HELSINGIN YLIOPISTO

TIETOJENKÄSITTELYTIETEEN LAITOS

OHJELMISTOTUOTANTOPROJEKTI

HABA 2004

Toteutusdokumentti

Teemu Virtanen
Riina Henriksson
Ahti Kare
Marko Lähde
Antti Mäki
Mika Stenberg

Versiohistoria

Versio	Päivämäärä	Kommentit/muutokset
0.1	05.12.04	Ensimmäinen versio.
1.0	17.12.04	Lopullinen versio. Tarkennettu Luokat-lukua.

Sisältö

1 Johdanto

Tämä dokumentti kuvaa syksyllä 2004 Helsingin yliopiston Tietojenkäsittelytieteen laitoksella Haba2004-ryhmän toteuttaman Liiketoimintaverkostojen hallintapaneeli -sovelluksen teknisen toteutuksen. Dokumentin sisältö noudattelee suunnitteludokumentin rakennetta ja sisältöä ja toimii siten myös suunnitteluvaiheen viimeisimpänä dokumentaationa. Sisältö on suunnattu sovelluksen ylläpitäjille. Tekniset ykityiskohdat vastaavat lopullista toteutusta. Viimeinen luku sisältää luokkakaavioita järjestelmän pakkausten välisistä suhteista sekä keskeisten luokkien sisäisestä rakenteesta.

1.1 Järjestelmän yleiskuvaus

. Projektissa toteutettiin käyttöliittymä liiketoimintaverkostojen osapuolten yhteistoimintaa helpottavaan järjestelmään. Järjestelmän kautta käyttäjät pääsevät tarkastelemaan verkoston tilatietoja sekä rajoitetusti muuttamaan ja hallinnoimaan niitä. Käyttäjille esitetään verkoston rakenne graafisesti, mikäli se ko. tilanteessa on mahdollista.

Käyttöliittymän pohjana toimii asiakkaan toteuttama liiketoimintaverkostojen hallintasovelluksen ydin, ns. sopimusvarasto. Tuotetun käyttöliittymän avulla hyödynnetään pohjalle rakennettua sovellusta hajautetusti WWW:n välityksellä.

Asiakkaan taustajärjestelmä ei ollut Haba2004 -projektin alkaessa vielä valmis, vaan myös sitä on kehitetty ja muokattu käyttöliittymän määrittely- ja suunnitteluvaiheen aikana. Nämä tehdyt muutokset on kuvattu tämän dokumentin ulkopuolella. Kyseisten muutosten dokumentoinnista vastaa taustajärjestelmän toimittaja Janne Metso.

1.2 Laite- ja ohjelmistoympäristö

Tässä luvussa kuvataan sovelluksen ajo- ja kehitysympäristö.

1.2.1 Ajoympäristö

Projektissa tuotettavat HTML -sivut noudattavat HTML 4.0 spesifikaatiota ja vaativat näin ollen käyttäjiltään selaimen, joka on yhteensopiva sen kanssa. Samoin CSS -tyylimäärittelyt vaativat toimiakseen selaimen, joka tukee CSS:n versio 2:sta. Kehitysvaiheessa sivut on järjestelmällisesti testattu IE 6.0 sekä Mozilla 1.5 selaimilla.

JSP- sivut vaativat ympäristökseen jonkun JSP:tä tukevan palvelinohjelmiston. Ensimmäiseksi käytettäväksi tuotantopalvelimeksi valittiin Apache Tomcat 4.0. Sovelluksen toimivuus on testattu tällä palvelimella. Sovelluksen asentaminen on kuvattu erillisessä ohjedokumentissa.

Toteutuskielenä käytettiin Javaa. Tuotettu Java- koodi on J2SE 1.4.2 -yhteensopivaa. Käyttöliittymäosissa käytetty JSP (Java Server Pages) koodi on Sun Microsystemsin JSP 2.0-spesifikaation kanssa yhteensopivaa. Käyttöliittymän tapahtumankäsittelijänä toimiva Java Servlet -luokka on Servlet -spesifikaatio version 2.4 mukainen. Sovelluksessa käytetyt ulkoiset ohjelmakirjastot on kuvattu asennusohjeessa.

Ohjelmisto on pitkälti laitteistoriippumaton. Java vaatii toimiakseen jonkin moniajoa tukevan käyttöjärjestelmän. Lisäksi järjestelmä vaatii käyttöönsä verkkoyhteyden, jotta yhteys sopimusvarastoon voidaan muodostaa.

Suorittimen tehoon tai muistinkäyttöön liittyviä listattuja vaatimuksia ei ole. Sovelluksen suorituskyky on testattu ja riittäväksi todettu Tietojenkäsittelytieteen laitoksen Alkokrunni-laitteiston Apache Tomcat 4.0 palvelinympäristössä. Suorituskyvyn riittävyys on todettu yhdessä asiakkaan kanssa pidetyssä sovelluksen hyväksymistestauksessa 7.12.2004.

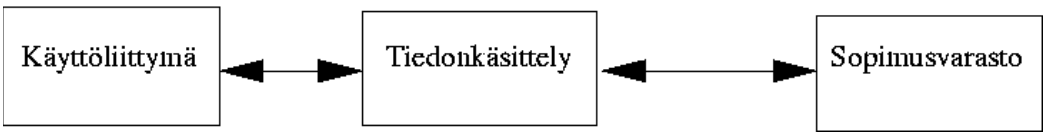
1.2.2 Kehitysympäristö

Toteutusvaiheessa käytettiin Java-koodin tuottamiseen Eclipse-kehitysympäristön versiota 3.0.1, sekä Java J2EE-teknologian vaatimaa sovelluksen Lomboz 3.0 -lisäkirjastoa. Koodin ulkoasun viimeistelyyn Java Coding Conventions -määrittelyn mukaiseksi käytettiin Eclipsen Jalopy -kirjastoa. Javadocin generoimista varten tarvittavat kommentit lisättiin ohjelmakoodiin manuaalisesti. Kehitysympäristön tukemaa CVS-versionhallintaa, käytettiin tuotetun ohjelmakoodin säilyttämiseen ja julkaisemiseen.

2 Arkkitehtuuri

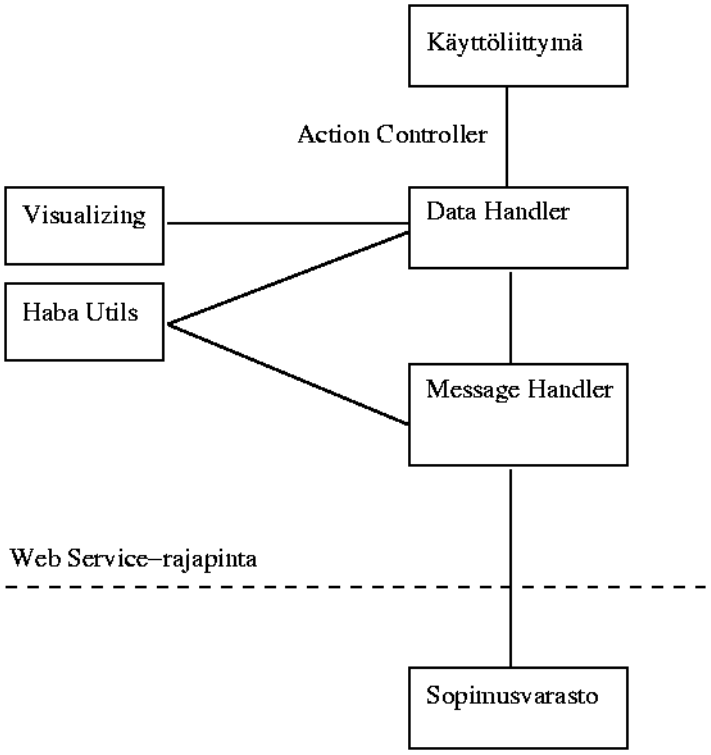
2.1 Yleiskuvaus

Toteutettu liiketoimintaverkoston hallintapaneeli voidaan jakaa kolmeen osaan: Selaimen avulla käytettävään käyttöliittymäosaan, käyttöliittymän hyödyntämään tiedonkäsittelyosaan ja taustalla olevaan sopimusvarastojärjestelmään. Kuvassa 1. on esitelty järjestelmän yleinen kolmikerrosmalli. Tässä tarkastellaan kahden ensimmäisen toteuttamiseen liittyviä kysymyksiä. Sopimusvarasto-järjestelmän on toetuttanut projektin asiakasorganisaatio (Web Pilarcos []). Sopimusvarasto on hallintapaneelin käytössä Web Services -rajapinnan kautta.



Kuva 1: Järjestelmän kolmikerrosmalli

Käyttöliittymä ja tiedon käsittely voidaan edelleen jakaa kolmeen pääosaan. Tässä osassa yleinen arkkitehtuuri vastaa MVC-suunnittelumallia. Selaimessa näkyvä osa, liiketoimintaverkoston hallintapaneeli tuotetaan JSP-sivuilla. Ne muodostavat yhdessä tarvittavien staattisten kuvien sekä HTML- ja CSS-tiedostojen kanssa yhden kokonaisuuden. Jatkossa osaa kutsutaan käyttöliittymäksi. Kokonaisuus vastaa MVC-mallin näkymä (view) osiota. Jsp-sivujen ja tiedonkäsittelyosan välissä on kontrolleri, joka ohjaa pyynnöt oikealle käsittelijälle. Se vastaa MVC-mallin controller -osaa. Sivujen taustalla, tiedonkäsittelyosassa toimivat DataHandler ja MessageHandler. DataHandler huolehtii käyttöliittymän tarvitsemien tietojen keräämisestä ja koostamisesta. Se muodostaa MVC-mallin data (model) osan. MessageHandler huolehtii kommunikaatiosta sopimusvarastojärjestelmään. Näiden lisäksi on kaksi apuluokkaa. Niiden avulla hoidetaan hallintapaneelin lokien kirjoitus ja asetusten lukeminen tiedostosta. Lisäksi on oma osajärjestelmä tarvittavien kuvien tuottamiseksi. Osia ei ole esitetty kuvassa 1. Kuvassa 2. on liiketoimintaverkoston hallintapaneelin osajärjestelmät ja niiden väliset liitokset.



Kuva 2: Liiketoimintaverkoston hallintapaneelin osajärjestelmät ja niiden väliset liitokset

2.2 Osajärjestelmien tehtävät ja rajapinnat

Käyttöliittymäosajärjestelmään kuuluvat kaikki käyttäjälle näkyvät osat ja niiden välittömään tuottamiseen tarvittava toiminnallisuus. Tarvittava toiminnallisuus on toteutettu JSP- ja Java Servlet -tekniikoilla. Muita käyttäjälle näkyviä osia, suoraan tai välillisesti, ovat staattiset HTML - ja CSS -tiedostot sekä käyttöliittymässä käytettävät staattiset kuvat. Käyttöliittymässä näytettävän dynaamisen tietosisällön osajärjestelmä saa alla olevalta DataHandler-osajärjestelmältä. Käyttöliittymä ei käytä muita osajärjestelmiä suoraan, eikä sillä ole omaa julkista palvelurajapintaa.

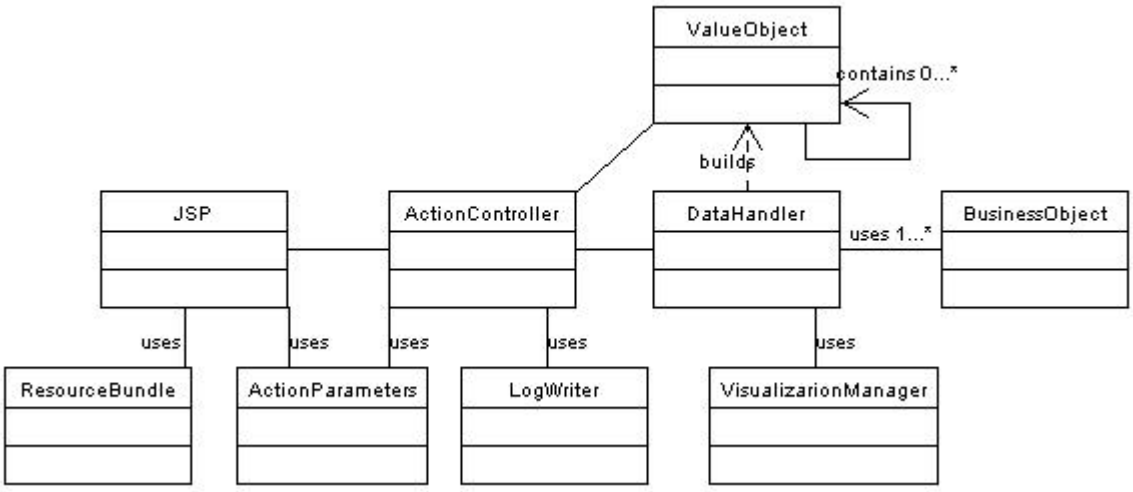
DataHandler -osajärjestelmä huolehtii näytettävän tietosisällön koostamisesta. Se palauttaa halutun tiedon yhdessä rakenteisessa oliossa käyttöliittymäosajärjestelmälle. Kaikki näytettävä tieto on alun perin haettu sopimusvarastojärjestelmästä. Monissa käyttötapauksissa tarvitaan tietoja, joita ei saada suoraan sopimusvarastojärjestelmästä, vaan haluttu tieto tulee koostaa eri kyselyillä saaduista tiedoista. Osajärjestelmä noudattaa arvo-olion koostaja (value object assembler) -suunnittelumallia. DataHandler-osajärjestelmässä on käyttötapauskohtaiset

käsittelijät palvelupyynnöille, sekä näkymäkohtaisen tietosisällön esittävät JavaBean-määrittelyn mukaiset luokat. Näkymäkohtainen tietosisältö koostuu osittain sovelluksen yleisiä tietokomponentteja kuvaavista apu-JavaBean-luokista. DataHandler välittää käyttöliittymälle myös dynaamisesti tuotetut kuvat, jotka muodostetaan Visualizing -osajärjestelmässä.

MessageHandler-osajärjestelmä huolehtii Web Service-rajapinnan kautta tapahtuvasta kommunikaatiosta sopimusvarasto-järjestelmään. Sen julkista rajapintaa käyttää ainoastaan DataHandler.

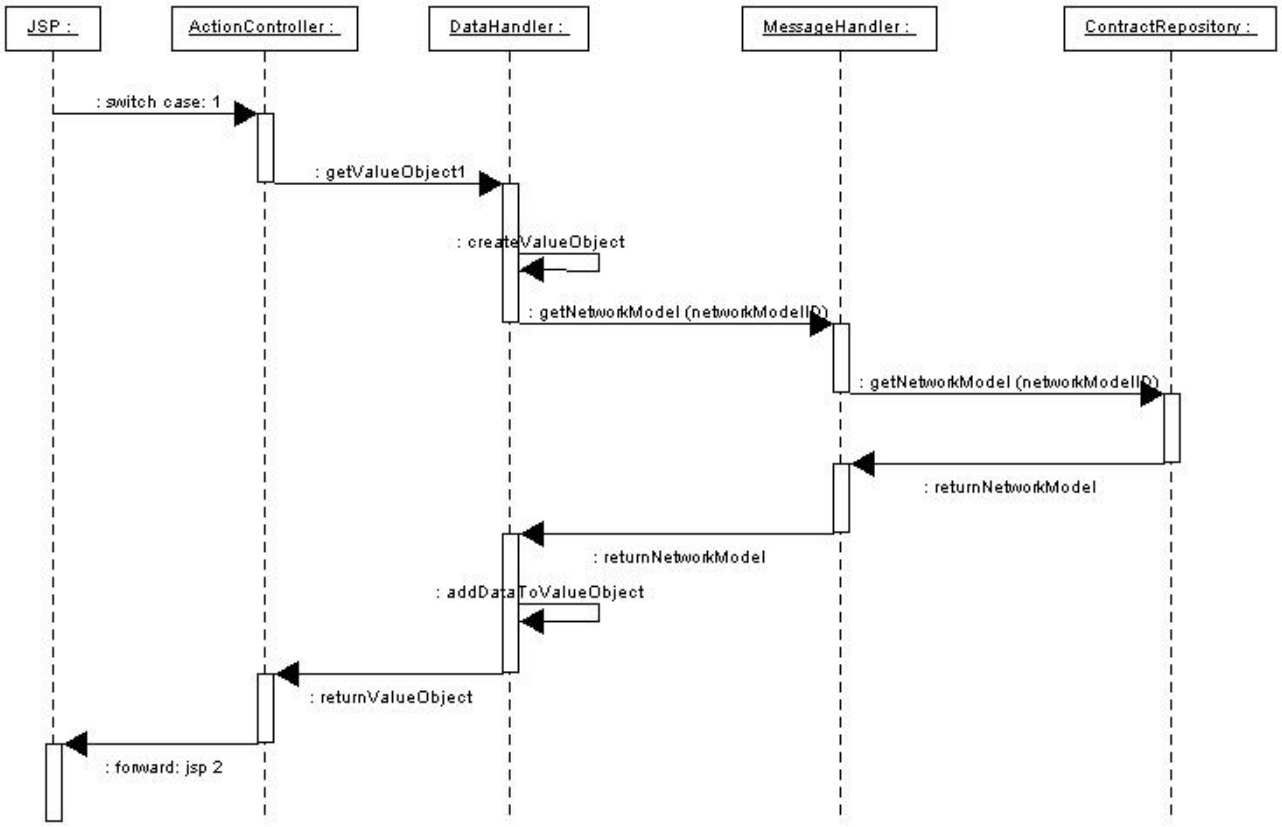
Visualizing -osajärjestelmässä tuotetaan tarvittavat dynaamiset kuvat. Sen julkinen rajapinta tarjoaa metodit verkoston ja automaatin visualisointiin. Osajärjestelmän rajapintaa käyttää DataHandler.

Haba utils -osajärjestelmään kuuluvat muut toteutettavat osat, joita ovat lokitiedoton kirjoittaminen ja asennustietojen lukeminen tiedostosta. Osajärjestelmän julkista rajapintaa käyttävät ActionController keskitettyyn lokin kirjoittamiseen ja MessageHandler sopimusvaraston yhteysparametrin lukemiseen.



Kuva 3: Luokkakaavio järjestelmän keskeisimmistä luokista

2.3 Järjestelmän toiminta



Kuva 4: Osajärjestelmien toiminta tyypillisessä tapauksessa

Kuvassa 4 on esitetty osajärjestelmien toiminta tyypillisessä tapauksessa. Käyttöliittymältä tulee palvelupyyntö DataHandlerille. Tapahtuman käsittelevä Handler -olio pyytää tarvittavan tiedon MessageHandler osajärjestelmältä, joka hakee sen edelleen sopimusvarastojärjestelmästä. Kun kaikki pyyntöön liittyvä tieto on koossa, DataHandler palauttaa sen yhdessä oliossa ActionControllerille, joka palauttaa ne edelleen käyttöliittymälle. Tiedot saatuaan JSP-sivu näyttää ne käyttäjälle. Käyttäjälle näkyvä ulkoasu määräytyy staattisten HTML- ja CSS-tiedostojen mukaan.

3 Käyttöliittymä

Järjestelmän käyttöliittymä on kuvattu käyttötapauskohtaisesti dokumentin liitteenä olevassa Antti Mäen PowerPoint-esityksessä [Mäk04].

Käyttöliittymä muodostuu JSP-sivujen generoimista näytöistä. JSP-sivut ja niiden väliset siirtymät sekä toiminnot ja kentät esitellään kappaleessa 3.2.

3.1 Yleiskuvaus

Järjestelmän visuaalinen käyttöliittymä jakautuu karkeasti kahteen osaan. Hallintapaneelin vasemmassa reunassa on harmaa navigointivalikko ja oikealla tietosisällöltään muuttuva näkymäosa. Osat on esitetty kuvassa 5.

Kuva 5: Käyttöliittymän osat

3.1.1 Navigointivalikko

Navigointivalikon tehtävä on toimia käyttäjän navigaatioväylänä järjestelmän tarjoaman sivuston keskeisimpien näyttöjen välillä. Se myös tarjoaa mahdollisuudet päivittää järjestelmän tiedot sekä poistua järjestelmästä. Navigointivalikko esittää listauksena käyttäjän sopimukset ja niiden yksittäiset sessiot.

Seuraavaksi esitellään navigointivalikon rakenne ylhäältä alaspäin tarkasteltuna.

3.1.1.1 Järjestelmän päivitys (Update)

Päivitysosassa ilmoitetaan viimeisimmän käyttöliittymän tekemän päivitysoperaation ajankohta. Esillä on myös ’Update’-linkki kuvakkeineen. Linkkiä tai kuvaa klikkaamalla järjestelmä päivittää tietonsa ajan tasalle sopimusvarastosta.

3.1.1.2 Etusivu-linkki (Menu)

Linkkiä klikkaamalla järjestelmä palaa käyttöliittymän etusivulle.

3.1.1.3 Lista sopimuksista ja sessioista

Käyttäjän omat sopimukset ja niiden kaikki sessiot esitetään listauksena allekkain. Lisäksi sopimusten tilat esitetään kuvien avulla. Sopimukset ja sessiot on kirjattu linkkeinä, joita klikkaamalla pääsee tarkastelemaan kyseistä sopimusta tai sessiota.

3.1.1.4 Yrityshaku (Search)

Yrityshakukenttään kirjoittamalla ja painamalla viereistä ’Search’-painiketta järjestelmä näyttää kaikki hakuehdoilla löydetyt yritykset sekä niihin liittyvät sopimukset ja sessiot.

3.1.1.5 Lopetus-linkki (Quit)

’Quit’-linkkiä klikkaamalla käyttäjä lopettaa hallintapaneelin käytön.

3.1.2 Näkymäosa

Näkymäosassa esitetään sopimuksiin ja verkostoon liittyvät tiedot. Näkymä vaihtelee tarkasteltavan sivun informaatiosisällön mukaan. Seuraavaksi esitellään karkeasti näkymäosan rakenne ylhäältä alaspäin tarkasteltuna.

3.1.2.1 Otsikkorivit

Pääotsikkona esitetään tarkasteltavan kohteen nimi, ID tai muu sivuun keskeisesti liittyvä esittelytieto. Sopimuksen alaisia tietoja tarkasteltaessa pääotsikon alla on lueteltu alaotsikkona tarkasteltavan kohteen sijainti sopimuksen, sessioiden ja rooliprosessien muodostamassa puurakenteessa. Alaotsikon kohteet ovat linkkejä, joita klikkaamalla pääsee siirtymään kyseiseen osaan puurakennetta.

3.1.2.2 Tietosisältö

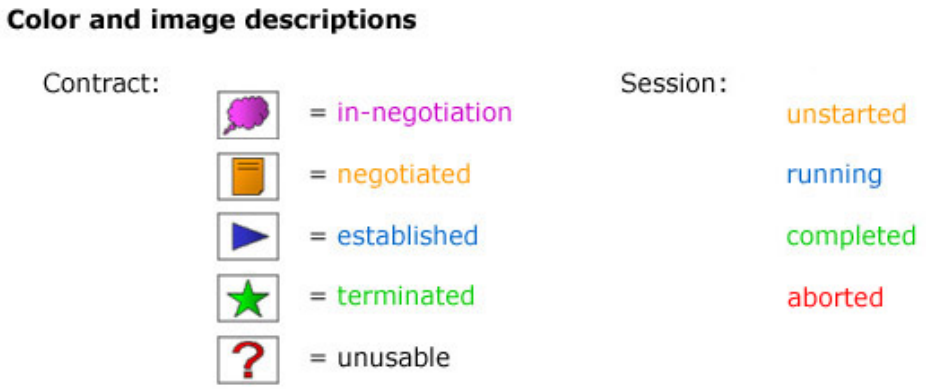
Otsikkojen alla näytetään kohteen tietosisältö. Kaikki alleviivatut tekstit ovat klikattavissa ja siirtävät tarkastelemaan kyseisen kohteen tietoja.

3.1.2.3 Kuvaosa

Useilla järjestelmän sivuilla tietosisällön alapuolella on esillä joko kyseiseen rooliin liittyvä rooliprosessin tila-automaatin esitys tai linkki verkoston arkkitehtuurimalliin. Etusivulla on lisäksi esillä kuva, jossa esitellään sopimuksiin ja sessioihin liittyvät värikoodit ja kuvat.

3.1.3 Värien ja kuvien selitykset

Järjestelmä käyttää eri värejä selkeyttämään sopimusten ja sessioiden tilaa nykyisellä ajan hetkellä. Lisäksi käyttöliittymässä on esillä sopimusten yhteydessä kuvallinen selostus sopimuksen tilasta. Kuvassa 6 esitellään sopimuksen tilojen kuvat ja värikoodit sekä sessioiden värikoodit.



Kuva 6: kuva- ja värikoodit

Sopimuksen tila jakautuu normaalitilanteessa neljään vaiheeseen, joista vielä neuvotteluvaiheessa oleva sopimus on esitetty violetilla värillä. Ennen suorituksen alkamista väri on keltainen, suorituksen aikana sininen ja suorituksen päätyttyä vihreä. Samat kolme väriä – keltainen, sininen ja vihreä – toistuvat myös sessioiden etenemisen kuvaajina: keltainen on vielä suorittamatta, sininen suorituksessa parhaillaan ja vihreä suoritettu.

Sopimuksen poikkeustilaa kuvaa musta väri verkon ollessa väliaikaisesti pois käytöstä. Sessioissa poikkeusta edustaa punainen väri toiminnan päätyttyä käyttäjän osalta kesken. Huomattava on, että sopimusta tarkasteltaessa vihreä väri kuvaa kaikkia käyttäjän kannalta päättyneitä sopimuksia – sekä keskeneräisiä että valmiita.

Jokaista sopimuksen tilaa esittää lisäksi kuva, joka on esillä sekä navigointivalikossa sopimuksen kohdalla että kyseistä sopimusta tarkasteltaessa. Kuva vaihtuu sopimuksen tilan muuttuessa.

3.2 JSP-sivut

JSP-sivut on kuvattu esittelemällä ensin kaikille sivuille yhteiset yleiset ominaisuudet luvussa 3.2.1. JSP-sivukohtainen käyttöliittymän kenttien kuvaus parametritietoineen esitellään luvussa 3.2.2.

3.2.1 Yleiset ominaisuudet

Jokaisella sivulla määritellään JSP:n error page -tagilla käytettäväksi yleinen controlpanelerror.jsp -virhesivu. Sivu reagoi kaikkiin käyttöliittymätasolle päätyneisiin poikkeuksiin. Sivu esittää virheestä sen kuvauksen sekä siihen liittyvän jäljityspinon. Käytännössä järjestelmässä ei tapahdu poikkeuksia, joista voitaisiin toipua, joten kaikki tapahtuneet poikkeukset esitetään virhesivulla käyttöliittymässä. Lisäksi ActionController kirjoittaa keskitetysti virhetiedon lokitiedostoon. Controlpanelerror.jsp tulkitsee onko kyseessä JSP-sivulla tapahtunut ajonaikainen poikkeus vai jo ennen jonkin näkymän kutsua tapahtunut poikkeus, jolloin virhesivua on kutsuttu eksplisiittisesti ActionControllerista.

```
<%@ page language="java" errorPage="controlpanelerror.jsp" %>
```

Käyttöliittymässä esitettävät staattiset tekstimuotoiset otsikkokentät on lokalisoitu termistön muuttumisen varalta Javan ResourceReader-luokan avulla. Kenttien arvot saadaan MyResources.properties tiedostosta:

```
<%@ page import="java.util.*;"%>
<% ResourceReader myResources =  new ResourceReader(); %>

<h1><%=myResources.getString("existingContractsTitle") %></h1>
```

Syy ResourceReaderin käyttöön ResourceBundlen sijasta on Tomcat-palvelimen huomattu epävakaa toiminta ResourceBundleja käsiteltäessä. ResourceReader saattaa joissain tilanteissa häiritä pullonkaulana järjestelmän jouhevaa toimintaa. Tarvittaessa vaihtaminen ResourceBundleihin on helppoa.

Sivuvalikon HTML:n generoinnista vastaa jokaiselle JSP-sivulle sisällytettävä sidemenu.jsp:

```
<body bgcolor="#FFFFFF" text="#000000" link="#000000" vlink="#000000">
<table width='100%' cellpadding='5'>
<tr>
<td width='350' valign='top'>
                <jsp:include page="sidemenu.jsp" flush="false" />
</td>
<td align='left'>
                Varsinainen sivun sisältö.
</td>
</tr>
</table>
</body>
```

Sivuihin liittyvät tyylit on määritelty styles.css -tyylitiedostossa.

3.2.2 Kentät

JSP-sivut tarjoavat kahta tyypillistä käyttöliittymän toiminnallisuutta:

- Esittävät järjestelmän tilaa kuvaavaa tietoa
- Tarjoavat linkkejä ja painikkeita näkymään liittyvien toimintojen suorittamiseksi.

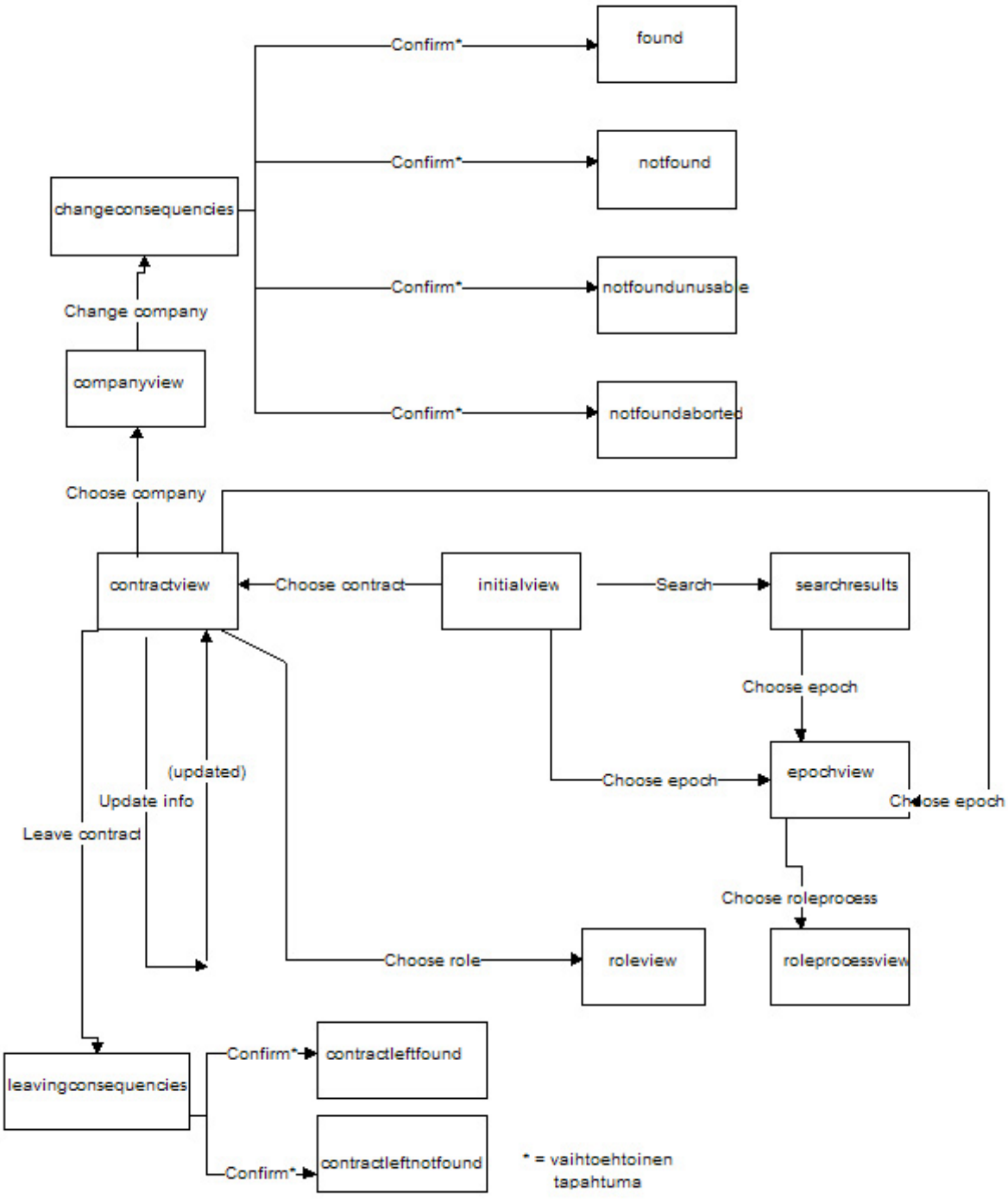
Esitettävät tiedot luetaan HTML-koodin sekaan kutsumalla näkymään liittyvien JavaBean-luokkien getter-metodeja. Kyseiset metodit voivat palauttaa esitettävän kentän String-tyyppisen arvon sellaisenaan, tai ne voivat muotoilla jonkin rakenteisen HTML-tagin valmiiksi JSP-sivua varten. JSP-sivun ulkopuolella tapahtuva muotoilu auttaa erottamaan varsinaisen HTML-koodin ja ohjelmalogiikan toisistaan.

Esimerkki sopimuksen valintaan liittyvän HTML-muotoisen toimintolinkin muodostamisesta JSP-sivulla:

```
<%=contractBean.getHtmlLink()%>
```

Kunkin toiminnon identifioi action-niminen parametri, jonka arvo on jokin ActionParameters-luokan int-arvoista. Näkymään liittyvien toimintojen parametrit on esitetty seuraavassa luvussa näkymäkohtaisissa taulukoissa. Samoissa taulukoissa on kuvattu mitä näkymää esittävää JavaBean-luokan kenttää kukin käyttöliittymän kenttä vastaa.

JSP-sivujen nimeäminen ja kuhunkin toimintoon liittyvä seuraavaksi kutsuttava sivu käy ilmi oheisesta näyttötilakaaviosta (Kuva 7).



Kuva 7: Näyttötilakaavio

Sidemenu.jsp

Sidemenu.jsp sisällytetään jokaiseen näkymään JSP-sivukohtaisesti include page –tagilla. Kuvassa 8 on esitetty valikon ulkoasu ja käyttöliittymän kentät. Toimintoihin liittyvät parametrit on lueteltu taulukossa 1.

Kuva 8: Navigointivalikko

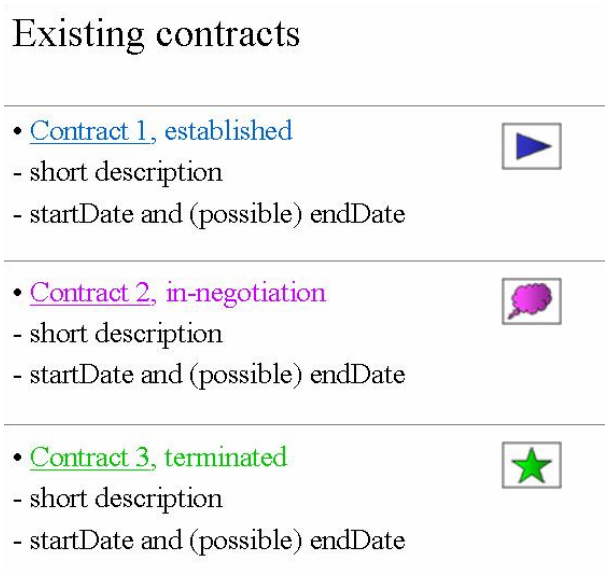
Kenttä	Kentän sisältö	Toimintoparametri (action)	Toiminnon parametrit	Huomautukset
Update-linkki		initiate_session		Tämä suoritetaan myös esim. järjestelmän käynnistyksen yhteydessä.
Contract-linkki	NavigationMenuBean.ContractBean.getId()	choose_contract	contract_id (ContractContent.contractID)	

Session-linkki	NavigationMenuBean.ContractBean.ContractSessionBean.getId()	choose_session	session_id (ContractSession.sessionID)	
Search for a company-tekstikenttä	-	search_company	company_identifier	Tämän parametrin arvo on osa yrityksen nimestä.
Quit-linkki				Linkki vie suoraan aloitussivulle. Tähän ei liity toimintoja.

Taulukko 1: Navigointivalikon parametrit

Initialview.jsp

Initialview –näkymässä esitellään yrityksen sopimukset. Kuvassa 9 on esitetty näkymän ulkoasu ja käyttöliittymän kentät. Toimintoihin liittyvät parametrit on lueteltu taulukossa 2.



Kuva 9: Perusnäyttö

Kenttä	**Kentän sisältö**	**Toimintoparametri (action)**	**Toiminnon parametrit**	**Huomautukset**
Contract-linkki	InitialViewBean.ContractBean.getHtmlLink()	choose_contract	contract_id (ContractContent.contractID)	
State (esim. established)	InitialViewBean.ContractBean.getStateString()			
short description	InitialViewBean.ContractBean.getDescription()			

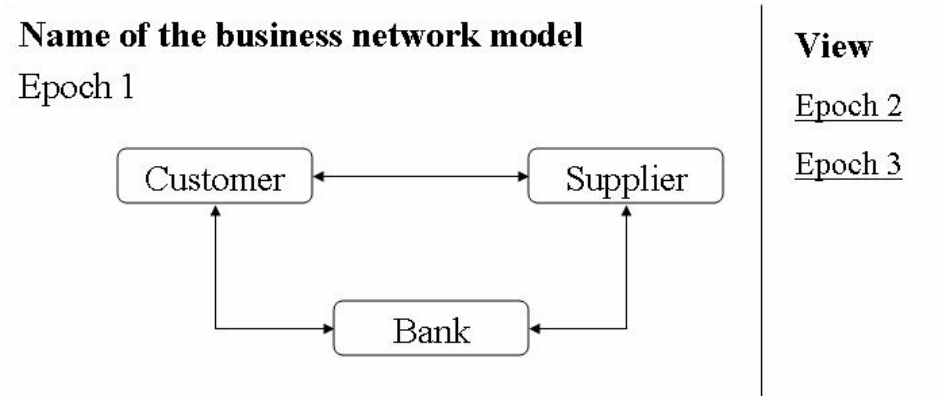
startDate	InitialViewBean.ContractBean.getHtmlStartState()			
endDate	InitialViewBean.ContractBean.getHtmlEndState()			

Taulukko 2: Perusnäytön parametrit

Networkimage.jsp

Networkimage –kuvanäkymä liittyy osana useisiin näkymiin, joihin se liitetään omana kehyksenä (HTML, IFrame). Esitettävään kuvaan liittyvät epokkiparametrit vaikuttavat kussakin näkymässä vain kuvaan. Tästä syystä kuvan esittäminen omassa muusta näkymästä eristetyssä kehyksessä on johdonmukaista. Kuvan perusrakenne ja siihen vaikuttavat toimintokentät on esitetty oheisessä kuvassa 10.

Kuvanäkymä on esitetty osana muita näkymiä, joihin kuva kuuluu. Sen toiminnot ja sisältö on kuitenkin kuvattu vain kertaalleen tässä luvussa.



Kuva 10: Verkoston rakennetta esittävä kuva

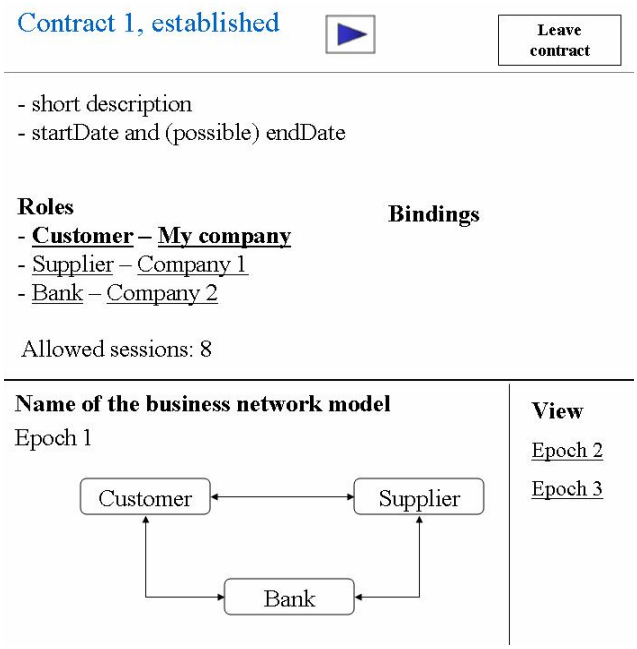
Kuvan kenttien ja toimintojen parametrit on esitetty taulukossa 3.

Kenttä	Kentän sisältö	Toimintoparametri (action)	Toiminnon parametrit	Huomautukset
Name of the business network model	NetworkImageBean.getName()			
Verkoston visualisointikuva	NetworkImageBean.getImageURL()			
Epoch-linkki	NetworkImageBean.getEpochHtmlLink()	choose_image_epoch	epoch_id	

Taulukko 3: Sivun parametrit

Contractview.jsp

Contractview–näkymässä esitellään yrityksen tietyn sopimuksen tiedot. Kuva 11 esittää näkymän ulkoasua ja käyttöliittymän kenttiä. Toimintoihin liittyvät parametrit on lueteltu taulukossa 4.



Kuva 11: Sopimuksen tiedot -näyttö

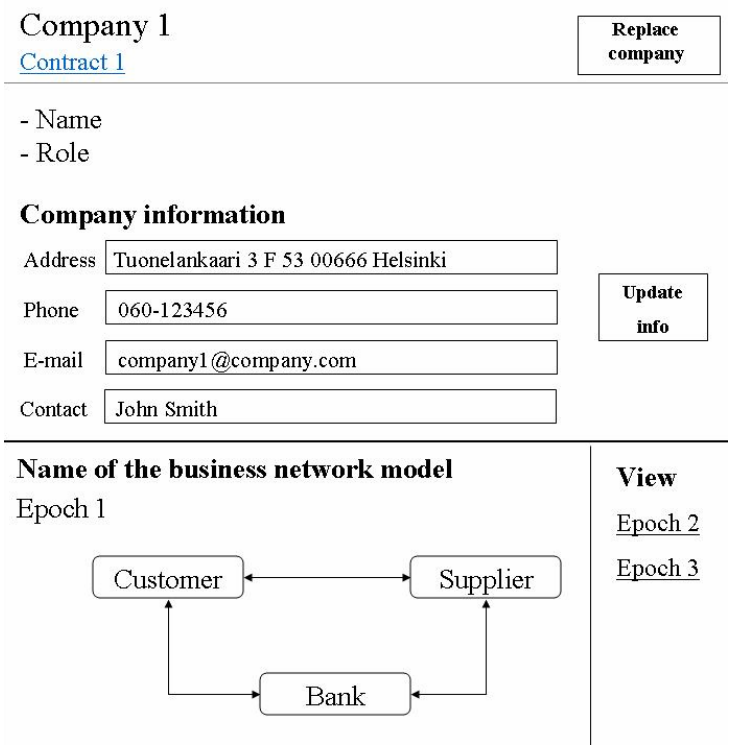
Kenttä	Kentän sisältö	Toimintoparametri (action)	Toiminnon parametrit	Huomautukset
Leave contract -painike		leave_contract	contract_id (ContractContent.contractID)	
short description	ContractViewBean.ContractBean.getDescription()			

startDate	ContractViewBean.ContractBean.getHtmlStartDate()			
endDate	ContractViewBean.ContractBean.getHtmlEndDate()			
Roles (customer-sarake)	ContractViewBean.ContractBean.getContractRolesAndParticipants[x][]	choose_role	role_name (ParticipantInfo.role)	
Roles (My company - sarake)	ContractViewBean.ContractBean.getContractRolesAndParticipants[][x]	choose_company	company_name (ParticipantInfo.name)	
Channels (=Bindings)				Tämä tieto esitetään alalaidan kuvassa.
Allowed sessions	ContractViewBean.ContractBean.getAllowedSessions()			

Taulukko 4: Sopimuksen tiedot –näytön parametrit

Companyview.jsp

Companyview–näkymässä esitetään ja tarjotaan editoitaviksi yrityksen yhteystiedot. Kuvassa 12 esitellään näkymän ulkoasu ja siihen liittyvät käyttöliittymän kentät. Toimintoihin liittyvät parametrit on lueteltu taulukossa 5.



Kuva 12: Yrityksen tiedot -näyttö

Kenttä	Kentän sisältö	Toimintoparametri (action)	Toiminnon parametrit	Huomautukset
Contract-linkki	CompanyViewBean.ContractBean.getHtmlLink()	choose_contract	contract_id (ContractContent.contractID)	
Replece company –painike		change_company	participant_signature (ParticipantInfo.signature)	

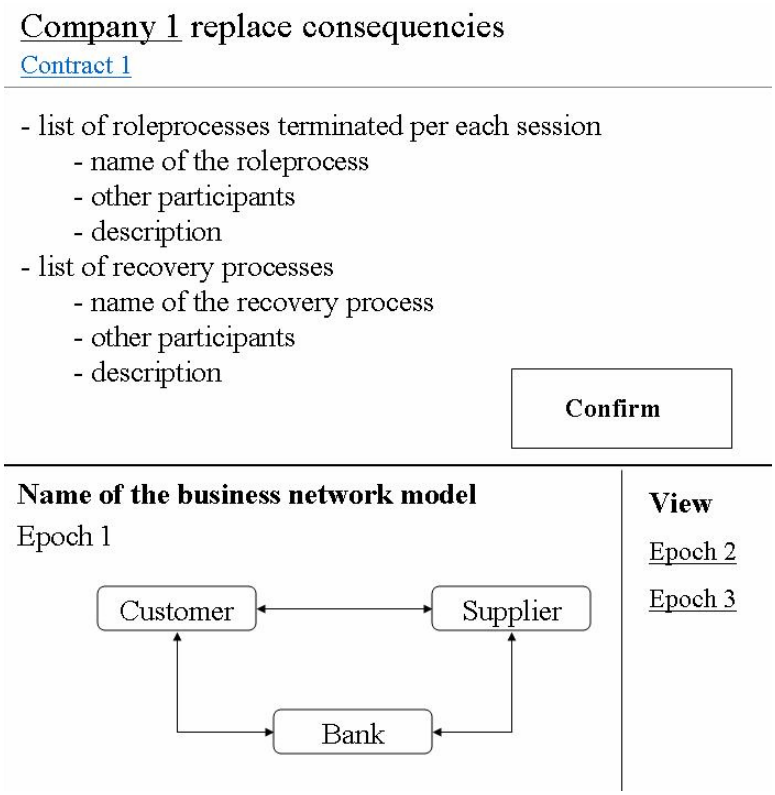
Name	CompanyViewBean.getName()			
Role	CompanyViewBean.getRole()			
Address	CompanyViewBean.getAddress()			
Phone	CompanyViewBean.getPhone()			
E-mail	CompanyViewBean. getEmail()			
Contact	CompanyViewBean. getContactPerson()			

Update info - painike		update_info	company_name (ParticipantInfo.name), contact_address (ParticipantInfo.address), phone ParticipantInfo.phone, contact_mail (ParticipantInfo.contactMail), contact_person (ParticipantInfo.contactPerson)	

Taulukko 5: Yrityksen tiedot –näytön parametrit

Changeconsequencies.jsp

Changeconsequencies –näkymässä listataan tiedot seurauksista, jotka seuraavat liiketoimintaverkon toimijan vaihtamisesta.. Kuvassa 13 on esitetty näkymän ulkoasu ja käyttöliittymän kentät. Toimintoihin liittyvät parametrit on lueteltu taulukossa 6.



Kuva 13: Osallistujan vaihtamisen seuraukset -näyttö

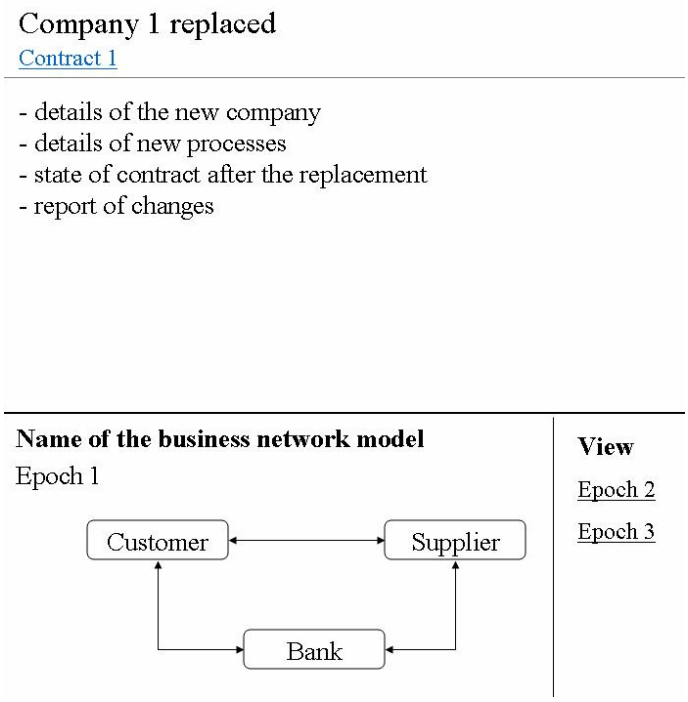
Kenttä	Kentän sisältö	Toimintoparametri (action)	Toiminnon parametrit	Huomautukset
Company-linkki	ChangeConsequenciesBean.getCompanyName()	choose_company	company_name (ParticipantInfo.name)	
Contract-linkki	ChangeConsequenciesBean.ContractBean.getHtmlLink()	choose_contract	contract_id (ContractContent.contractID)	

name of role	ChangeConsequenciesBean.RecoveryProcessBean.getRoleName()			
session ID	ChangeConsequenciesBean.RecoveryProcessBean.getSessionID()			
conversation ID	ChangeConsequenciesBean.RecoveryProcessBean.getConversationID()			
affected process name	ChangeConsequenciesBean.RecoveryProcessBean.getAffectedProcesses()			Näitä voi olla useita.
name of the recovery process	ChangeConsequenciesBean.RecoveryProcessBean.getRecoveryProcessName()			
Confirm-painike		confirm_change_company	participant_signature (ParticipantInfo.signature)	

Taulukko 6: Sivun parametrit

changereport.jsp

Changereport –näkymä liittyy kiinteästi Changeconsequencies –näkymään. Tässä esitellään vahvistetun toimijan vaihtamisen seuraukset. Kuvassa 14 esitellään ulkoasu ja käyttöliittymän kentät. Taulukkoon 7 on listattu näkymän toimintojen parametrit.



Kuva 14: Raportti osallistujan vaihtamisesta

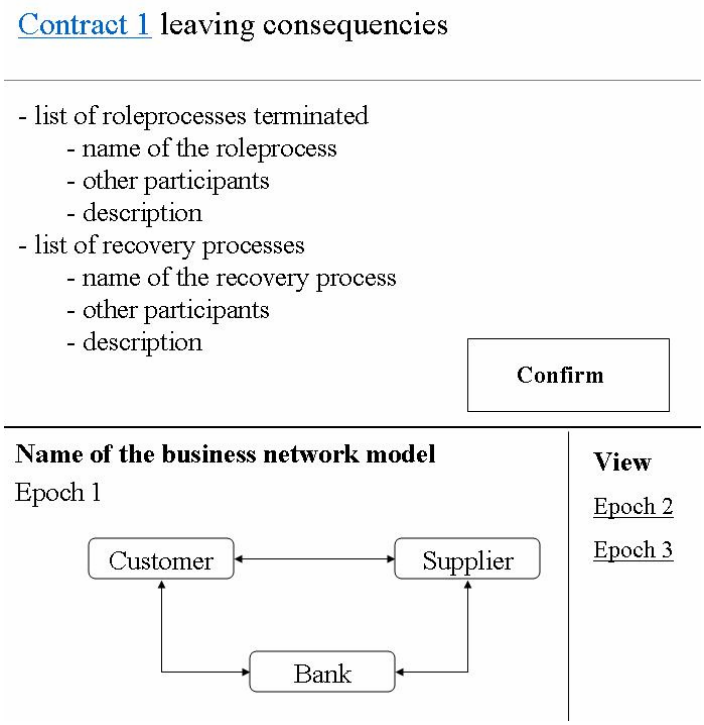
Kenttä	Parametrin nimi tietoa esitettäessä	Toimintoparametri (action)	Toiminnon parametrit	Huomautukset
Company name	ChangeConfirmationBean.ContractBean.getCompanyName()			

name of the role	ChangeConfirmationBean.ContractBean.getContractRolesAndParticipants[x][]			.
name of the company	ChangeConfirmationBean.ContractBean.getContractRolesAndParticipants[][x]			
state of contract after the replacement	ChangeConfirmationBean.getHtmlStateReport()			

Taulukko 7: Sivun parametrit

Leavingconsequencies.jsp

Näkymän tietosisältö vastaa pitkälle Changeconsequencies –näkymän tietosisältöä. Tässä esitellään seuraukset, jotka seuraavat hallintapaneelia parhaillaan käyttävän yrityksen poistumisesta sopimuksesta. Ulkoasu ja kentät on esitetty kuvassa 15.



Kuva 15: Sopimuksesta lähtemisen seuraukset -näyttö

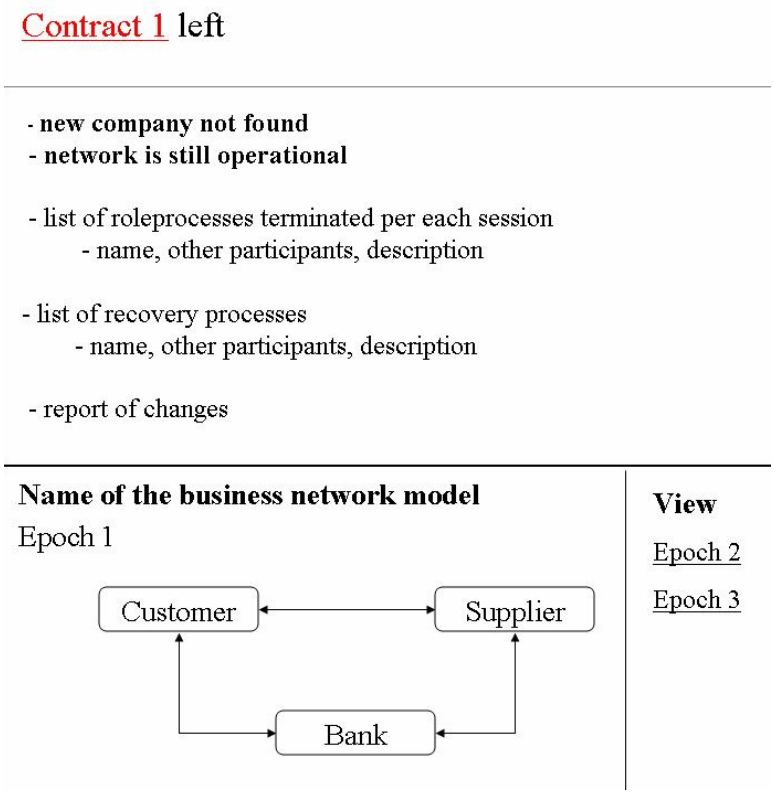
Näkymän tietosisältö on lähes sama kuin changeconsequencies.jsp-sivulla. Vain Confirm-painikkeen parametrit eroavat näkymien välillä. Ne on kuvattu taulukossa 8.

Kentä	**Parametrin nimi tietoa esitettäessä**	**Toimintoparametri (action)**	**Toiminnon parametrit**	**Huomautukset**
Confirm-painike		confirm_leave_contract	participant_signature (ParticipantInfo.signature)	

Taulukko 8: Sivun parametrit

Leavingreport.jsp

Leavingreport –näkymässä on kuvattu yrityksen sopimuksesta poistumisen vahvistus siihen liittyvine seurauksineen. Näkymän ulkoasu ja kentät on esitetty kuvassa 16.



Kuva 16: Raportti sopimuksesta lähtemisestä

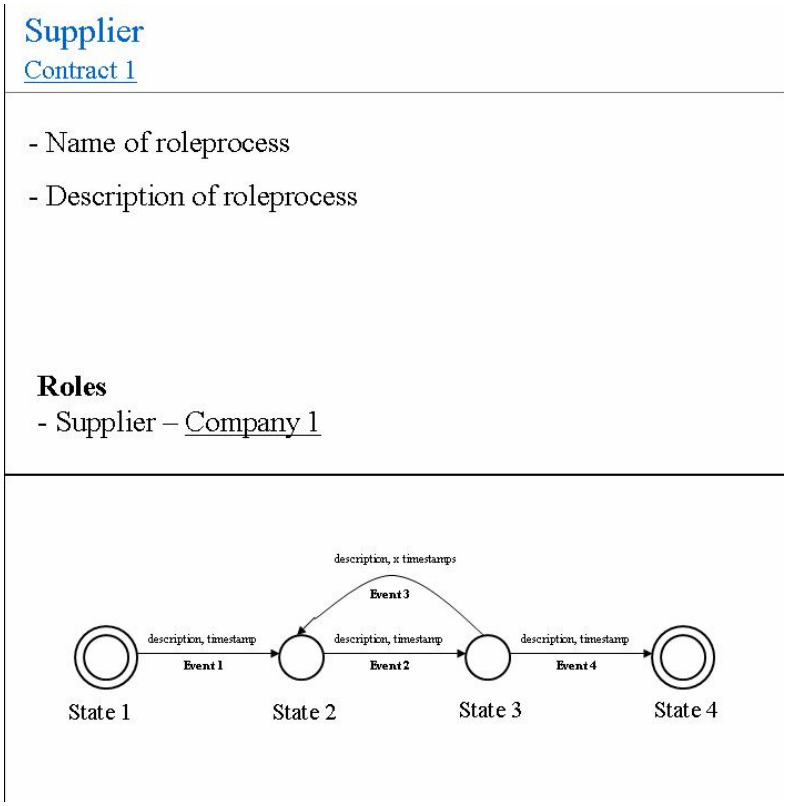
Näkymän tietosisältö on pääosin sama kuin changereport.jsp -sivulla. Changereport-näkymästä poikkeavat osat on listattu oheisessa taulukossa 9.

Kenttä	Parametrin nimi tietoa esitettäessä	Toimintoparametri (action)	Toiminnon parametrit	Huomautukset
selitys (new company not found…)	report_text			Tämä tieto kulkee yhdessä String-muuttujassa.

Taulukko 9: Sivun parametrit

Roleviewmin.jsp

Roleviewmin –näkymässä esitellään sopimukseen liittyvän roolin tiedot. Ulkoasu kenttineen on kuvattu kuvassa 17. Roolinäkymän parametrit on lueteltu taulukossa 10.



Kuva 17: Roolinäkymä

Kenttä	Parametrin nimi tietoa esitettäessä	Toimintoparametri (action)	Toiminnon parametrit	Huomautukset
Supplier	RoleProcessViewBean.getRoleName()			
Contract-linkki	RoleProcessViewBean.ContractBean.getHtmlLink()	choose_contract	contract_id (ContractContent.contractID)	

Session -linkki	RoleProcessViewBean.ContractSessionBean.getHtmlLink()	choose_session	session_id (ContractSession.sessionID)	Olemassa vain jos roolia tarkastellaan sessiotietojen kautta
Name of roleprocess	RoleProcessViewBean.getRoleProcessName()			
Description of roleprocess	RoleProcessViewBean.getRoleProcessDescription()			
Rooliprosessin kuva Aikaleimat Transitions	RoleProcessViewBean.getAutomataURL() RoleProcessViewBean.getTimeStamps() RoleProcessViewBean.getTrasitions()			

Taulukko 10: Sivun parametrit

Searchresults.jsp

Searchresults –näkymä liittyy Sidemenu –näkymän Search company –kenttään. Searchresults –näkymässä listataan hakuaehtoa vastaavat yritykset sopimuksineen ja niihin liittyvine sessioineen. Ulkoasu ja mainitut kentät on esitetty kuvassa 18. Näkymän parametrit on listattu taulukossa 11.

Kuva 18: Yrityshaun tulokset

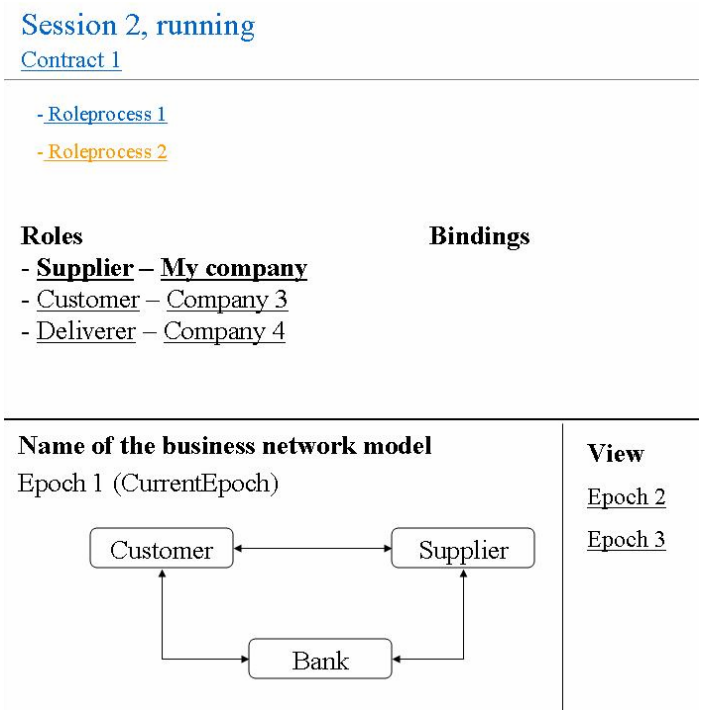
Kentä	Parametrin nimi tietoa esitettäessä	Toimintoparametri (action)	Toiminnon parametrit	Huomautukset
Company-otsikko	SearchResultsBean.CompanyBean.getName()			
Contract-linkki	SearchResultsBean.CompanyBean.ContractBean.getHtmlLink()	choose_contract	contract_id (ContractContent.contractID)	

Session-linkki	ContractSessionBean.getHtmlLink()	choose_session	session_id (ContractSession.sessionID)	

Taulukko 11: Sivun parametrit

Sessionview.jsp

Sessionview –näkymässä esitellään sopimuksen valittuun sessioon liittyvät tiedot. Käyttöliittymän ulkoasu ja kentät on esitetty kuvassa 19. Toimintoihin liittyvät parametrit on listattu taulukossa 12.



Kuva 19: Sessionäyttö

Kenttä	Parametrin nimi tietoa esitettäessä	Toimintoparametri (action)	Toiminnon parametrit	Huomautukset
Session	SessionViewBean.ContractSessionBean.getSessionID()			

Tila (esim. running)	SessionViewBean.ContractSessionBean.getSessionState()			
Contract-linkki	SessionViewBean.ContractSessionBean.getContractID()	choose_contract	contract_id (ContractContent.contractID)	
Roles (supplier-sarake)	SessionViewBean.ContractSessionBean.getSessionRoles()	choose_role	role_name (ParticipantInfo.role)	

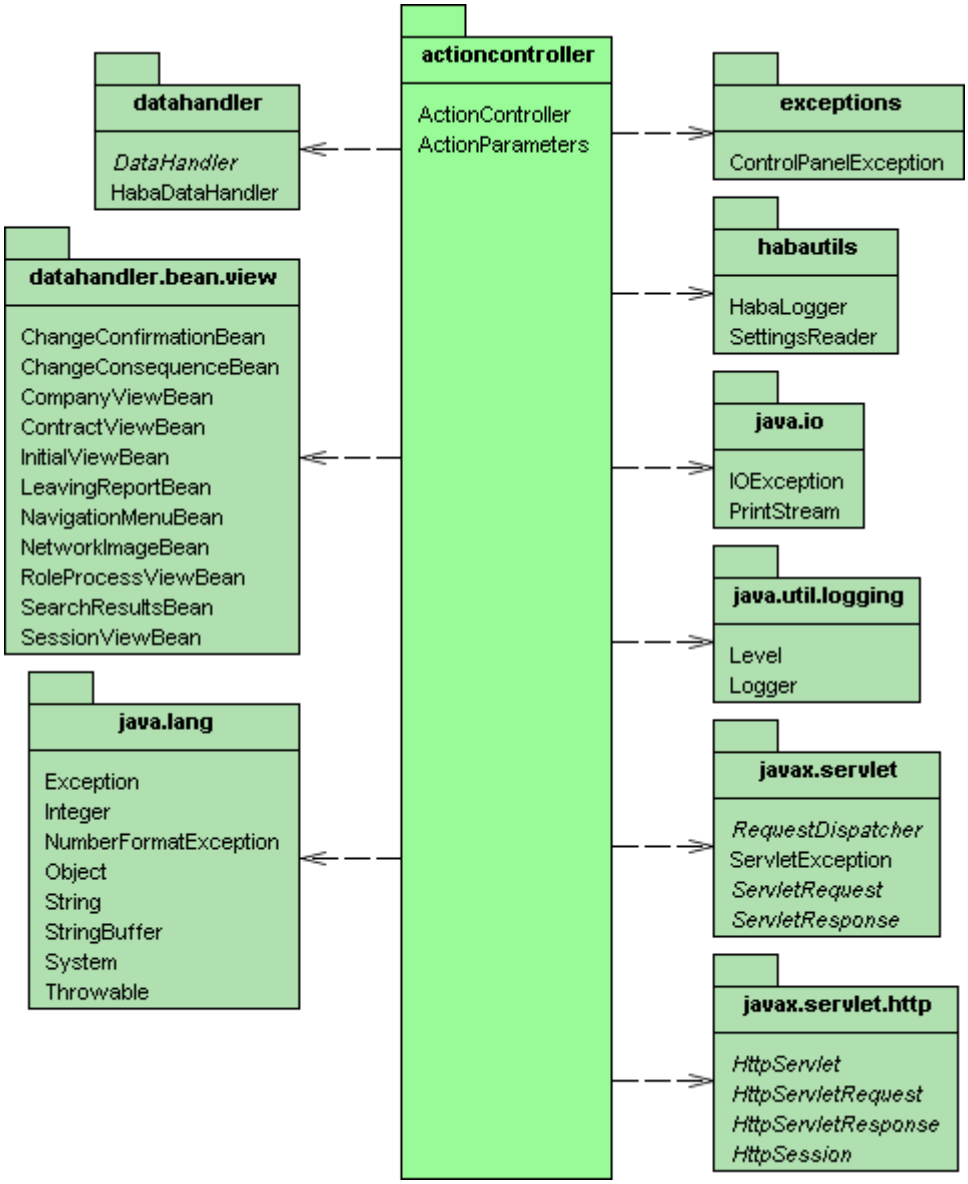
Taulukko 12: Sivun parametrit

4 Luokat ja rajapinnat

Tässä luvussa esitellään toteutettavan järjestelmän sisäiset rajapinnat ja niihin liittyvät pakkaukset ja keskeiset luokat. Pakkausten väliset yhteydet liittyvät sovelluksen arkkitehtuuriin ja ne on kuvattu tämän dokumentin arkkitehtuuriluvussa.

4.1 Actioncontroller-pakkaus

Pakkaus sisältää luokat, joita käytetään käyttöliittymän JSP-sivuilta saapuvien HTTP-kutsujen tulkintaan. Kuvassa 20 esitetään actioncontroller-pakkauksen yhteydet luokkakaaviona. Pakkaukseen liittyvä keskeinen toiminta on kuvattu tarkemmin ActionController-luokkaa käsittelevässä luvussa.



Kuva 20 : actioncontroller-pakkauksen luokkakaavio

4.1.1 ActionParameters

ActionParameters-luokka sisältää listauksen kaikista järjestelmän toimintoparametreista. Parametrit vastaavat tämän dokumentin JSP-sivukohtaisissa kuvauksissa lueteltuja toimintoja. JSP-sivut ja ActionController-luokka käyttävät ActionParameters-luokkaa järjestelmän toimintojen indeksointiin. JSP-sivut käyttävät luokkaa linkkien parametrien generoimiseen.

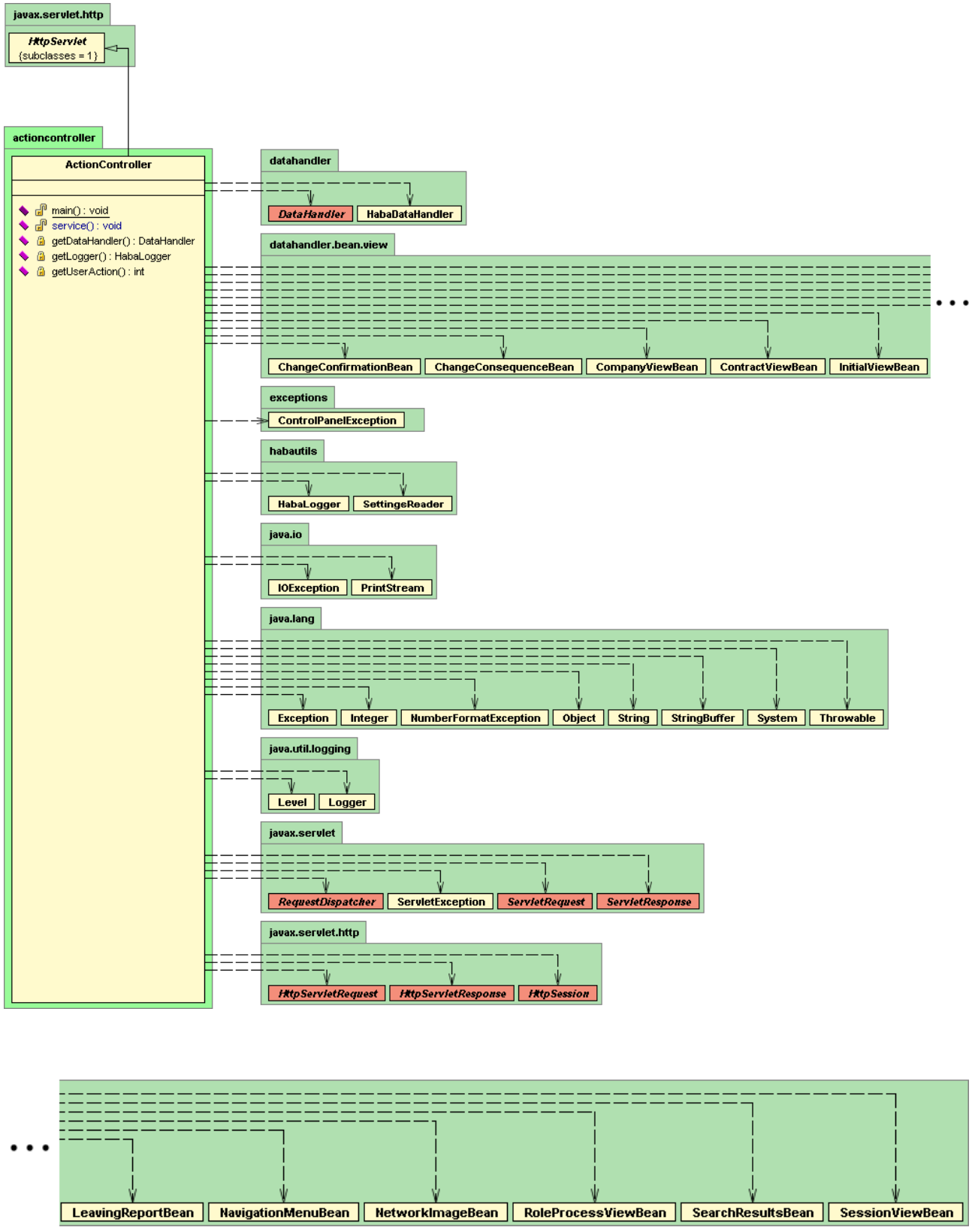
4.1.2 ActionController

ActionController on Java Servlet –luokka. Se ottaa vastaan HTML-käyttöliittymän HTTP -kutsut ja tulkitsee ActionParameters-luokan avulla indeksoiden mitä DataHandler-rajapinnan metodia kunkin järjestelmän toiminnon yhteydessä kutsutaan.

ActionController hoitaa kaikkien järjestelmävirheiden käsittelyn välittämällä virhetiedon HabaLogger-luokalle lokin kirjoittamista varten. Virheen tapahtuessa ActionController välittää virheluokan tulkittavaksi controlpanelerror.jsp- sivulla.

ActionController saa HabaDataHandler-luokalta paluuarvona kutakin tapahtumaa seuraavan näytön tarvitsemat tiedot näkymäkohtaisessa luokassa (XxxViewBean). ActionController asettaa paluuarvona saadun luokan HttpRequest-olioon ja välittää tämän näyttötilakaavion (kuva 7) mukaiselle JSP-sivulle.

Kuvassa 21 esitetään ActionController-luokan sisältö sekä sen käyttämät luokat. Muista luokista ActionControllerin toiminnalle keskeisiä ovat HabaDataHandler-luokan palauttamat näkymäkohtaiset Bean-luokat.

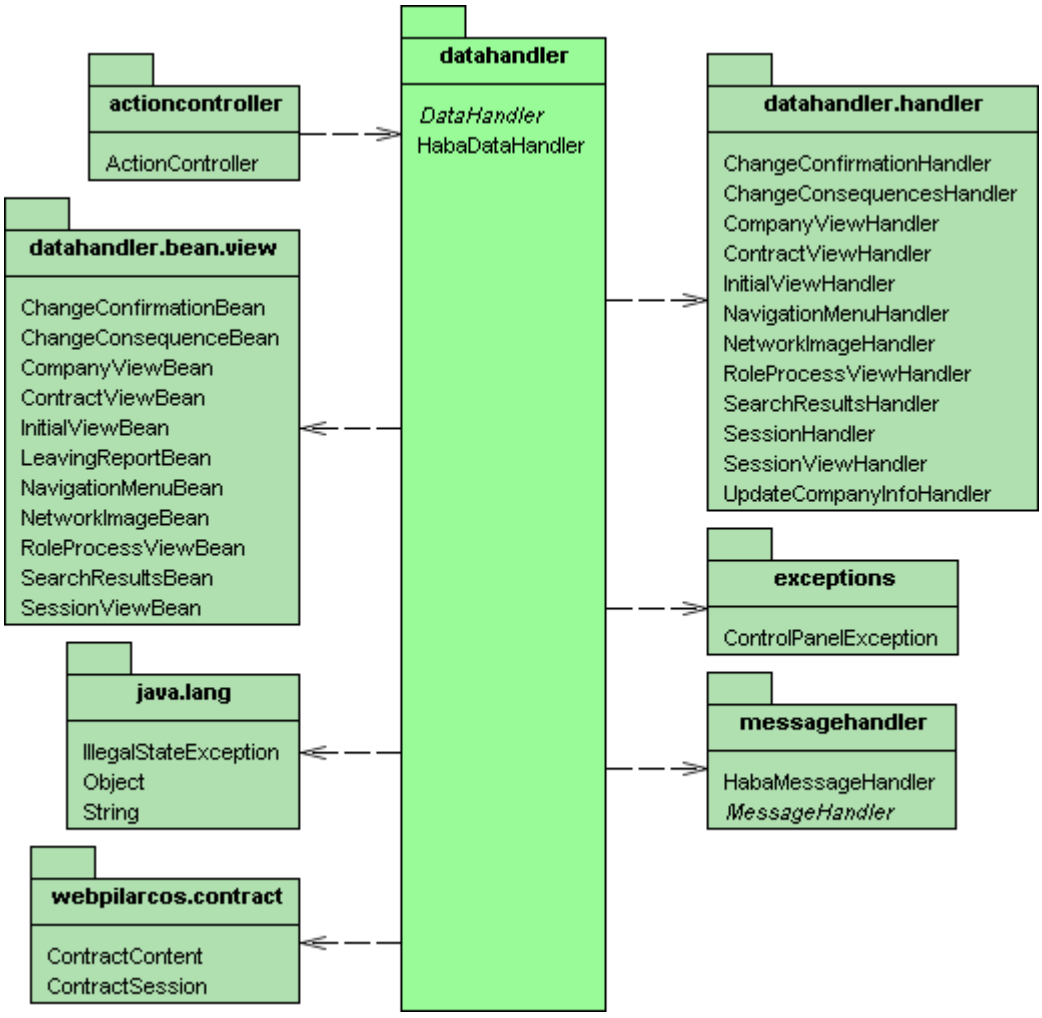


Kuva 21 : ActionController-luokan luokkakaavio

4.2 Datahandler-pakkaus

Datahandler-pakkaukseen kuuluvat DataHandler-rajapinta sekä sen toteuttava HabaDataHandler-luokka. Näkymäkohtaisen tietosisällön sisältävät luokat ovat pakkauksessa datahandler.bean.view. Jotkin näkymäkohtaiset bean-luokat sisältävät lisäksi pienempiä useissa näkymissä toistuvia tietokokonaisuuksia esittäviä datahandler.bean.helper –pakkauksen luokkia. Kunkin näkymän tietosisällön koostamisesta bean –luokkaan vastaa näkymäkohtainen datahandler.handler –pakkauksen luokka.

Kuvassa 22 esitetään datahandler-pakkauksen yhteydet luokkakaaviona.



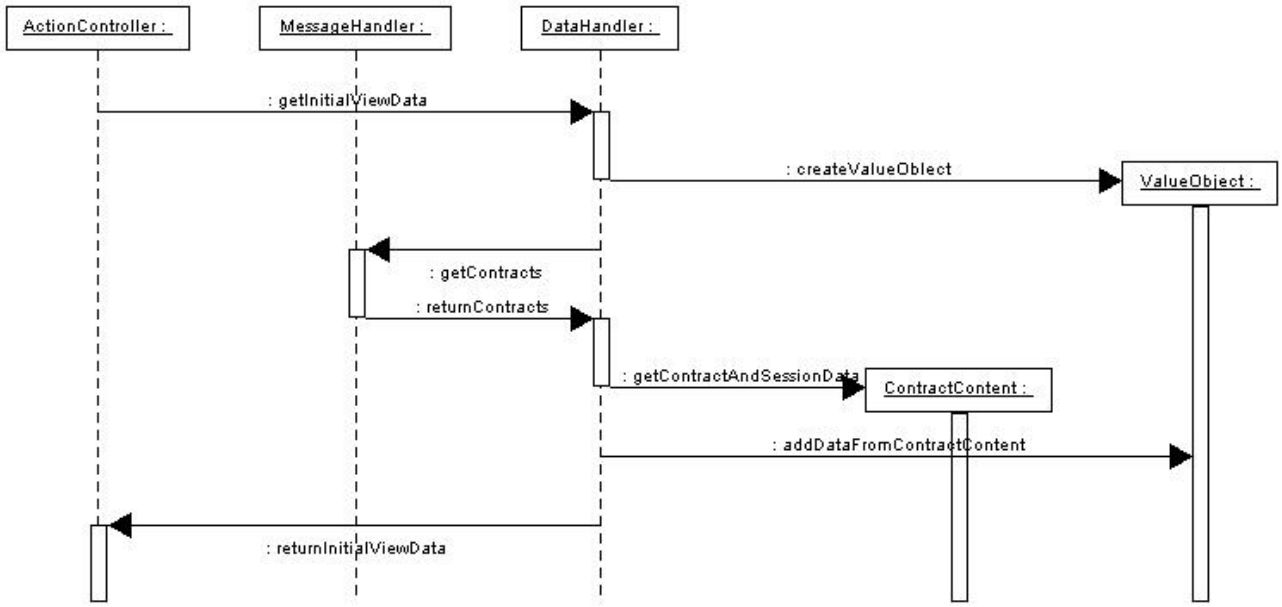
Kuva 22 : datahandler-pakkauksen luokkakaavio

4.2.1 HabaDataHandler.java

HabaDataHandler toimii välimuistina kulloinkin käyttöliittymässä käsiteltävänä olevalle sopimukselle ja sessiolle. Tätä ominaisuutta käytetään joidenkin järjestelmän toimintojen kohdalla.

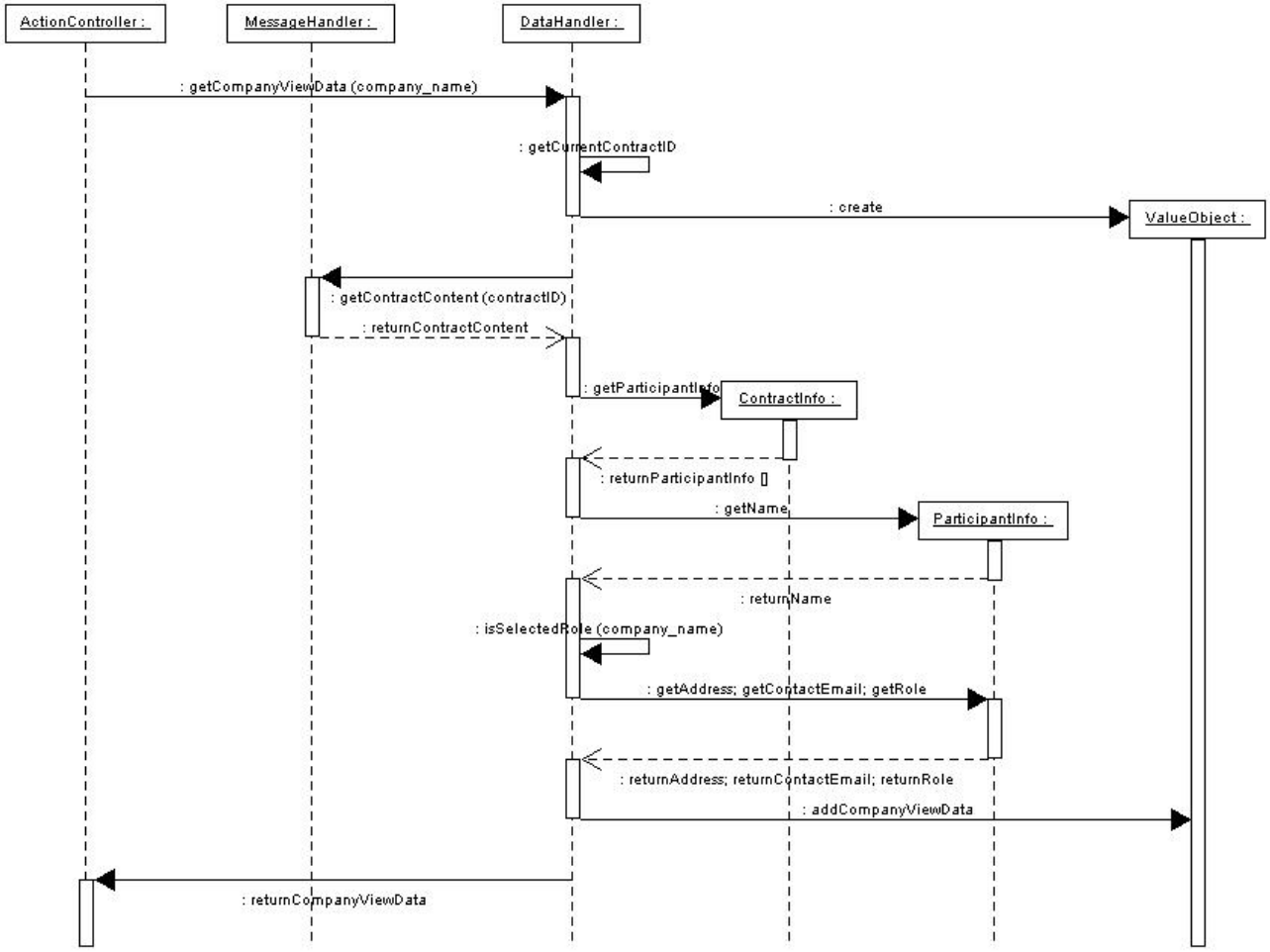
HabaDataHandler on keskeinen luokka sovelluksen osajärjestelmien välisten kutsusekvenssien kannalta. Alla on esitelty tyypilliset järjestelmäsekvenssit HabaDataHandlerin keskeisiin metodeihin liittyen. Sekvenssikuvissa esiintyvät kokonaisuudet voidaan hahmottaa tarkemmin tässä luvussa myöhemmin esiteltävästä HabaDataHandler-luokan luokkakaaviosta.

Kuvassa 23 on esitetty pelkistetysti aloitusnäkymään liittyvän tietosisällön koostamiseen liittyvä kustsusekvenssi ActionConrollerista lähtien.



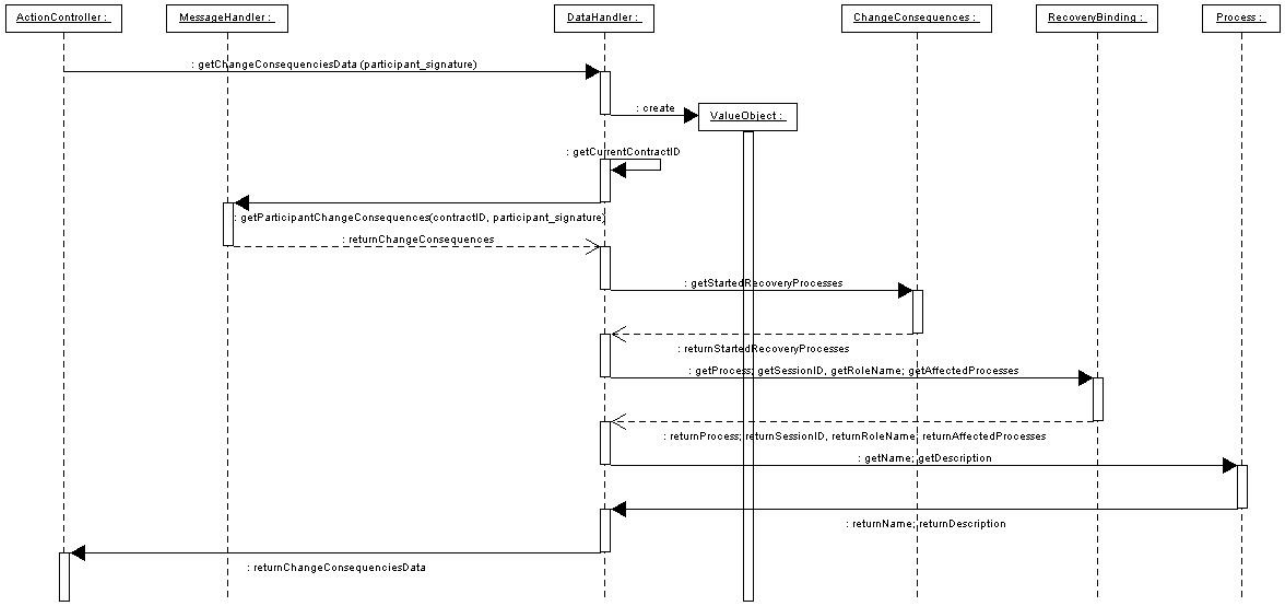
Kuva 23: Metodin getInitialViewBean() kutsua vastaava sekvenssi

Kuvassa 24 on esitetty yrityksen sopimuksen tietojen hakua vastaava kutsusekvenssi.



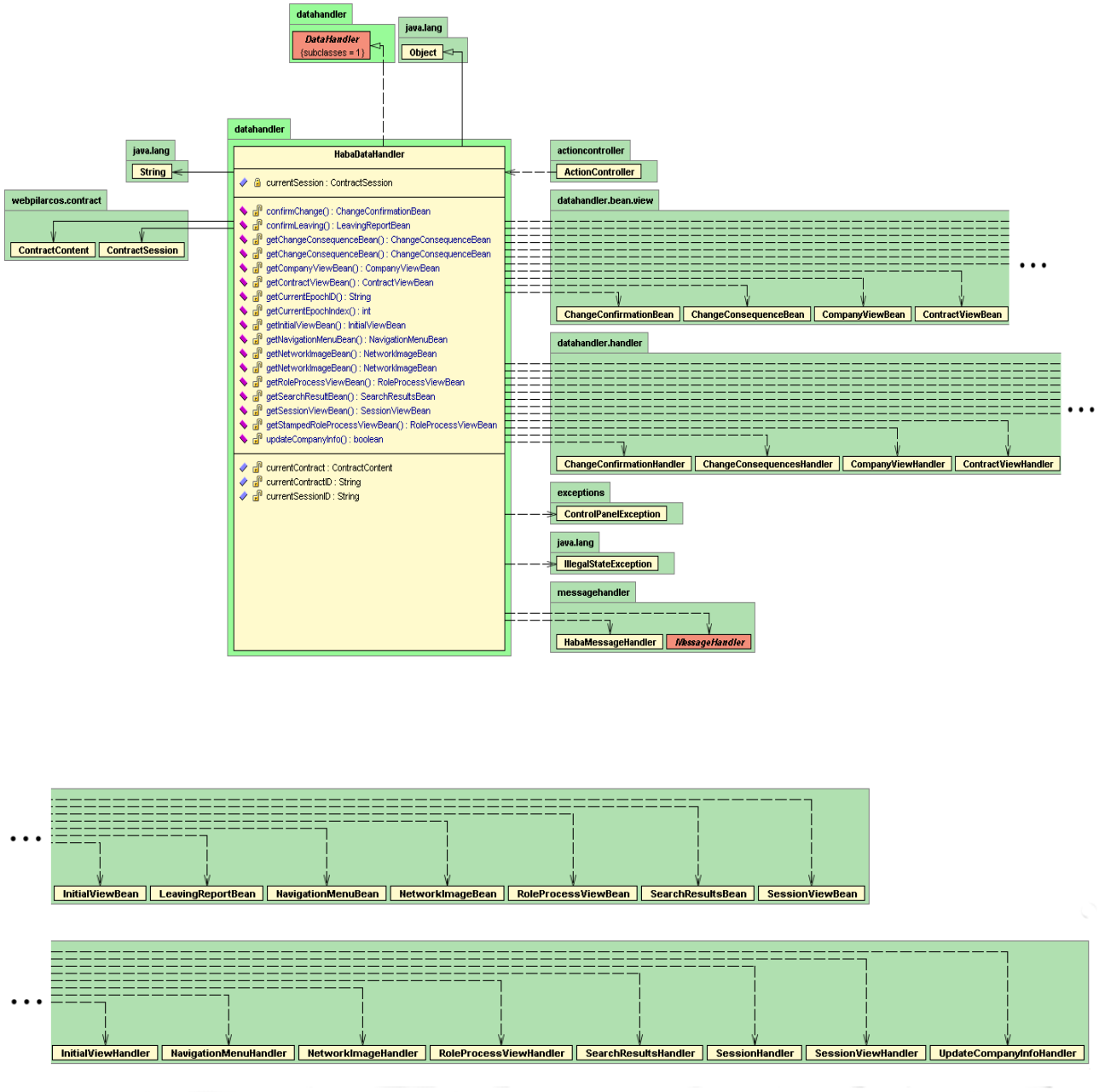
Kuva 24: Metodin getContractViewBean() kutsua vastaava sekvenssi.

Sopimuksen toimijan vaihtamisyrityksestä seuraava kutsuketju on esitetty kuvassa 25.



Kuva 25: Metodin getChangeConsequenceBean() kutsua vastaava sekvenssi

Kuvassa 26 on esitetty HabaDataHandler-luokan yhteydet luokkakaaviona. Luokan sisällössä keskeisenä näkyvät ActionControllerin käyttämät näkymäkohtaiset metodit.

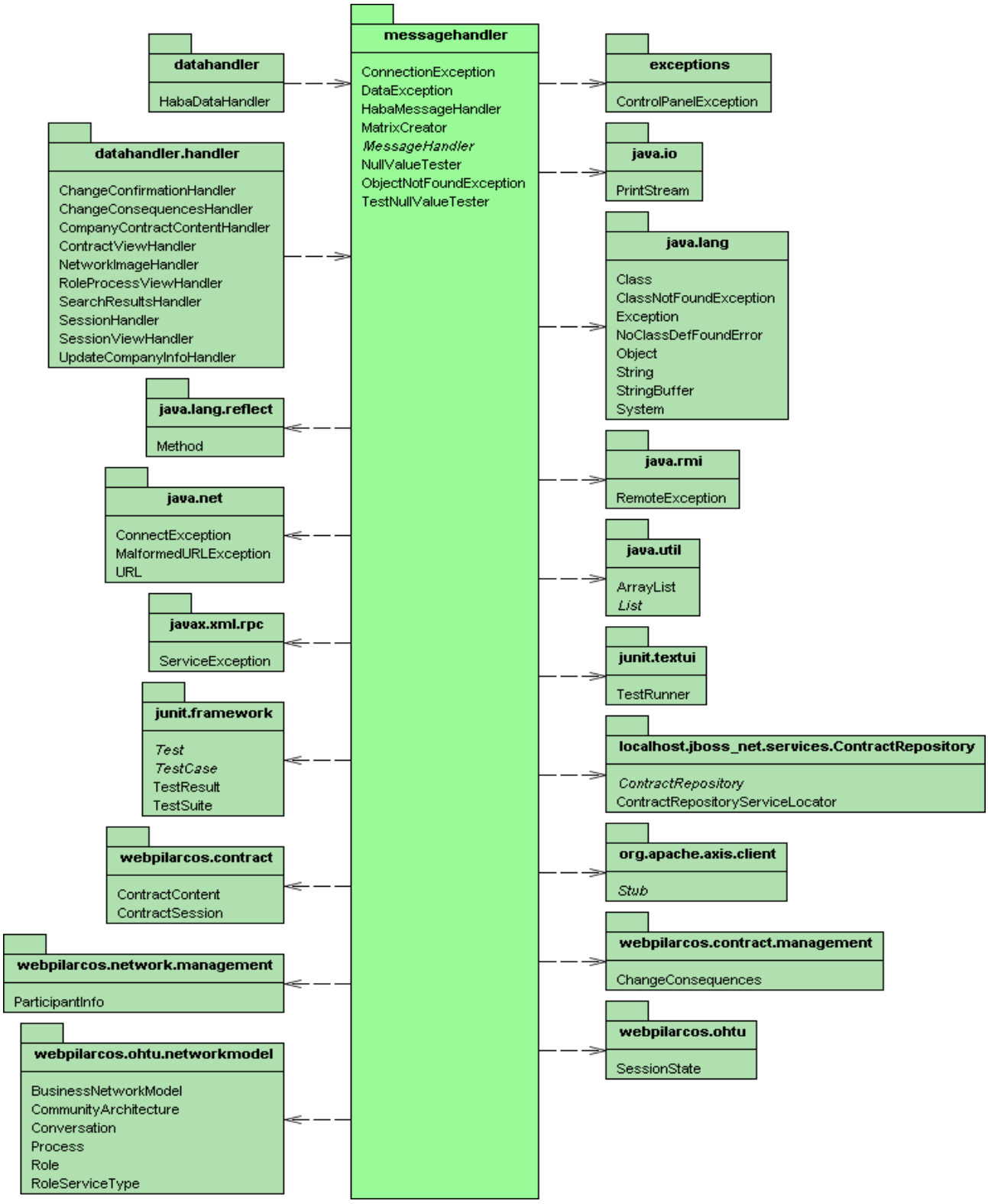


Kuva 26 : HabaDataHandler-luokan luokkakaavio

4.3 Messagehandler-pakkaus

Messagehandler-pakkaukseen kuuluvat MessageHandler-rajapinta sekä sen toteuttava HabaMessageHandler-luokka. Lisäksi pakkaus sisältää HabaControlPanelException –luokan aliluokat, joita HabaMessageHandler käyttää sopimusvarastoon liittyvien poikkeusten yhteydessä. Myös sopimusvaraston testaamiseen käytetty NullValueTester –luokka sijaitsee tässä pakkauksessa.

Kuvassa 27 on esitetty messagehandler-pakkauksen yhteydet luokkakaaviona.

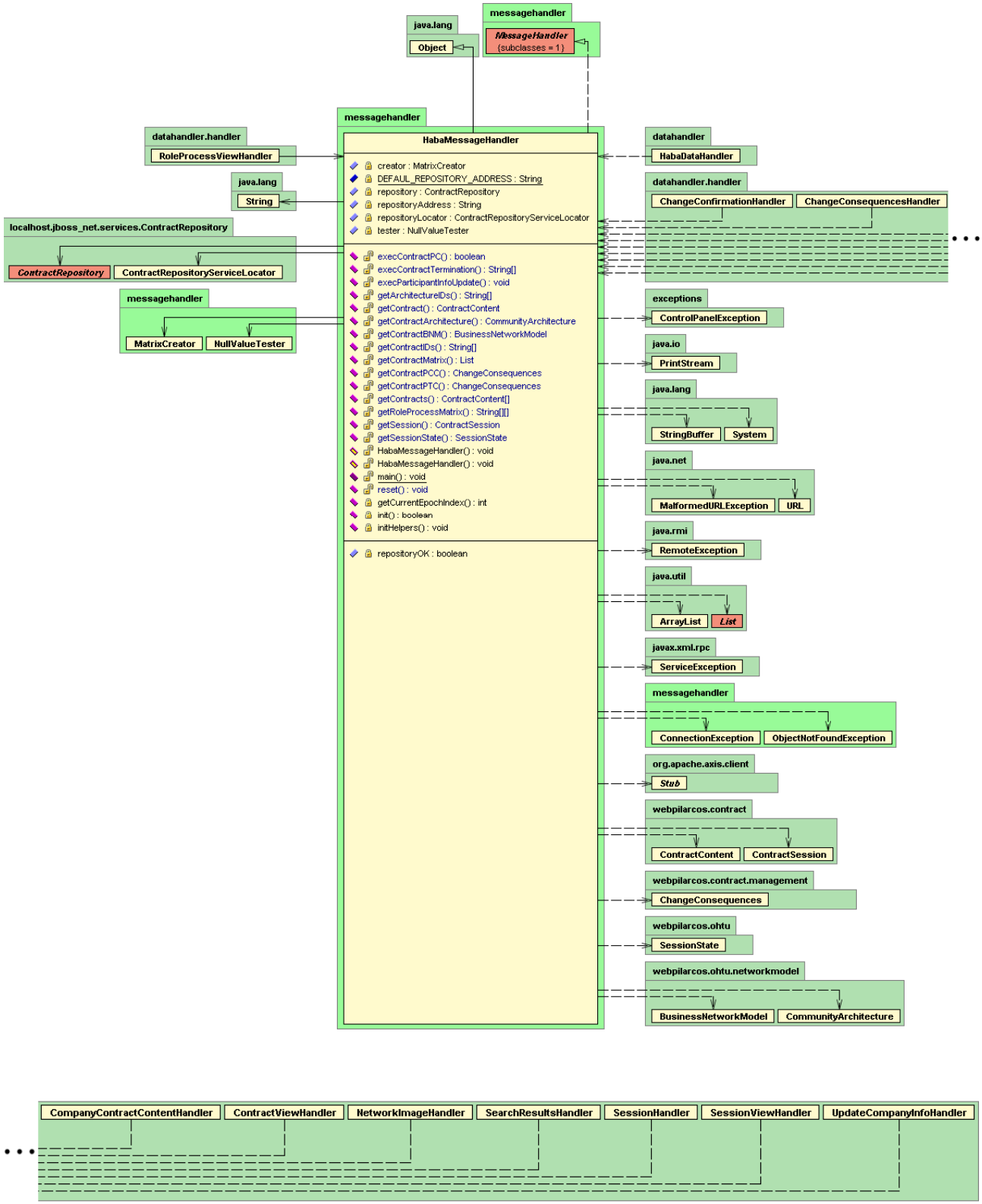


Kuva 27 : messagehandler-pakkauksen luokkakaavio

4.4 HabaMessageHandler

HabaDataHandler käyttää HabaMessageHandler -luokkaa tietosisällön noutamiseen sopimusvarastojärjestelmästä. Luokka koostaa myös visualisoinnissa käytettävän matriisin MatrixCreator luokan avulla. Luokka käyttää samassa pakkauksessa olevia HabaControlPanelException –luokan aliluokkia sopimusvarastoon liittyvien poikkeusten yhteydessä. Koska mahdollisista poikkeuksista ei voida toipua, ne välitetään throws –lauseella kutsuketjussa ActionControllerille, joka hoitaa virheiden käsittelyn.

Kuvassa 28 on esitetty HabaMessageHandler-luokan yhteydet luokkakaaviona. Keskeisenä sisältönä näkyy luokan julkinen rajapinta.



Kuva 28 : HabaMessageHandler-luokan luokkakaavio

4.5 Visualization-pakkaus

Ohjelman käyttöliittymä tarjoaa visualisoinnin yleisesti tarkasteltavan verkoston tilasta (epokkikohtaisesti) sekä rooliprosessitilasta. Näistä ensimmäisen kohdalla tila esitetään ns. suuntaamattomana verkkona. Jälkimmäisen tila puolestaan esitetään ns. suunnattuna verkkona (automaattina).

Koko visualisointi toteutetaan käyttäen hyväksi verkosta ladattavaa GNU-lisenssin alaista JUNG – kirjastoa [Jun04]. Käytetty kirjaston versionumero on 1.5.

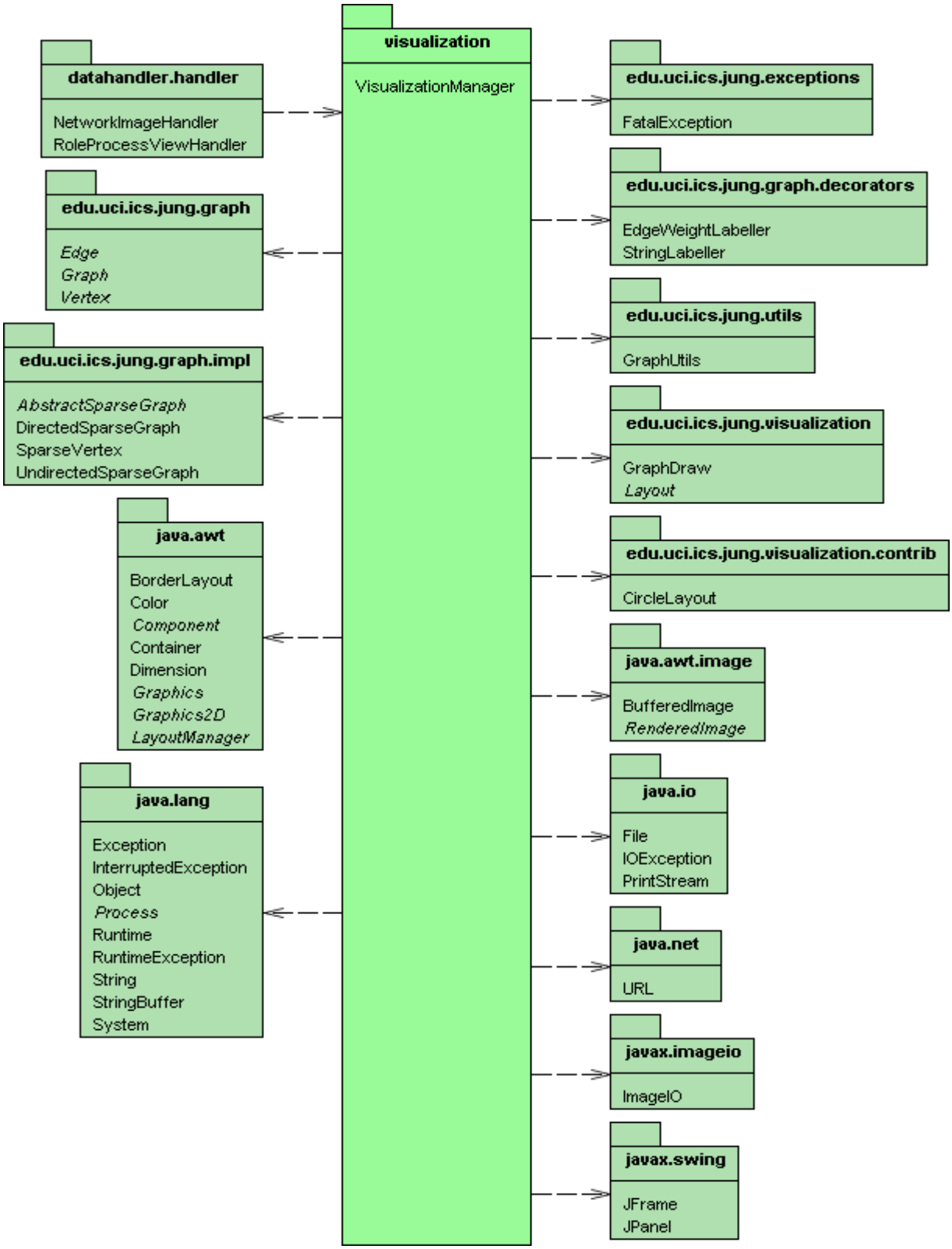
Visualisoinnin tuottamista varten julkisessa rajapinnassa on kaksi metodia: yksi molempia eri malleja varten (suunnattu, suuntaamaton verkko). Tätä rajapintaa käyttää DataHandler muodostaessaan näkymäkohtaista tietosisältöä JSP-sivuja varten.

Molemmat metodit saavat parametrinaan String – taulukon, jossa esitetään verkon rakenne matriisina. Jokainen matriisin alkiopari muodostaa solmuparin, joka on yhdistetty verkon esityksessä kaarella.

Jungin avulla tuotettu verkoston rakennetta esittävä kuva luetaan sen piirtoalustasta Javan omilla metodeilla, ja siitä muodostetaan PNG- muotoinen kuva. Tuotettu kuva tallennetaan levylle erikseen määriteltyyn hakemistoon. Kuvan nimeämiseen käytetään satunnaisfunktiota.

Funktio palauttaa generoidun kuvan URL- osoitteen, jonka avulla visualisointi sijoitetaan JSP-sivulla haluttuun kohtaan.

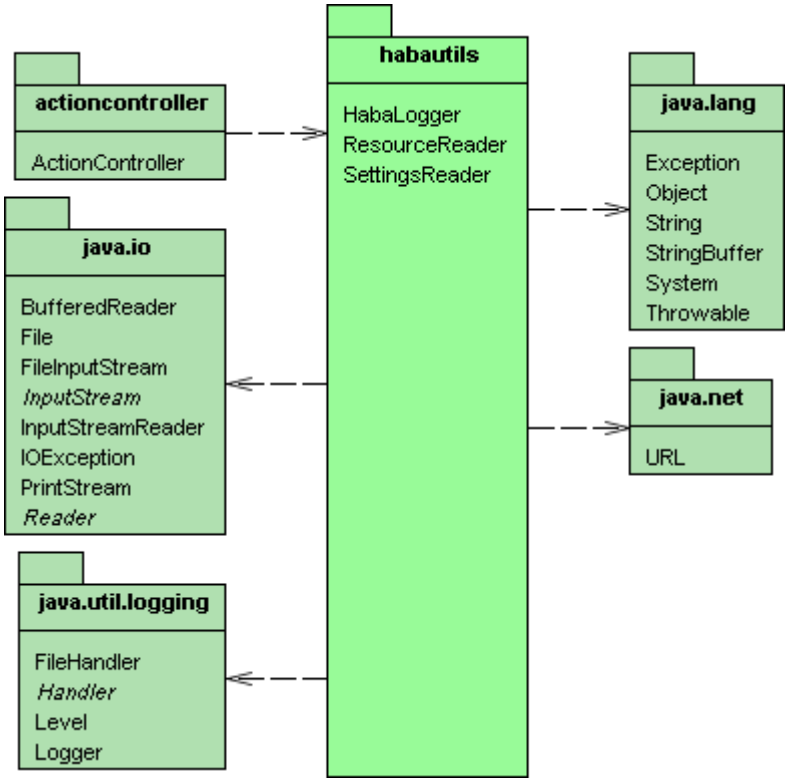
Kuvassa 29 on esitetty visualization-pakkauksen yhteydet luokkakaaviona.



Kuva 29 : visualization-pakkauksen luokkakaavio

4.6 Habautils-pakkaus

Habautils-pakkaus pitää sisällään järjestelmän lokin kirjoituksen sekä JSP- sivujen tulosteiden lokalisoinnin. Kuvassa 30 on esitetty habautils-pakkauksen yhteydet luokkakaaviona.



Kuva 30 : habautils-pakkauksen luokkakaavio

4.6.1 HabaLogger.java

HabaLogger käyttää Javan java.util.logging.Logger-luokan tarjoamia välineitä lokitiedon kirjoittamiseen.

Luokka tarjoaa JAVA_LOGGER- nimisen vakion lokin käsittelyyn. Lokin kirjoitus hoidetaan suoraan Logger-luokan log-metodilla, joka saa parametrinaan lokiin kirjoitettavan viestin sekä sen tyypin. Kirjoitettava viesti tallentuu näin asennustiedostossa määriteltyyn tekstitiedostoon. Loki kirjoitetaan XML - muodossa ja se noudattaa ohjelman mukana tulevan DTD:n mukaista rakennetta.

Lokiin kirjattavia virhetyyppejä on kolme: INFO, WARNING ja SEVERE. Näistä ensimmäinen ilmoittaa järjestelmän asianmukaisesta käynnistymisestä. WARNING virhetyyppiä käytetään varoittamaan järjestelmässä tapahtuneesta erikoistilanteesta, joka ei kuitenkaan vaaranna järjestelmän toimintaa. SEVERE virhetyyppiä käytetään ilmoittamaan poikkeuksesta, joka aiheuttaa järjestelmän toiminnan keskeytymisen ja erillisen virhesivun ilmaantumisen. Lokitiedoston kirjoittaminen on keskitetty ActionController-luokkaan.

4.6.2 SettingsReader.java

Tätä luokkaa ei käytetä sovelluksen lopullisessa jakeluversiossa, sillä vastaava toiminnallisuus on toteutettu lukemalla asetusarvot suoraan SystemSettings-luokasta. Luokka on kuitenkin mukana jakelupaketissa mahdollista myöhempää hyödyntämistä varten.

Luokka lukee järjestelmän asetukset tekstitiedostosta. Luettuja asetuksia voidaan käsitellä eri getSetting-metodeilla. Metodi palauttaa tiedostossa muuttujalle määritellyn arvon. Asennustiedostossa voidaan määritellä mm. sopimusvarastojärjestelmän osoite, lokitiedostojen nimi ja kuvien tallennushakemisto sekä kuvahakemiston URL-muotoinen osoite.

4.6.3 ResourceReader.java

Luokka on olemassa JSP-sivujen lokalisointia varten. Lokalisoinnilla tarkoitetaan tässä yhteydessä käyttöliittymän tekstinimistön keskitettyä hallintaa. Se hakee asennusohjeessa määritellystä tiedostosta avaimia ja niihin liitettyjä String-arvoja. Lokalisointitiedostossa jokaisella rivillä on esitetty yksi arvopari.

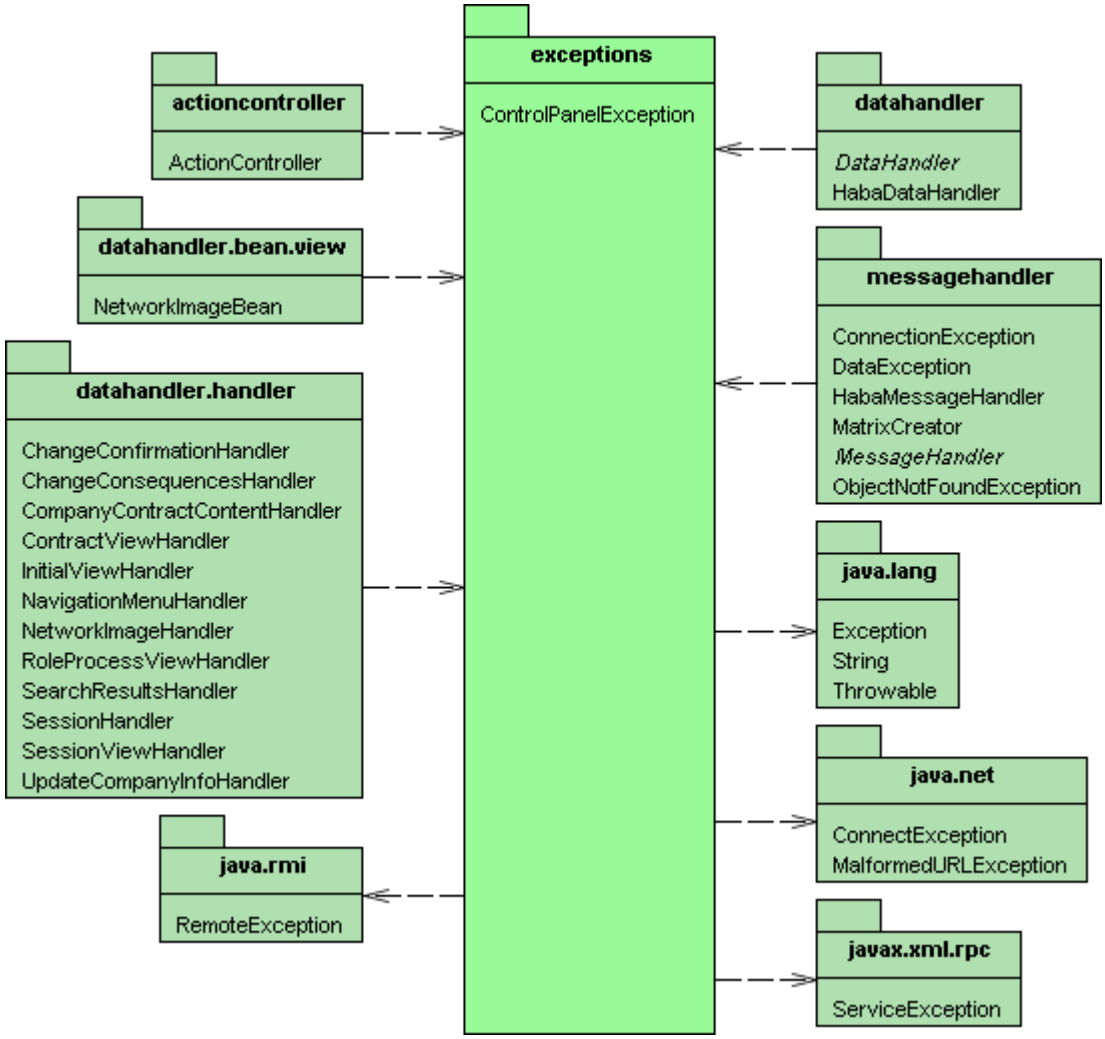
Tiedosto noudattelee muotoa KEY = VALUE. Esim pageTitle = Sessionview.

4.7 Exceptions-pakkaus

Exceptions-pakkaus pitää sisällään yliluokan järjestelmävirheiden käsittelyyn. Erilliset virheitä käsittelevät luokat löytyvät messagehandler-pakkauksesta.

Järjestelmävirheet siirretään kutsuketjun avulla actioncontrolleriin, joka hoitaa viime kädessä virhetilanteiden käsittelyn.

Kuvassa 31 on esitetty exceptions-pakkauksen yhteydet luokkakaaviona. Kaaviosta ilmenee missä luokissa on varauduttu järjestelmävirheiden esiintymiseen.



Kuva 31 : exceptions-pakkauksen luokkakaavio

5 Lähdeluettelo

[Jun04]

Jung-grafiikkakirjaston kotisivu, http://jung.sourceforge.net (3.11.2004)

[Mäk04]

Antti Mäen PowerPoint-esitys käyttöliittymästä,
http://www.cs.helsinki.fi/group/haba2004/docs/UI/kali8.ppt