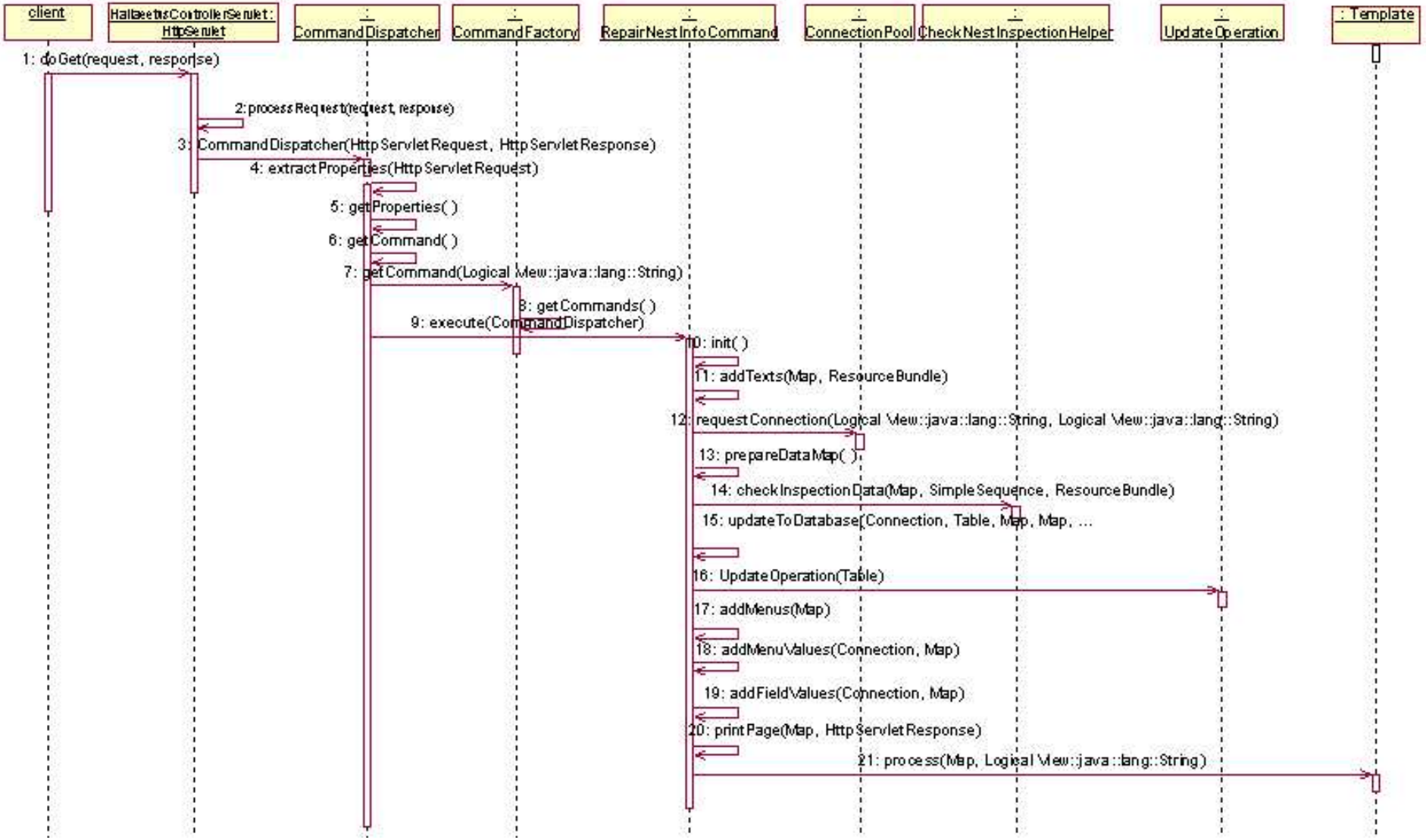
Kuva 33: Tarkastajan epäonnistunut lisäys

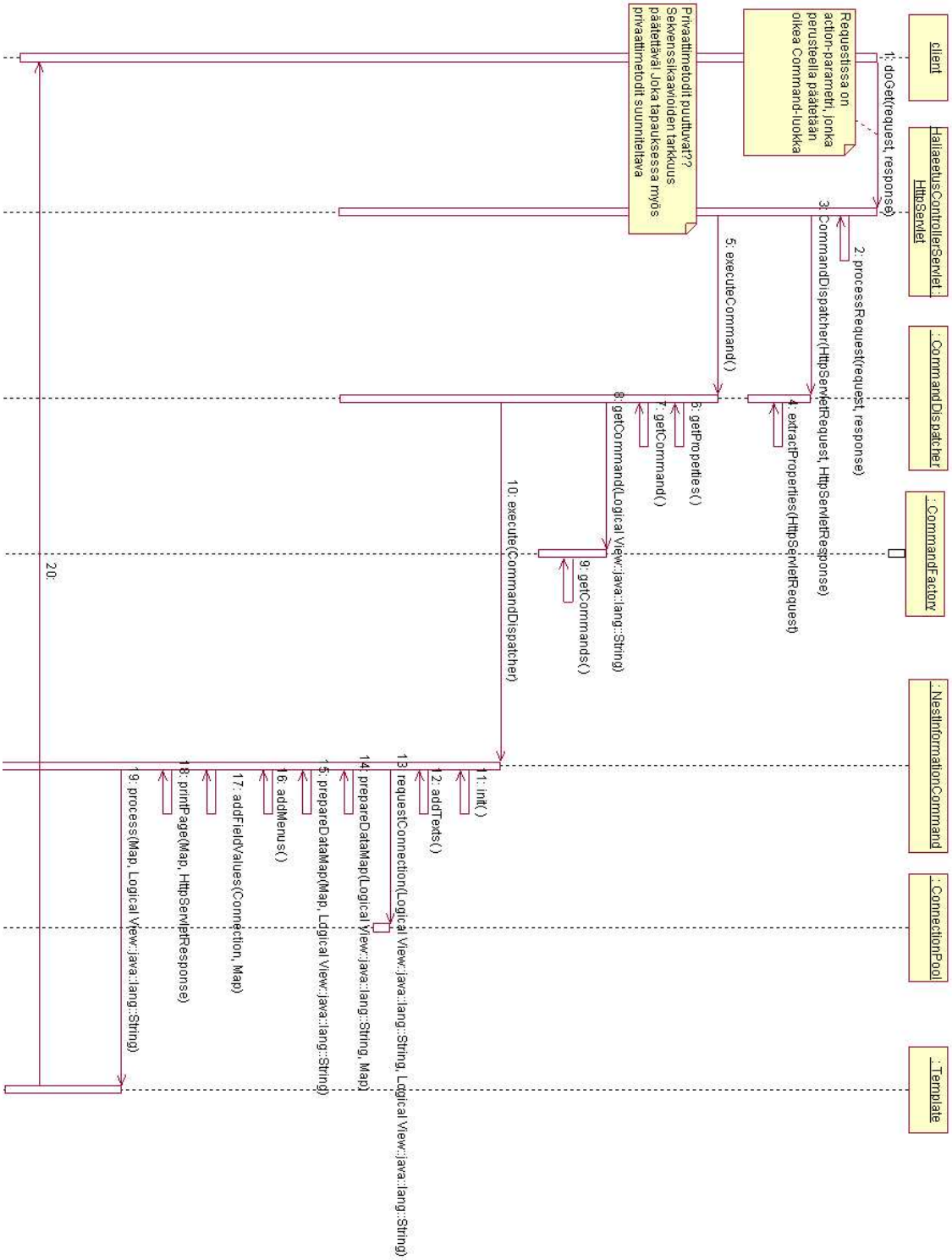


Kuva 34: Haku-näytön generointi

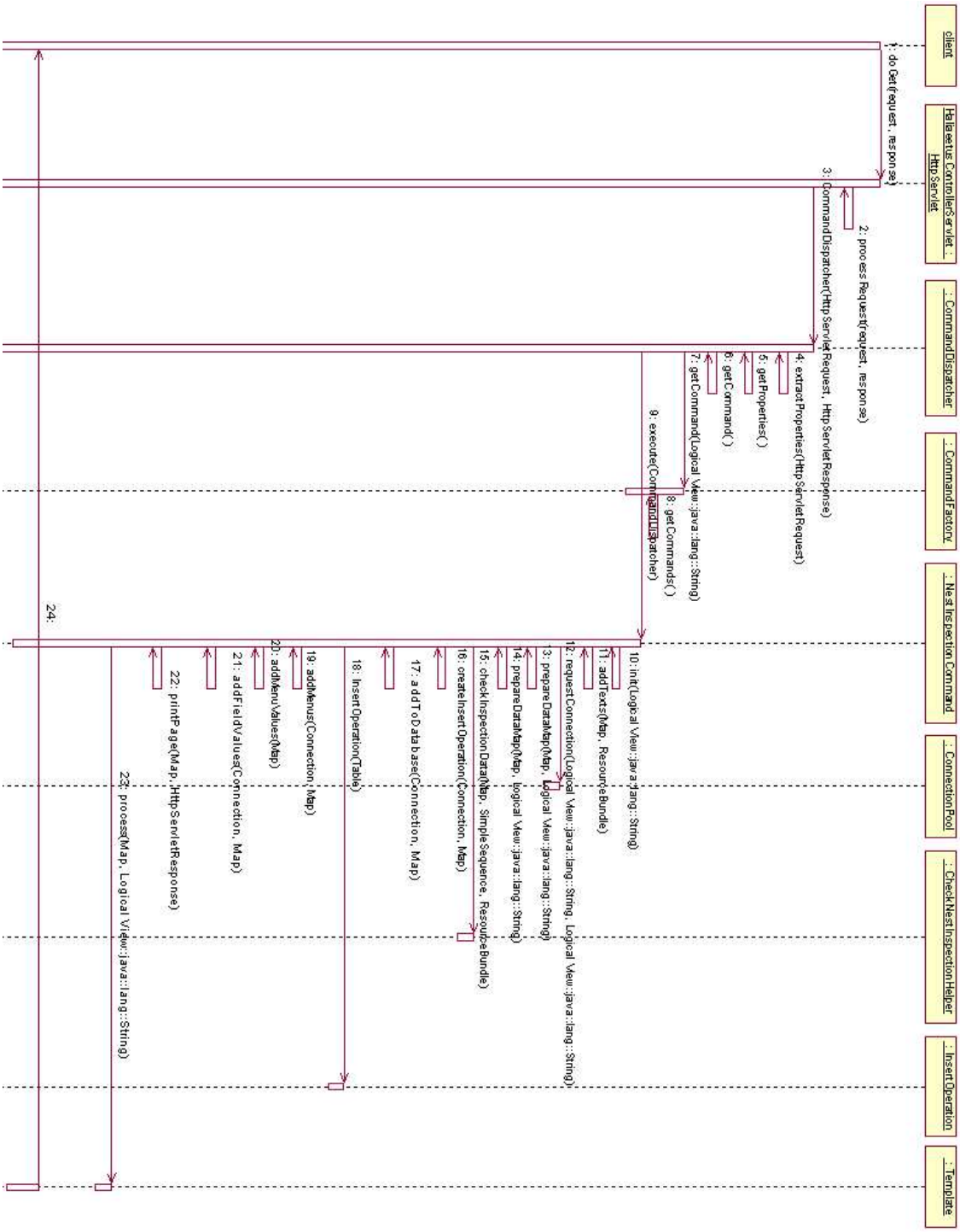


Kuva 35: Kuntien haku


client
HaltaeetusControllerServlet : HttpServlet
: CommandDispatcher
: CommandFactory
: RepairNestInfoCommand
: ConnectionPool
: CheckNestInspectionHelper
: UpdateOperation
: Template
1: doGet(request, response)
2: processRequest(request, response)
3: CommandDispatcher(HttpServletRequest, HttpServletResponse)
4: extractProperties(HttpServletRequest)
5: getProperties( )
6: getCommand( )
7: getCommand(Logical View::java::lang::String)
8: getCommands( )
9: execute(CommandDispatcher)
10: init( )
11: addTexts(Map, ResourceBundle)
12: requestConnection(Logical View::java::lang::String, Logical View::java::lang::String)
13: prepareDataMap( )
14: checkInspectionData(Map, SimpleSequence, ResourceBundle)
15: updateToDatabase(Connection, Table, Map, Map, ...
16: UpdateOperation(Table)
17: addMenus(Map)
18: addMenuValues(Connection, Map)
19: addFieldValues(Connection, Map)
20: printPage(Map, HttpServletResponse)
21: process(Map, Logical View::java::lang::String)



Kuva 37: Pesän tiedot



Kuva 38: Uusi pesä

B navi.ftl

```
|
| + kieli
|     - fi
|
| + teksti
|     |
|     | + navi_raportit
|     |     - Raportit
|     |
|     | + navi_kunnat
|     |     - Kunnat
|     |
|     | + navi_laji
|     |     - Laji
|     |
|     | + navi_mainscreen
|     |     - Haku
|     |
|     | + navi_ulos
|     |     - Kirjaudu ulos
|     |
|     | + navi_aputaulut
|     |     - Aputaulut
|     |
|     | + navi_tarkastajat
|     |     - Tarkastajat
|     |
|     | + navi_reviirit
|     |     - Reviirit
|     |
|     | + navi_uusipesa
|     |     - Uusi Pesä
|
| + kutsuja
|     - mainscreen
```

C aputaulut.ftl

```
|
| + tulos
|     |
|     | + aputaulut_selite
|     |     - Selite
|     |
|     | + aputaulut_koodi
|     |     - Koodi
|     |
|     | + aputaulut_muutatietoja
|     |     - Muuta Selitettd
|     |
|     | + olemassa
|     |     - false
|     |
|     | + aputaulut_lisaa_koodi
|     |     - Lisdd uusi koodi
|
| + ilmoitus
|     |
|     | + syy
|     |     - freemarker.template.SimpleSequence@24eafa
|     |
|     | + olemassa
|     |     - false
|
| + teksti
|     |
|     | + aputaulut_haku
|     |     - Hae
|     |
|     | + aputaulut_taulu
|     |     - Taulut
|     |
|     | + aputaulut_hae_attribuutti
|     |     - Hae Koodit
|     |
|     | + aputaulut_attribuutti
|     |     - Taulun Attribuutit
|     |
|     | + aputaulut_haetut_rivit
|     |     - Haetut Koodit
|     |
|     | + aputaulut_otsikko
|     |     - Aputaulut
|     |
|     | + aputaulut_hae_taulu
|     |     - Hae attribuutit
|     |
|     | + aputaulut_h1
|     |     - Aputaulut
|     |
|     | + static_url
|     |     - http://localhost:8080\Haliaeetus\HaliaeetusController
|
| + ilmoitus_lisays
|     |
|     | + syy
|     |     - freemarker.template.SimpleSequence@1bef987
|     |
|     | + olemassa
|     |     - false
|
| + navi
|     |
|     | + kieli
|     |     - fi
|     |
|     | + teksti
|     |     |
|     |     | + navi_raportit
|     |     |     - Raportit
```

```
|     |     |
|     |     | + navi_kunnat
|     |     |     - Kunnat
|     |     |
|     |     | + navi_laji
|     |     |     - Laji
|     |     |
|     |     | + navi_mainscreen
|     |     |     - Haku
|     |     |
|     |     | + navi_ulos
|     |     |     - Kirjaudu ulos
|     |     |
|     |     | + navi_aputaulut
|     |     |     - Aputaulut
|     |     |
|     |     | + navi_tarkastajat
|     |     |     - Tarkastajat
|     |     |
|     |     | + navi_reviirit
|     |     |     - Reviirit
|     |     |
|     |     | + navi_uusipesa
|     |     |     - Uusi Pesd
|     |
|     | + kutsuja
|     |     - helpertable
|
| + muuta
|     |
|     | + selite
|     |     -
|     |
|     | + koodi
|     |     -
|
| + arvo
|     |
|     | + taulut
|     |     -
|     |
|     | + attribuutti
|     |     -
|
| + syy_otsikko_lisays
|     |
|     | + olemassa
|     |     - false
|
| + lisaa
|     |
|     | + selite
|     |     -
|     |
|     | + koodi
|     |     -
|
| + taulu
|     |
|     | + haettu
|     |     - false
```

D mainscreen.ftl

```
|
| + tulos
|     |
|     | + haku_tulos_vuosi
|     |     - Vuosi
|     |
|     | + haku_tarkastus
|     |     - Tarkastus
|     |
|     | + haku_tulos_reviiri
|     |     - Reviiri
|     |
|     | + olemassa
|     |     - false
|     |
|     | + haku_tulos_nimi
|     |     - Nimi
|     |
|     | + haku_tulos_id
|     |     - Id
|     |
|     | + haku_tulos_kunta
|     |     - Kunta
|     |
|     | + haku_muutatietoja
|     |     - Muuta tietoja
|     |
|     | + haku_vuodelle
|     |     - vuodelle
|
| + ilmoitus
|     |
|     | + syy
|     |     - freemarker.template.SimpleSequence@40627c
|     |
|     | + olemassa
|     |     - false
|
| + teksti
|     |
|     | + haku_reviiri
|     |     - Reviiri
|     |
|     | + haku_leveys
|     |     - Leveys
|     |
|     | + haku_h1
|     |     - Haku
|     |
|     | + haku_hakuehdot
|     |     - Hakuehdot
|     |
|     | + haku_aste
|     |     - Astekoordinaatisto
|     |
|     | + haku_ymp_keskus
|     |     - Ympdristvkeskus
|     |
|     | + static_url
|     |     - http://localhost:8080\Haliaeetus\HaliaeetusController
|     |
|     | + haku_pesa
|     |     - Pesd
|     |
|     | + haku_koordinaatit
|     |     - Koordinaatit
|     |
|     | + haku_yhtenais
|     |     - Yhtendiskoordinaatisto
|     |
|     | + haku_pituus
|     |     - Pituus
```

```
|   |
|   | + haku_id
|   |   - id
|   |
|   | + haku_hae_pesat
|   |   - Hae Pesdt
|   |
|   | + haku_kunta
|   |   - kunta
|   |
|   | + haku_otsikko
|   |   - Haliaeetus - Haku
|   |
|   | + haku_nimi
|   |   - nimi
|   |
|   | + haku_vuosi
|   |   - Vuosivdli
|   |
|   | + haku_haetut_pesat
|   |   - Haetut Pesdt
|   |
|   | + haku_suuralue
|   |   - Suuralue
|
| + navi
|   |
|   | + kieli
|   |   - fi
|   |
|   | + teksti
|   |   |
|   |   | + navi_raportit
|   |   |   - Raportit
|   |   |
|   |   | + navi_kunnat
|   |   |   - Kunnat
|   |   |
|   |   | + navi_laji
|   |   |   - Laji
|   |   |
|   |   | + navi_mainscreen
|   |   |   - Haku
|   |   |
|   |   | + navi_ulos
|   |   |   - Kirjaudu ulos
|   |   |
|   |   | + navi_aputaulut
|   |   |   - Aputaulut
|   |   |
|   |   | + navi_tarkastajat
|   |   |   - Tarkastajat
|   |   |
|   |   | + navi_reviirit
|   |   |   - Reviirit
|   |   |
|   |   | + navi_uusipesa
|   |   |   - Uusi Pesd
|   |
|   | + kutsuja
|   |   - mainscreen
|
| + muuta
|   |
|   | + vuosi
|   |   -
|   |
|   | + vuodelle
|   |   -
|   |
|   | + id
|   |   -
|
| + arvo
```

```
|   |
|   | + aste_pit_sek
|   |   -
|   |
|   | + aste_lev_sek
|   |   -
|   |
|   | + mista_vuosi
|   |   -
|   |
|   | + reviiri_kunta
|   |   -
|   |
|   | + minne_vuosi
|   |   -
|   |
|   | + pesa_kunta
|   |   -
|   |
|   | + yht_pituus
|   |   -
|   |
|   | + aste_lev_ast
|   |   -
|   |
|   | + suuralue
|   |   -
|   |
|   | + yht_leveys
|   |   -
|   |
|   | + pesa_id
|   |   -
|   |
|   | + aste_pit_ast
|   |   -
|   |
|   | + aste_pit_min
|   |   -
|   |
|   | + aste_lev_min
|   |   -
|   |
|   | + reviiri_nimi
|   |   -
|   |
|   | + pesa_nimi
|   |   -
|   |
|   | + reviiri_id
|   |   -
|   |
|   | + ymparistokeskus
|   |   -
```

E municipality.ftl

```
|
| + ilmoitus
|     |
|     | + syy
|     |     - freemarker.template.SimpleSequence@e16785
|
| + teksti
|     |
|     | + hali_sisaan
|     |     - Sisddn
|     |
|     | + hali_otsikko
|     |     - Haliaeetus - Sisddnkirjautuminen
|     |
|     | + hali_salasana
|     |     - Salasana
|     |
|     | + hali_h1
|     |     - Haliaeetus-jdrjestelmd
|     |
|     | + hali_tunnus
|     |     - Kdyttdjdtunnus
|     |
|     | + static_url
|     |     - http://localhost:8080\Haliaeetus\HaliaeetusController
|
| + arvo
|     |
|     | + kieli
|     |     - fi
|     |
|     | + tunnus
|     |     -
```

F pesa.ftl

```
|
| + ilmoitus
|     - false
|
| + teksti
|     |
|     | + pesa_lentopoik_lkm
|     |     - Lentopoikasten lkm
|     |
|     | + pesa_pesiva_laji
|     |     - Pesivälaji, jos ei merikotka
|     |
|     | + pesa_reng_poik_lkm
|     |     - Rengatusikäisten poikasten lkm
|     |
|     | + pesa_p_nilkka_max
|     |     - Nilkan paksuus maksimi
|     |
|     | + pesa_valokuva_ei
|     |     - Ei
|     |
|     | + pesa_et_tie
|     |     - Autolla ajettavaan tiehen
|     |
|     | + pesa_kuolleita_lkm
|     |     - Kuolleiden poikasten lkm
|     |
|     | + static_url
|     |     - http://localhost:8080/Haliaeetus/HaliaeetusController
|     |
|     | + pesa_rauh_kommentti
|     |     - Kommentti
|     |
|     | + pesa_et_jarvi
|     |     - Järven tai lammen rantaan
|     |
|     | + pesa_pesa_halk_max
|     |     - Pesän pinnan suurin halkaisija
|     |
|     | + pesa_ast_leveys
|     |     - leveys
|     |
|     | + pesa_et_moottorikelkka
|     |     - Moottorikelkkareittiin
|     |
|     | + pesa_reviiri
|     |     - Reviirin nimi
|     |
|     | + pesa_yht_pituus
|     |     - Yhtenäiskoord. pituus
|     |
|     | + pesa_tie_ei
|     |     - Ei
|     |
|     | + pesa_as_lkm_1000
|     |     - Säde 1000m
|     |
|     | + pesa_tarkastaja2
|     |     - Tarkastaja2
|     |
|     | + pesa_p_munanpituus
|     |     - Munan pituus * leveys
|     |
|     | + pesa_elavia_lkm
|     |     - Elävien poikasten lkm
|     |
|     | + pesa_r_taulu_kieli
|     |     - Kieli
|     |
|     | + pesa_rauh_pvm
|     |     - Tietojen pvm
|     |
```

| | + pesa_ast_pituus
| | - Astekoord. pituus
| |
| | + pesa_k_tai_n
| | - Koiraan tai naaraan
| |
| | + pesa_tarkastusvuodet
| | - Edelliset tarkastukset
| |
| | + pesa_tyvi_ymparys
| | - Rungon ympärys
| |
| | + pesa_rauhoitettu_ei_ilmoitettu
| | - Ei ilmoitettu
| |
| | + pesa_tekopesa_kylla
| | - Kyllä
| |
| | + pesa_pesan_nakyvyys
| | - Pesän näkyvyys maastosta/vesiltä
| |
| | + pesa_n_oksennus
| | - Oksennuspalloja
| |
| | + pesa_epaonni_syy
| | - Jos pesintä epäonnistui, niin epäonnistumisen syy
| |
| | + pesa_muutpesat
| | - Reviirin muut pesät
| |
| | + pesa_tarkastaja1
| | - Tarkastaja1
| |
| | + pesa_n_kuolleita_a
| | - Kuolleita aikuisia
| |
| | + pesa_p_nilkka_min
| | - Nilkan paksuus minimi
| |
| | + pesa_k
| | - Koiras
| |
| | + pesa_b_naytteita_lahetetty
| | - Näytteitä pesältä lähetetty
| |
| | + pesa_nimi
| | - Pesän nimi
| |
| | + pesa_otsikko_uusi_tarkastus
| | - HALIAEETUS - Uusi tarkastus
| |
| | + pesa_koordinaatit
| | - Koordinaatit
| |
| | + pesa_n
| | - Naaras
| |
| | + pesa_puusto_ika
| | - Ikä
| |
| | + pesa_r_mannyt
| | - Männyt
| |
| | + pesa_rengas_oikea
| | - Oikean tunnus
| |
| | + pesa_omistaja
| | - Pesän palstan omistaja
| |
| | + pesa_latva_ymparys
| | - Rungon ympärys
| |
| | + pesa_b_rauhoitus
| | - Rauhoitus

| |
| + pesa_et_viljapelto
| - Viljeltyyn peltoon
| |
| + pesa_p_siiven_pituus
| - Siiven pituus (mm)
| |
| + pesa_n_kuolleita_p
| - Kuolleita poikasia
| |
| + pesa_p_nayte
| - Näytteet
| |
| + pesa_tekopesa_ei
| - Ei
| |
| + pesa_p_siiven_mittaus
| - Siiven mittausmenetelmä
| |
| + pesa_korkeus
| - Puun korkeus
| |
| + pesa_mittaustapa
| - Mittaustapa
| |
| + pesa_mitattu
| - (mitattu)
| |
| + pesa_et_talvitie
| - Talvitiehen
| |
| + pesa_et_as
| - (Kesä)asuntoon
| |
| + pesa_tark_pvm
| - Tarkastuspvm
| |
| + pesa_r_pohjoiseen
| - Pesältä
| |
| + pesa_aikuisia_lkm
| - Aikuisten lkm
| |
| + pesa_pesa_halk_min
| - Pienin halkaisija
| |
| + pesa_laji
| - Pesän rakentanut laji
| |
| + pesa_p_rengas_o_vari
| - Oikean jalan renkaan väri
| |
| + pesa_p_rengas_v_vari
| - Vasemman jalan renkaan väri
| |
| + pesa_saaliit
| - Saaliit
| |
| + pesa_pvm_muoto
| - pp.kk.vvvv
| |
| + pesa_tyvi_halkaisija
| - Rungon halkaisija
| |
| + pesa_tekopesa
| - Tekopesä
| |
| + pesa_et_avosuo
| - Avosuon reunaan
| |
| + pesa_tark_tunti
| - Kellonaika
| |
| + pesa_epaonni_tark

| | | - Tarkkuus
| | |
| | + pesa_pestul_kommentti
| | | - Pesimistuloksen kommentti
| | |
| | + pesa_tark_pvm_tark
| | | - Päivämäärän tarkkuus
| | |
| | + pesa_as_lkm
| | | - Asuintalojen ja kesäasuntojen määrä
| | |
| | + pesa_r_muu
| | | - Muut
| | |
| | + pesa_pesa_kunto
| | | - Pesän kunto
| | |
| | + pesa_ymp_mit_pvm
| | | - Mittauspäivämäärä
| | |
| | + pesa_p_muna4
| | | - 4
| | |
| | + pesa_historia
| | | - Historia
| | |
| | + pesa_et_maasta
| | | - 25m pohjoiseen
| | |
| | + pesa_p_rengas_vasen
| | | - Vasemman jalan renkaan tunnus
| | |
| | + pesa_b_tiedot_aikuisista
| | | - Tiedot aikuisista
| | |
| | + pesa_r_lkm
| | | - lkm
| | |
| | + pesa_p_kupu
| | | - Kupu
| | |
| | + pesa_p_muna3
| | | - 3
| | |
| | + pesa_r_taulu_pvm
| | | - Kiinnityspäivä
| | |
| | + pesa_rengas_vasen
| | | - Vasemman tunnus
| | |
| | + pesa_luonnonpesa
| | | - Jos luonnonpesä, niin löytymisvuosi
| | |
| | + pesa_p_rengas_oikea
| | | - Oikean jalan renkaan tunnus
| | |
| | + pesa_tie
| | | - Tieyhteys (myös autoja kuljettavalla lautalla tms.)
| | |
| | + pesa_et_latvasta
| | | - Etäisyys latvasta
| | |
| | + pesa_latva_halkaisija
| | | - Rungon halkaisija
| | |
| | + pesa_id
| | | - PesäID
| | |
| | + pesa_n_saaliit
| | | - Saalisnäytteitä
| | |
| | + pesa_k_rengas
| | | - Ilmoita onko koiraalla nähty renkaita
| | |

| | + pesa_maastotyyppi
| | - Maasto
| |
| | + pesa_pussia
| | - pussia
| |
| | + pesa_pesa_merkit
| | - Merkit pesän ympärillä
| |
| | + pesa_sijainti
| | - Pesän sijainti
| |
| | + pesa_puulaji
| | - Puulaji
| |
| | + pesa_pesa_korkeus
| | - Pesän korkeus
| |
| | + pesa_et_kalanviljely
| | - Kalanviljelylaitokseen
| |
| | + pesa_p_paino
| | - Paino
| |
| | + pesa_p_nokka_tyvi
| | - Nokan korkeus tyvestä
| |
| | + pesa_suojelualue
| | - Jos pesä suojelualueella, niin suojelualueen virallinen nimi
| |
| | + pesa_et_avohakkuu
| | - Avohakkuun tai siemenpuuston reunaan
| |
| | + pesa_r_itaan
| | - 25m itään
| |
| | + pesa_ei_ilmoitettu
| | - Ei ilmoitettu
| |
| | + pesa_saari_tyyppi
| | - Saarityyppi
| |
| | + pesa_munia_lkm
| | - Lopullinen munamäärä
| |
| | + pesa_kunta
| | - Kunta
| |
| | + pesa_puusto
| | - Puusto
| |
| | + pesa_b_pesan_mittoja
| | - Pesäpuun ja pesän mittoja
| |
| | + pesa_ei
| | - Ei
| |
| | + pesa_pesan_alla
| | - Välittömästi pesän alla
| |
| | + pesa_n_irtosulkia
| | - Merikotkan irtosulkia ja höyheniä
| |
| | + pesa_tekopesa_ei_ilmoitettu
| | - Ei ilmoitettu
| |
| | + pesa_tie_kylla
| | - Kyllä
| |
| | + pesa_rauhoitettu_ei
| | - Ei
| |
| | + pesa_onko_etsitty
| | - Onko uusia pesiä etsitty

```
|   |
|   | + pesa_otsikko_uusi_pesa
|   |   - HALIAEETUS - Uusi pesä
|   |
|   | + pesa_pesimistulos
|   |   - Pesimistulos
|   |
|   | + pesa_et_ilmajohto
|   |   - Ilmajohtoon
|   |
|   | + pesa_n_siruja
|   |   - Munan siruja
|   |
|   | + pesa_myrkky
|   |   - Myrkyt
|   |
|   | + pesa_kpl
|   |   - kpl
|   |
|   | + pesa_rauhoitettu_kylla
|   |   - Kyllä
|   |
|   | + pesa_rinnan_korkeus
|   |   - Rinnan korkeudella (130cm)
|   |
|   | + pesa_valokuva_ei_ilmoitettu
|   |   - Ei ilmoitettu
|   |
|   | + pesa_h1_pesatiedot
|   |   - Pesätiedot
|   |
|   | + pesa_b_puuston_mittaukset
|   |   - Puuson relaskooppimittaukset
|   |
|   | + pesa_r_etelaan
|   |   - 25m etelään
|   |
|   | + pesa_p_muna1
|   |   - muna1/pull1
|   |
|   | + pesa_p_nokka_pituus
|   |   - Nokan pituus
|   |
|   | + pesa_p_sukupuoli
|   |   - Sukupuoli
|   |
|   | + pesa_r_keskipituus
|   |   - keskipit.
|   |
|   | + pesa_b_ymparisto
|   |   - Pesän ympäristön tietoja
|   |
|   | + pesa_tie_ei_ilmoitettu
|   |   - Ei ilmoitettu
|   |
|   | + pesa_h1_uusi_tarkastus
|   |   - Uusi tarkastus
|   |
|   | + pesa_palsta_rauhoitus
|   |   - Pesän palstan rauhoitustilanne
|   |
|   | + pesa_omist_kommentti
|   |   - Kommentti
|   |
|   | + pesa_elavyys
|   |   - Elävyys
|   |
|   | + pesa_pesatarkastus_kommentti
|   |   - Kommentti
|   |
|   | + pesa_h1_uusi_pesa
|   |   - Uusi pesä
|   |
|   | + pesa_r_kuuset
```

```
|   |   - Kuuset
|   |
|   | + pesa_pesimist_tark
|   |   - Tarkkuus
|   |
|   | + pesa_vari
|   |   - väri
|   |
|   | + pesa_kuoriutumattomia_lkm
|   |   - Kuoriutumattomien munien lkm
|   |
|   | + pesa_rauhoitettu
|   |   - Rauhoitustaulun kiinnityspvm
|   |
|   | + pesa_n_rengas
|   |   - Ilmoita onko naaraalla nähty renkaita
|   |
|   | + pesa_r_lanteen
|   |   - 25m länteen
|   |
|   | + pesa_p_muna2
|   |   - 2
|   |
|   | + pesa_n_munia
|   |   - Munia
|   |
|   | + pesa_tark_tapa
|   |   - Tarkastustapa
|   |
|   | + pesa_et_lahipuu
|   |   - Sopivaan toiseen pesäpuuhun
|   |
|   | + pesa_yht_leveys
|   |   - leveys
|   |
|   | + pesa_tarkka_sijainti
|   |   - Kylä, saari, tms.
|   |
|   | + pesa_pesa_mit_pvm
|   |   - Mittauspäivämäärä
|   |
|   | + pesa_valokuva_kylla
|   |   - Kyllä
|   |
|   | + pesa_as_lkm_500
|   |   - Säde 500m
|   |
|   | + pesa_b_tiedot_poikasista
|   |   - Tiedot poikasista
|   |
|   | + pesa_puusto_kasittely
|   |   - Käsittelyaste
|   |
|   | + pesa_rak_vuosi
|   |   - Rakentamisvuosi
|   |
|   | + pesa_nahdyt_merkit
|   |   - Nähdyt pesinnän merkit
|   |
|   | + pesa_r_pesalta
|   |   - Pesältä
|   |
|   | + pesa_vuosi
|   |   - Pesän tarkastustiedot vuonna
|   |
|   | + pesa_rak_vuosi_tark
|   |   - Tarkkuus
|   |
|   | + pesa_uhat
|   |   - Havaitut uhkatekijät
|   |
|   | + pesa_r_taulu_nro
|   |   - nro
|   |
```

```
|     | + pesa_kylla
|     |    - Kyllä
|     |
|     | + pesa_otsikko_pesatiedot
|     |    - HALIAEETUS - Pesätiedot
|     |
|     | + pesa_valokuva
|     |    - Pesän ympäristöstä otettu valokuva ja lähetetty Eläinmuseoon
|     |
|     | + pesa_et_meri
|     |    - Merenrantaan
|     |
|     | + pesa_tallenna
|     |    - Tallenna
|     |
|     | + pesa_h2_tarkastustiedot
|     |    - Tarkastustiedot
|
| + kutsuja
|    - pesatiedot
|
| + navi
|     |
|     | + kieli
|     |    - fi
|     |
|     | + teksti
|     |     |     |
|     |     |     | + navi_raportit
|     |     |     |    - Raportit
|     |     |     |
|     |     |     | + navi_kunnat
|     |     |     |    - Kunnat
|     |     |     |
|     |     |     | + navi_laji
|     |     |     |    - Laji
|     |     |     |
|     |     |     | + navi_mainscreen
|     |     |     |    - Haku
|     |     |     |
|     |     |     | + navi_ulos
|     |     |     |    - Kirjaudu ulos
|     |     |     |
|     |     |     | + navi_aputaulut
|     |     |     |    - Aputaulut
|     |     |     |
|     |     |     | + navi_tarkastajat
|     |     |     |    - Tarkastajat
|     |     |     |
|     |     |     | + navi_reviirit
|     |     |     |    - Reviirit
|     |     |     |
|     |     |     | + navi_uusipesa
|     |     |     |    - Uusi Pesä
|     |
|     | + kutsuja
|     |    - uusipesa
|
| + arvo
|     |
|     | + rak_laji
|     |    - HALALB
|     |
|     | + tarkastus
|     |     |     |
|     |     |     | + lentopoik_lkm
|     |     |     |    - 1
|     |     |     |
|     |     |     | + munia_lkm
|     |     |     |    -
|     |     |     |
|     |     |     | + reng_poik_lkm
|     |     |     |    - 1
|     |     |     |
```

```
|   |   |   | + kuoriutumattomia_lkm
|   |   |   |   -
|   |   |   |
|   |   |   | + tark_vuosi
|   |   |   |   - 2003
|   |   |   |
|   |   |   | + pesimist_kommentti
|   |   |   |   -
|   |   |   |
|   |   |   | + kuolleita_lkm
|   |   |   |   - 2
|   |
|   | + rak_vuosi_tark
|   |   - 0
|   |
|   | + varit_vasen_4
|   |   -
|   |
|   | + varit_vasen_1
|   |   -
|   |
|   | + dna_nayte_2
|   |   -
|   |
|   | + nayte_o
|   |   -
|   |
|   | + kupu_3
|   |   -
|   |
|   | + k_rengas_v_vari
|   |   -
|   |
|   | + pesimistulos
|   |   - M
|   |
|   | + dna_nayte_1
|   |   -
|   |
|   | + kupu_4
|   |   -
|   |
|   | + k_rengas_o_vari
|   |   -
|   |
|   | + dna_nayte_3
|   |   -
|   |
|   | + elavia_lkm
|   |   - 1
|   |
|   | + epaonni_syy
|   |   -
|   |
|   | + tark_pvm
|   |   - 2003-08-01 00:00:00.0
|   |
|   | + r_taulu_kieli
|   |   -
|   |
|   | + siipi_pituus_m_2
|   |   -
|   |
|   | + vakio_tark_pvm_tark
|   |   - 1
|   |
|   | + nayte_s
|   |   -
|   |
|   | + muulaji
|   |   -
|   |
|   | + puusto
|   |   - M
```

```
|   |
|   | + n_rengas
|   |   - Y
|   |
|   | + dna_nayte_4
|   |   -
|   |
|   | + pesamitat
|   |   |   |
|   |   |   | + pesa_korkeus
|   |   |   |   - 120
|   |   |   |
|   |   |   | + pesa_halk_max
|   |   |   |   - 75
|   |   |   |
|   |   |   | + pesa_mit_pvm
|   |   |   |   - 1990-08-01 00:00:00.0
|   |   |   |
|   |   |   | + pesa_halk_min
|   |   |   |   - 60
|   |
|   | + tekopesa
|   |   -
|   |
|   | + valokuva
|   |   -
|   |
|   | + varit_vasen_3
|   |   -
|   |
|   | + saari_tyyppi
|   |   - P
|   |
|   | + pesan_nakyvyys
|   |   - P
|   |
|   | + autoyhteys
|   |   -
|   |
|   | + pesa_merkit
|   |   - Y
|   |
|   | + tark_tunti
|   |   - 16
|   |
|   | + varit_vasen_2
|   |   -
|   |
|   | + poikaset
|   |   |   |
|   |   |   | + rengas_vasen_1
|   |   |   |   -
|   |   |   |
|   |   |   | + nokka_pituus_1
|   |   |   |   -
|   |   |   |
|   |   |   | + siipi_pituus_3
|   |   |   |   -
|   |   |   |
|   |   |   | + paino_1
|   |   |   |   -
|   |   |   |
|   |   |   | + rengas_oikea_4
|   |   |   |   -
|   |   |   |
|   |   |   | + nilkka_min_2
|   |   |   |   -
|   |   |   |
|   |   |   | + rengas_vasen_2
|   |   |   |   -
|   |   |   |
|   |   |   | + rengas_oikea_1
|   |   |   |   -
|   |   |   |
```

```
|   |   |   | + siipi_pituus_2
|   |   |   |   -
|   |   |   |
|   |   |   | + nokka_tyvi_1
|   |   |   |   -
|   |   |   |
|   |   |   | + nokka_pituus_3
|   |   |   |   -
|   |   |   |
|   |   |   | + nilkka_min_1
|   |   |   |   -
|   |   |   |
|   |   |   | + nokka_pituus_2
|   |   |   |   -
|   |   |   |
|   |   |   | + rengas_oikea_3
|   |   |   |   -
|   |   |   |
|   |   |   | + rengas_vasen_4
|   |   |   |   -
|   |   |   |
|   |   |   | + nokka_tyvi_4
|   |   |   |   -
|   |   |   |
|   |   |   | + rengas_oikea_2
|   |   |   |   -
|   |   |   |
|   |   |   | + nokka_tyvi_3
|   |   |   |   -
|   |   |   |
|   |   |   | + nilkka_max_3
|   |   |   |   -
|   |   |   |
|   |   |   | + siipi_pituus_1
|   |   |   |   -
|   |   |   |
|   |   |   | + paino_2
|   |   |   |   -
|   |   |   |
|   |   |   | + siipi_pituus_4
|   |   |   |   -
|   |   |   |
|   |   |   | + nilkka_min_3
|   |   |   |   -
|   |   |   |
|   |   |   | + nokka_tyvi_2
|   |   |   |   -
|   |   |   |
|   |   |   | + nilkka_max_2
|   |   |   |   -
|   |   |   |
|   |   |   | + nilkka_max_4
|   |   |   |   -
|   |   |   |
|   |   |   | + nilkka_max_1
|   |   |   |   -
|   |   |   |
|   |   |   | + rengas_vasen_3
|   |   |   |   -
|   |   |   |
|   |   |   | + nokka_pituus_4
|   |   |   |   -
|   |   |   |
|   |   |   | + paino_3
|   |   |   |   -
|   |   |   |
|   |   |   | + nilkka_min_4
|   |   |   |   -
|   |   |   |
|   |   |   | + paino_4
|   |   |   |   -
|   |
|   | + sukupuoli_3
|   |   -
```

```
|   |
|   | + tarkastaja2_id
|   |   -
|   |
|   | + puusto_kasittely
|   |   - H
|   |
|   | + nahdyt_merkit
|   |   - Y
|   |
|   | + varit_oikea_2
|   |   -
|   |
|   | + elavyys
|   |   - E
|   |
|   | + puut
|   |   |   |
|   |   |   | + muu_pit_e
|   |   |   |   -
|   |   |   |
|   |   |   | + kuusi_pit_p
|   |   |   |   -
|   |   |   |
|   |   |   | + kuusi_lkm_e
|   |   |   |   -
|   |   |   |
|   |   |   | + manty_lkm_l
|   |   |   |   -
|   |   |   |
|   |   |   | + manty_pit_l
|   |   |   |   -
|   |   |   |
|   |   |   | + kuusi_lkm_p
|   |   |   |   -
|   |   |   |
|   |   |   | + kuusi_lkm_l
|   |   |   |   -
|   |   |   |
|   |   |   | + muu_lkm_l
|   |   |   |   -
|   |   |   |
|   |   |   | + kuusi_pit_l
|   |   |   |   -
|   |   |   |
|   |   |   | + muu_lkm_i
|   |   |   |   -
|   |   |   |
|   |   |   | + kuusi_pit_e
|   |   |   |   -
|   |   |   |
|   |   |   | + manty_lkm_p
|   |   |   |   -
|   |   |   |
|   |   |   | + manty_pit_e
|   |   |   |   -
|   |   |   |
|   |   |   | + muu_pit_i
|   |   |   |   -
|   |   |   |
|   |   |   | + muu_lkm_e
|   |   |   |   -
|   |   |   |
|   |   |   | + muu_lkm_p
|   |   |   |   -
|   |   |   |
|   |   |   | + manty_pit_p
|   |   |   |   -
|   |   |   |
|   |   |   | + kuusi_pit_i
|   |   |   |   -
|   |   |   |
|   |   |   | + manty_lkm_i
|   |   |   |   -
```

```
|   |   |   |
|   |   |   | + muu_pit_p
|   |   |   |   -
|   |   |   |
|   |   |   | + manty_lkm_e
|   |   |   |   -
|   |   |   |
|   |   |   | + muu_pit_l
|   |   |   |   -
|   |   |   |
|   |   |   | + kuusi_lkm_i
|   |   |   |   -
|   |   |   |
|   |   |   | + manty_pit_i
|   |   |   |   -
|   |
|   | + lahetetty
|   |   |   |
|   |   |   | + nayte_p
|   |   |   |   -
|   |   |   |
|   |   |   | + nayte_r
|   |   |   |   -
|   |   |   |
|   |   |   | + nayte_a
|   |   |   |   -
|   |   |   |
|   |   |   | + nayte_m
|   |   |   |   -
|   |
|   | + omist_kommentti
|   |   -
|   |
|   | + aikuiset
|   |   |   |
|   |   |   | + k_rengas_vasen
|   |   |   |   -
|   |   |   |
|   |   |   | + n_rengas_vasen
|   |   |   |   -
|   |   |   |
|   |   |   | + rengas_vasen
|   |   |   |   -
|   |   |   |
|   |   |   | + n_rengas_oikea
|   |   |   |   -
|   |   |   |
|   |   |   | + k_rengas_oikea
|   |   |   |   -
|   |   |   |
|   |   |   | + rengas_oikea
|   |   |   |   -
|   |
|   | + varit_oikea_3
|   |   -
|   |
|   | + pesa_kunto
|   |   - P
|   |
|   | + sukupuoli_4
|   |   -
|   |
|   | + aikuisia_lkm
|   |   - 2
|   |
|   | + varit_oikea_1
|   |   -
|   |
|   | + n_rengas_o_vari
|   |   -
|   |
|   | + tarkastaja3_id
|   |   - 886
|   |
```

```
|   | + sukupuoli_2
|   |   -
|   |
|   | + pesapuu
|   |   |   |
|   |   |   | + latvahalkaisija
|   |   |   |   - 62
|   |   |   |
|   |   |   | + tyviymparys
|   |   |   |   - 82
|   |   |   |
|   |   |   | + et_latvasta
|   |   |   |   - 0
|   |   |   |
|   |   |   | + korkeus
|   |   |   |   - 18
|   |   |   |
|   |   |   | + et_maasta
|   |   |   |   - 17
|   |   |   |
|   |   |   | + latvaymparys
|   |   |   |   - 122
|   |   |   |
|   |   |   | + tyvihalkaisija
|   |   |   |   - 41
|   |
|   | + suojelualue
|   |   - Raumankorpi
|   |
|   | + pesa_kommentti
|   |   -
|   |
|   | + uhat
|   |   -
|   |
|   | + sukupuoli_1
|   |   -
|   |
|   | + n_rengas_v_vari
|   |   -
|   |
|   | + tarkastusvuodet
|   |   -
|   |
|   | + lentopoik_lkm
|   |   - 1
|   |
|   | + palsta_rauhoitus
|   |   - P
|   |
|   | + tark_tapa
|   |   - 1
|   |
|   | + korkeus_tark
|   |   - M
|   |
|   | + koordinaatit
|   |   |   |
|   |   |   | + ast_leveys_min
|   |   |   |   -
|   |   |   |
|   |   |   | + ast_pituus_sek
|   |   |   |   -
|   |   |   |
|   |   |   | + ast_leveys_sek
|   |   |   |   -
|   |   |   |
|   |   |   | + ast_leveys
|   |   |   |   - 6520
|   |   |   |
|   |   |   | + yht_pituus
|   |   |   |   - 35100
|   |   |   |
|   |   |   | + ast_pituus
```

```
|   |   |   |   - 3510
|   |   |   |
|   |   |   | + ast_pituus_min
|   |   |   |   -
|   |   |   |
|   |   |   | + yht_leveys
|   |   |   |   - 65200
|   |
|   | + siipi_pituus_m_3
|   |   -
|   |
|   | + rengas_v_vari
|   |   -
|   |
|   | + siipi_pituus_m_4
|   |   -
|   |
|   | + et_maasta_tark
|   |   - M
|   |
|   | + vaihtopesa
|   |   -
|   |
|   | + puusto_ika
|   |   - N
|   |
|   | + kupu_2
|   |   -
|   |
|   | + etaisyys
|   |   |   |
|   |   |   | + et_lahipuu
|   |   |   |   - 2
|   |   |   |
|   |   |   | + et_as
|   |   |   |   -
|   |   |   |
|   |   |   | + et_talvitie
|   |   |   |   -
|   |   |   |
|   |   |   | + et_jarvi
|   |   |   |   -
|   |   |   |
|   |   |   | + as_lkm_500
|   |   |   |   -
|   |   |   |
|   |   |   | + et_meri
|   |   |   |   - 10
|   |   |   |
|   |   |   | + et_tie
|   |   |   |   -
|   |   |   |
|   |   |   | + et_viljapelto
|   |   |   |   -
|   |   |   |
|   |   |   | + et_avohakkuu
|   |   |   |   -
|   |   |   |
|   |   |   | + et_avosuo
|   |   |   |   -
|   |   |   |
|   |   |   | + et_moottorikelkka
|   |   |   |   -
|   |   |   |
|   |   |   | + et_kalaviljely
|   |   |   |   -
|   |   |   |
|   |   |   | + as_lkm_1000
|   |   |   |   -
|   |   |   |
|   |   |   | + et_ilmajohto
|   |   |   |   -
|   |
|   | + ymp_mit_pvm
```

```
|   |   - 1990-08-01 00:00:00.0
|   |
|   + siipi_pituus_m_1
|   |   -
|   |
|   + et_latvasta_tark
|   |   - M
|   |
|   + reviirinimi
|   |   - Mytsyrä
|   |
|   + kupu_1
|   |   -
|   |
|   + puulaji
|   |   - M
|   |
|   + maastotyyppi
|   |   - Y
|   |
|   + epaonni_tark
|   |   -
|   |
|   + palsta_omistaja
|   |   - V
|   |
|   + sijainti
|   |   - P
|   |
|   + tarkastaja1_id
|   |   - 886
|   |
|   + rauhoitus
|   |   |   |
|   |   |   + r_taulu_nro
|   |   |   |   -
|   |   |   |
|   |   |   + palsta_rauh_pvm
|   |   |   |   - 1990-01-01 00:00:00.0
|   |   |   |
|   |   |   + r_taulu_pvm
|   |   |   |   -
|   |   |   |
|   |   |   + rauh_kommentti
|   |   |   |   -
|   |
|   + rengas_o_vari
|   |   -
|   |
|   + k_rengas
|   |   - Y
|   |
|   + pesimist_tark
|   |   - V
|   |
|   + varit_oikea_4
|   |   -
|   |
|   + pesa
|   |   |   |
|   |   |   + pesa_id
|   |   |   |   - 3
|   |   |   |
|   |   |   + tarkka_sijainti
|   |   |   |   - Saaren keskellä männyssä
|   |   |   |
|   |   |   + kommentti
|   |   |   |   - Mutasaari
|   |   |   |
|   |   |   + rak_vuosi
|   |   |   |   - 1990
|   |   |   |
|   |   |   + loyt_vuosi
|   |   |   |   - 1990
```

```
|   |   |   |
|   |   |   | + pesanimi
|   |   |   |   - Merikotkanpesä alavus
|   |
|   | + vakio_tark_pvm
|   |   - 1990-08-01 00:00:00.0
|   |
|   | + tyvihal_tark
|   |   - M
|   |
|   | + tarkastaja4_id
|   |   -
|   |
|   | + koord_mittaus
|   |   - G
|   |
|   | + tark_pvm_tark
|   |   -
|   |
|   | + latvahal_tark
|   |   - M
|   |
|   | + kunta_tunnus
|   |   - UUKUNI
|   |
|   | + nayte_i
|   |   -
|
| + ilmoitukset
|   |
|   | + syy
|   |   - freemarker.template.SimpleSequence@d9205
|
| + valikko
|   |
|   | + rak_laji
|   |   - freemarker.template.SimpleSequence@edbca8
|   |
|   | + maastotyyppi
|   |   - freemarker.template.SimpleSequence@1fcca7b
|   |
|   | + palsta_rauhoitus
|   |   - freemarker.template.SimpleSequence@1517e5e
|   |
|   | + poikaset
|   |   |   |
|   |   |   | + varit_oikea_1
|   |   |   |   - freemarker.template.SimpleSequence@13f5841
|   |   |   |
|   |   |   | + siipi_pituus_m_2
|   |   |   |   - freemarker.template.SimpleSequence@1bb205a
|   |   |   |
|   |   |   | + varit_vasen_4
|   |   |   |   - freemarker.template.SimpleSequence@48fbc0
|   |   |   |
|   |   |   | + kupu_2
|   |   |   |   - freemarker.template.SimpleSequence@18837f1
|   |   |   |
|   |   |   | + sukupuoli_3
|   |   |   |   - freemarker.template.SimpleSequence@10f0a0
|   |   |   |
|   |   |   | + varit_vasen_1
|   |   |   |   - freemarker.template.SimpleSequence@25bd56
|   |   |   |
|   |   |   | + dna_nayte_2
|   |   |   |   - freemarker.template.SimpleSequence@1b18235
|   |   |   |
|   |   |   | + kupu_3
|   |   |   |   - freemarker.template.SimpleSequence@db95a1
|   |   |   |
|   |   |   | + siipi_pituus_m_3
|   |   |   |   - freemarker.template.SimpleSequence@3c9616
|   |   |   |
|   |   |   | + sukupuoli_2
```

```
|   |   |   |   - freemarker.template.SimpleSequence@1a18ee2
|   |   |   |
|   |   |   | + dna_nayte_1
|   |   |   |   - freemarker.template.SimpleSequence@1abcd9b
|   |   |   |
|   |   |   | + varit_oikea_2
|   |   |   |   - freemarker.template.SimpleSequence@47b35d
|   |   |   |
|   |   |   | + kupu_4
|   |   |   |   - freemarker.template.SimpleSequence@150b45a
|   |   |   |
|   |   |   | + dna_nayte_4
|   |   |   |   - freemarker.template.SimpleSequence@170a650
|   |   |   |
|   |   |   | + varit_oikea_4
|   |   |   |   - freemarker.template.SimpleSequence@b6f7f5
|   |   |   |
|   |   |   | + siipi_pituus_m_4
|   |   |   |   - freemarker.template.SimpleSequence@5113f0
|   |   |   |
|   |   |   | + siipi_pituus_m_1
|   |   |   |   - freemarker.template.SimpleSequence@f41393
|   |   |   |
|   |   |   | + sukupuoli_1
|   |   |   |   - freemarker.template.SimpleSequence@314585
|   |   |   |
|   |   |   | + varit_vasen_3
|   |   |   |   - freemarker.template.SimpleSequence@cb1edc
|   |   |   |
|   |   |   | + varit_oikea_3
|   |   |   |   - freemarker.template.SimpleSequence@1570f5c
|   |   |   |
|   |   |   | + dna_nayte_3
|   |   |   |   - freemarker.template.SimpleSequence@b18494
|   |   |   |
|   |   |   | + sukupuoli_4
|   |   |   |   - freemarker.template.SimpleSequence@773d03
|   |   |   |
|   |   |   | + varit_vasen_2
|   |   |   |   - freemarker.template.SimpleSequence@46a09b
|   |   |   |
|   |   |   | + kupu_1
|   |   |   |   - freemarker.template.SimpleSequence@da5bc0
|   |   |
|   |   | + tarkastus
|   |   |   |   |
|   |   |   |   | + lentopoik_lkm
|   |   |   |   |   - freemarker.template.SimpleSequence@1bdbf9d
|   |   |   |   |
|   |   |   |   | + epaonni_tark
|   |   |   |   |   - freemarker.template.SimpleSequence@6f19d5
|   |   |   |   |
|   |   |   |   | + pesimist_tark
|   |   |   |   |   - freemarker.template.SimpleSequence@91520
|   |   |   |   |
|   |   |   |   | + tark_tapa
|   |   |   |   |   - freemarker.template.SimpleSequence@4a0ac5
|   |   |   |   |
|   |   |   |   | + pesa_merkit
|   |   |   |   |   - freemarker.template.SimpleSequence@1092447
|   |   |   |   |
|   |   |   |   | + tark_tunti
|   |   |   |   |   - freemarker.template.SimpleSequence@12cd8d4
|   |   |   |   |
|   |   |   |   | + epaonni_syy
|   |   |   |   |   - freemarker.template.SimpleSequence@14f5021
|   |   |   |   |
|   |   |   |   | + pesimistulos
|   |   |   |   |   - freemarker.template.SimpleSequence@15da7d
|   |   |   |   |
|   |   |   |   | + muulaji
|   |   |   |   |   - freemarker.template.SimpleSequence@bb6255
|   |   |   |   |
|   |   |   |   | + elavia_lkm
```

```
|   |   |   |   - freemarker.template.SimpleSequence@34f445
|   |   |   |
|   |   |   + nahdyt_merkit
|   |   |   |   - freemarker.template.SimpleSequence@90ed81
|   |   |
|   |   + palsta_omistaja
|   |   |   - freemarker.template.SimpleSequence@d8c3ee
|   |   |
|   |   + tarkastaja2_id
|   |   |   - freemarker.template.SimpleSequence@1277a30
|   |   |
|   |   + tarkastaja1_id
|   |   |   - freemarker.template.SimpleSequence@c707c1
|   |   |
|   |   + lahetetty
|   |   |   |   |
|   |   |   |   + nayte_o
|   |   |   |   |   - freemarker.template.SimpleSequence@ce3b62
|   |   |   |   |
|   |   |   |   + nayte_s
|   |   |   |   |   - freemarker.template.SimpleSequence@19ccb73
|   |   |   |   |
|   |   |   |   + nayte_i
|   |   |   |   |   - freemarker.template.SimpleSequence@f12b72
|   |   |   |
|   |   + aikuiset
|   |   |   |   |
|   |   |   |   + k_rengas
|   |   |   |   |   - freemarker.template.SimpleSequence@15b6aad
|   |   |   |   |
|   |   |   |   + k_rengas_o_vari
|   |   |   |   |   - freemarker.template.SimpleSequence@b890dc
|   |   |   |   |
|   |   |   |   + aikuisia_lkm
|   |   |   |   |   - freemarker.template.SimpleSequence@12e7cb6
|   |   |   |   |
|   |   |   |   + rengas_v_vari
|   |   |   |   |   - freemarker.template.SimpleSequence@fd9b97
|   |   |   |   |
|   |   |   |   + n_rengas_o_vari
|   |   |   |   |   - freemarker.template.SimpleSequence@1f1e666
|   |   |   |   |
|   |   |   |   + n_rengas_v_vari
|   |   |   |   |   - freemarker.template.SimpleSequence@4d40df
|   |   |   |   |
|   |   |   |   + k_rengas_v_vari
|   |   |   |   |   - freemarker.template.SimpleSequence@1de041e
|   |   |   |   |
|   |   |   |   + rengas_o_vari
|   |   |   |   |   - freemarker.template.SimpleSequence@e05ad6
|   |   |   |   |
|   |   |   |   + n_rengas
|   |   |   |   |   - freemarker.template.SimpleSequence@16bb7d9
|   |   |   |
|   |   + pesa
|   |   |   |   |
|   |   |   |   + tekopesa
|   |   |   |   |   - freemarker.template.SimpleSequence@f346dc
|   |   |   |   |
|   |   |   |   + rak_vuosi_tark
|   |   |   |   |   - freemarker.template.SimpleSequence@1b14530
|   |   |   |   |
|   |   |   |   + sijainti
|   |   |   |   |   - freemarker.template.SimpleSequence@135877f
|   |   |   |   |
|   |   |   |   + pesa_kunto
|   |   |   |   |   - freemarker.template.SimpleSequence@152b6f5
|   |   |   |   |
|   |   |   |   + kunta_tunnus
|   |   |   |   |   - freemarker.template.SimpleSequence@169a50b
|   |   |   |
|   |   + vaihtopesa
|   |   |   - freemarker.template.SimpleSequence@d297c0
|   |   |
```

```
|   | + pesapuusto
|   |   |   |
|   |   |   | + puusto_ika
|   |   |   |   - freemarker.template.SimpleSequence@1e8e5a7
|   |   |   |
|   |   |   | + puusto_kasittely
|   |   |   |   - freemarker.template.SimpleSequence@13d422d
|   |   |   |
|   |   |   | + puusto
|   |   |   |   - freemarker.template.SimpleSequence@c3d062
|   |
|   | + rauhoitettu
|   |   - freemarker.template.SimpleSequence@1a5fb5a
|   |
|   | + r_taulu_kieli
|   |   - freemarker.template.SimpleSequence@1906df
|   |
|   | + tarkastaja4_id
|   |   - freemarker.template.SimpleSequence@1123c5f
|   |
|   | + koord_mittaus
|   |   - freemarker.template.SimpleSequence@39bf12
|   |
|   | + vakio_tark_pvm_tark
|   |   - freemarker.template.SimpleSequence@12f4538
|   |
|   | + tarkastaja3_id
|   |   - freemarker.template.SimpleSequence@69695f
|   |
|   | + pesapuu
|   |   |   |
|   |   |   | + tyvihal_tark
|   |   |   |   - freemarker.template.SimpleSequence@84de3c
|   |   |   |
|   |   |   | + puulaji
|   |   |   |   - freemarker.template.SimpleSequence@11a47df
|   |   |   |
|   |   |   | + latvahal_tark
|   |   |   |   - freemarker.template.SimpleSequence@648938
|   |   |   |
|   |   |   | + et_latvasta_tark
|   |   |   |   - freemarker.template.SimpleSequence@6cbecf
|   |   |   |
|   |   |   | + et_maasta_tark
|   |   |   |   - freemarker.template.SimpleSequence@16321e6
|   |   |   |
|   |   |   | + korkeus_tark
|   |   |   |   - freemarker.template.SimpleSequence@14683c0
|   |   |   |
|   |   |   | + elavyys
|   |   |   |   - freemarker.template.SimpleSequence@73bc22
|   |
|   | + tark_pvm_tark
|   |   - freemarker.template.SimpleSequence@449afc
|   |
|   | + valokuva
|   |   - freemarker.template.SimpleSequence@1b3967
|   |
|   | + saari_tyyppi
|   |   - freemarker.template.SimpleSequence@155d4be
|   |
|   | + pesan_nakyvyys
|   |   - freemarker.template.SimpleSequence@21a1e
|   |
|   | + autoyhteys
|   |   - freemarker.template.SimpleSequence@e312
|   |
|   | + tarkastusvuodet
|   |   - freemarker.template.SimpleSequence@3bce70
|   |
|   | + reviirinimi
|   |   - freemarker.template.SimpleSequence@6e56ae
```

G report.ftl

```
|
| + ilmoitus
|     |
|     | + syy
|     |     - freemarker.template.SimpleSequence@aa3e5b
|
| + teksti
|     |
|     | + raportti_aikavali
|     |     - Aikavdli
|     |
|     | + raportti_ympkeskus
|     |     - Ympdristvkeskus
|     |
|     | + raportti_uusipesa
|     |     - Uusi pesd
|     |
|     | + raportti_otsikko
|     |     - Raportit
|     |
|     | + raportti_suuralue
|     |     - Suuralue
|     |
|     | + raportti_alaotsikko
|     |     - Raporttityyppi
|     |
|     | + static_url
|     |     - http://localhost:8080\Haliaeetus\HaliaeetusController
|     |
|     | + raportti_vanhapesa
|     |     - Vanha pesd
|     |
|     | + raportti_pesa
|     |     - Pesd
|     |
|     | + raportti_alueet
|     |     - Alueet
|     |
|     | + raportti_tallenna
|     |     - Tallenna Tiedostoon
|     |
|     | + raportti_reviiri
|     |     - Reviiri
|     |
|     | + raportti_kunta
|     |     - Kunta
|
| + navi
|     |
|     | + kieli
|     |     - fi
|     |
|     | + teksti
|     |     |
|     |     | + navi_raportit
|     |     |     - Raportit
|     |     |
|     |     | + navi_kunnat
|     |     |     - Kunnat
|     |     |
|     |     | + navi_laji
|     |     |     - Laji
|     |     |
|     |     | + navi_mainscreen
|     |     |     - Haku
|     |     |
|     |     | + navi_ulos
|     |     |     - Kirjaudu ulos
|     |     |
|     |     | + navi_aputaulut
|     |     |     - Aputaulut
|     |     |
```

```
|     |     | + navi_tarkastajat
|     |     |     - Tarkastajat
|     |     |
|     |     | + navi_reviirit
|     |     |     - Reviirit
|     |     |
|     |     | + navi_uusipesa
|     |     |     - Uusi Pesd
|     |
|     | + kutsuja
|     |     - raportit
|
| + arvo
|     |
|     | + kieli
|     |     - fi
|     |
|     | + tunnus
|     |     -
|
| + valikko
|     |
|     | + reviiri_kunta
|     |     - freemarker.template.SimpleSequence@e3e881
|     |
|     | + pesa_kunta
|     |     - freemarker.template.SimpleSequence@b57af1
|     |
|     | + ymparistokeskus
|     |     - freemarker.template.SimpleSequence@9a99eb
|     |
|     | + suuralue
|     |     - freemarker.template.SimpleSequence@1ef45e0
```

H reviirit.ftl

```
|
| + syy_otsikko_haku
|     |
|     | + olemassa
|     |     - false
|
| + tulos
|     |
|     | + olemassa
|     |     - false
|
| + ilmoitus
|     |
|     | + syy
|     |     - freemarker.template.SimpleSequence@6d670a
|     |
|     | + olemassa
|     |     - false
|
| + ilmoitus_haku
|     |
|     | + syy
|     |     - freemarker.template.SimpleSequence@162f16
|     |
|     | + olemassa
|     |     - false
|
| + teksti
|     |
|     | + reviirit_reviirinimi
|     |     - Reviirin nimi
|     |
|     | + reviirit_h3_haetut_reviirit
|     |     - Haetut reviirit
|     |
|     | + reviirit_muuta
|     |     - Muuta
|     |
|     | + reviirit_h1_reviirit
|     |     - REVIIRIT
|     |
|     | + reviirit_hae
|     |     - Hae
|     |
|     | + static_url
|     |     - http://localhost:8080\Haliaeetus\HaliaeetusController
|     |
|     | + reviirit_reviiri_id
|     |     - Reviiri id
|     |
|     | + reviirit_kuntatunnus
|     |     - Kuntatunnus
|     |
|     | + reviirit_vanha_reviirinro
|     |     - Reviirin vanha numero
|     |
|     | + reviirit_kommentti
|     |     - Kommentti
|     |
|     | + reviirit_otsikko
|     |     - Reviirit
|     |
|     | + reviirit_h3_reviiritietojen_haku
|     |     - Reviiritietojen haku
|     |
|     | + reviirit_h3_uuden_reviirin_lisays
|     |     - Uuden reviirin lisdys
|     |
|     | + kunnat_kuntatunnus
|     |     - Kuntatunnus
|     |
|     | + reviirit_lisaa
```

```
|     |     - Lisdd
|
| + ilmoitus_lisays
|     |
|     | + syy
|     |     - freemarker.template.SimpleSequence@87e9bf
|     |
|     | + olemassa
|     |     - false
|
| + navi
|     |
|     | + kieli
|     |     - fi
|     |
|     | + teksti
|     |     |
|     |     | + navi_raportit
|     |     |     - Raportit
|     |     |
|     |     | + navi_kunnat
|     |     |     - Kunnat
|     |     |
|     |     | + navi_laji
|     |     |     - Laji
|     |     |
|     |     | + navi_mainscreen
|     |     |     - Haku
|     |     |
|     |     | + navi_ulos
|     |     |     - Kirjaudu ulos
|     |     |
|     |     | + navi_aputaulut
|     |     |     - Aputaulut
|     |     |
|     |     | + navi_tarkastajat
|     |     |     - Tarkastajat
|     |     |
|     |     | + navi_reviirit
|     |     |     - Reviirit
|     |     |
|     |     | + navi_uusipesa
|     |     |     - Uusi Pesd
|     |
|     | + kutsuja
|     |     - reviirit
|
| + muuta
|     |
|     | + kommentti
|     |     -
|     |
|     | + reviirin_kunta
|     |     -
|     |
|     | + kuntatunnus
|     |     -
|     |
|     | + vanha_reviirinro
|     |     -
|     |
|     | + reviirinimi
|     |     -
|     |
|     | + reviiri_id
|     |     -
|
| + syy_otsikko_lisays
|     |
|     | + olemassa
|     |     - false
|
| + lisaa
|     |
```

```
|     | + reviirinimi
|     |    -
|     |
|     | + kunta_tunnus
|     |    -
|
| + haku
|     |
|     | + kommentti
|     |    -
|     |
|     | + reviirin_kunta
|     |    -
|     |
|     | + vanha_reviirinro
|     |    -
|     |
|     | + reviirinimi
|     |    -
|     |
|     | + reviiri_id
|     |    -
|     |
|     | + kunta_tunnus
|     |    -
```

I login.ftl

```
|
| + ilmoitus
|     |
|     | + syy
|     |     - freemarker.template.SimpleSequence@e16785
|
| + teksti
|     |
|     | + hali_sisaan
|     |     - Sisddn
|     |
|     | + hali_otsikko
|     |     - Haliaeetus - Sisddnkirjautuminen
|     |
|     | + hali_salasana
|     |     - Salasana
|     |
|     | + hali_h1
|     |     - Haliaeetus-jdrjestelmd
|     |
|     | + hali_tunnus
|     |     - Kdyttdjdtunnus
|     |
|     | + static_url
|     |     - http://localhost:8080\Haliaeetus\HaliaeetusController
|
| + arvo
|     |
|     | + kieli
|     |     - fi
|     |
|     | + tunnus
|     |     -
```

J tarkastajat.ftl

```
|
| + ilmoitus
|     |
|     | + syy
|     |     - freemarker.template.SimpleSequence@e16785
|
| + teksti
|     |
|     | + hali_sisaan
|     |     - Sisddn
|     |
|     | + hali_otsikko
|     |     - Haliaeetus - Sisddnkirjautuminen
|     |
|     | + hali_salasana
|     |     - Salasana
|     |
|     | + hali_h1
|     |     - Haliaeetus-jdrjestelmd
|     |
|     | + hali_tunnus
|     |     - Kdyttdjdtunnus
|     |
|     | + static_url
|     |     - http://localhost:8080\Haliaeetus\HaliaeetusController
|
| + arvo
|     |
|     | + kieli
|     |     - fi
|     |
|     | + tunnus
|     |     -
```