CONCEPT-Sovellusprojekti

Sovellussuunnitelma

Pekka Kuuva
Tatu Repo
Pasi Saari
Anna Seppänen

Versio: 1.0
Julkinen
5. tammikuuta 2006

Jyväskylän yliopisto

Tietotekniikan laitos

Jyväskylä

Hyväksyjä	Päivämäärä	Allekirjoitus	Nimenselvennys
Projektipäällikkö	__.__.2006		
Tilaaja	__.__.2006		
Ohjaaja	__.__.2006		

Tietoa dokumentista

Tekijät:

- Pekka Kuuva (PK) pekuuva@cc.jyu.fi 044-2722979
- Tatu Repo (TR) tamikare@cc.jyu.fi 050-5851213
- Pasi Saari (PS) parisaar@cc.jyu.fi 044-3428411
- Anna Seppänen (AS) anhesepp@cc.jyu.fi 050-3275575

Dokumentin nimi: CONCEPT-Projekti, Sovellussuunnitelma
Sivumäärä: 25
Tiedosto: sovellussuunnitelma.tex

Tiivistelmä: Sovellussuunnitelmassa kuvataan CONCEPT-projektin tuottaman aikataulutussovelluksen suunniteltu toteutus ja tekniset valinnat.
Avainsanat: Aikataulu, konferenssi, sovellusprojekti.

Versiohistoria

Versio	Päivämäärä	Muutokset	Tekijät
0.1	9.11.2005	Ensimmäinen hahmotelma rungosta.	TR
0.2	21.11.2005	Laadittu sisältöä kaikkiin lukuihin.	TR
0.3	26.11.2005	Paranneltu kirjoitusasua,selkeytetty sisältöä kaikissa luvuissa.	TR
0.4	1.12.2005	Toimintalogiikkaa koskeva osio lisätty.	TR
0.5	2.12.2005	Käyttöliittymää koskeva osio lisätty, kaaviokuvat lisätty.	AS, TR
0.6	7.12.2005	Sovelluksen käynnistymistä ja autentikointia koskeva kappale sekä kaaviokuva lisätty.	TR
0.7	8.12.2005	Esimerkkikoodi, hakuosio, lisäluokat ja metodikuvaukset lisätty.	AS, TR
0.8	10.12.2005	Virheitä korjattu kirjoituksessa ja ladonnassa.	TR, PK
0.9	14.12.2005	Taulukko 7.1, kuvat 7.1 ja 8.1 ja liite A.	PK
0.91	21.12.2005	Parannuksia dokumentin tekstiasuun.	PK
0.92	3.1.2006	Tekstiasun viimeistely.	TR
1.0	4.1.2006	Dokumentin viimeistely, lisätty taulukko tietokannan tauluista.	TR

Tietoa projektista

CONCEPT-projekti toteuttaa liikuntabiologian ja tietotekniikan laitoksille, sekä LIKES-tutkimuskeskukselle työasemasovelluksen, jolla laaditaan aikataulu konferenssin esitelmille ja tapahtumille.

Tekijät:

- Pekka Kuuva (PK) pekuuva@cc.jyu.fi 044-2722979
- Tatu Repo (TR) tamikare@cc.jyu.fi 050-5851213
- Pasi Saari (PS) parisaar@cc.jyu.fi 044-3428411
- Anna Seppänen (AS) anhesepp@cc.jyu.fi 050-3275575

Tilaaja:

- Jouni Kallio jouni.kallio@sport.jyu.fi 014-2602054
- Janne Avela janne.avela@sport.jyu.fi 014-2603164
- Paavo Komi paavo.komi@sport.jyu.fi 014-2602073
- Veikko Vihko veikko.vihko@likes.fi 014-2601573
- Jyrki Komulainen jyrki.komulainen@likes.fi 014-2601574
- Kirsi Majava majkir@mit.jyu.fi 014-2602754
- Tuomo Rossi tro@mit.jyu.fi 014-2602755
- Lassi Paavolainen lopaavol@cc.jyu.fi 040-7183690
- Vesa Linnamo vesa.linnamo@sport.jyu.fi 040-5044800

Ohjaajat:

- Lari Kannisto kalahe@mit.jyu.fi 014-2603056
- Petteri Kela kapekela@cc.jyu.fi 040-7595922

Yhteystiedot:

- Sähköpostilistat: concept@korppi.jyu.fi
- Sähköpostiarkistot: https://korppi.jyu.fi/list-archive/concept/ind.html
- Työhuone: AgC 224.1 014-2604967

Sisältö

1 Johdanto

CONCEPT-projekti toteuttaa syksyn 2005 aikana aikataulunlaatimissovelluksen, joka tulee ensisijaisesti liitettäväksi ECSS07-kongressinhallintajärjestelmään. Projektin tavoitteena on kuitenkin yleiskäyttöisen konferenssien aikatauluttamiseen tarkoitetun työasemasovelluksen kehittäminen.

Tässä dokumentissa kuvataan CONCEPT-projektin tuottaman aikataulunlaatimissovelluksen tekniset yksityiskohdat, rakenne ja toiminta vaatimusmäärittelyssä [1] määriteltyjen vaatimusten toteuttamiseksi. Luvussa 2 kuvataan sovellussuunnitelmaan sisältyvät termit. Luvussa 3 esitetään yleiskatsaus sovelluksen komponentteihin ja niiden muodostamaan arkkitehtuuriin. Luvussa 4 esitellään sovelluksen käynnistämis- ja autentikointiprosessi. Luvuissa 5, 6, 7 ja 8 käsitellään tarkemmin toteutettavien komponenttien toimintaa. Projektin ohjelmakoodin kommentointi ja nimeäminen selvitetään luvussa 9. Dokumentin yhteenveto on luvussa 10.

2 Termit

Dokumentin aihealueen termejä ovat seuraavat:

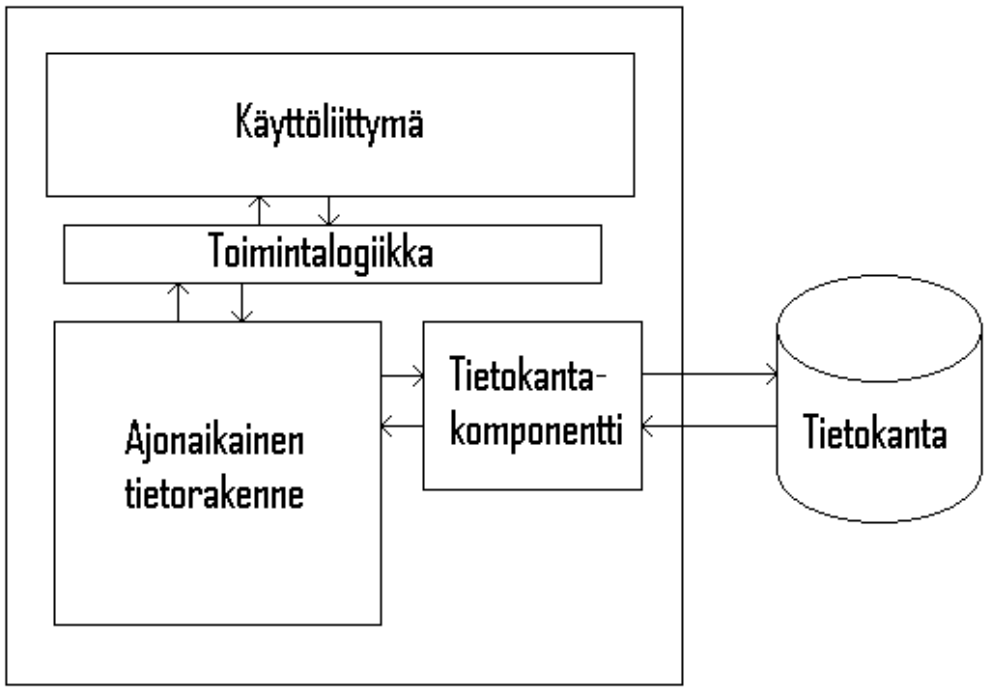
Konferenssi	Muutamasta päivästä viikkoon kestävä tapahtuma, jossa tutkijat tai heidän edustajansa tulevat esittelemään tutkimustuloksiaan ja kuuntelemaan muiden tuloksia.
Kongressi	Ks. Konferenssi.
Blokki	Blokki tarkoittaa tiettyä ajanjaksoa, esim. ma 10.12. klo 12:30 - 14:00.
ECSS	European College of Sport Sciences.
Sessio	Määrätyssä tilassa pidettävä sarja samantyyppisiä esitelmiä, jotka kaikki kuuluvat samaan aihealueeseen.
LIKES	Liikunnan ja kansanterveyden edistämissäätiö

Dokumentissa esiintyviä teknisiä termejä ovat seuraavat:

Java	Oliopohjainen laitteistoriippumaton ohjelmointikieli.
JBuilder	Borlandin kehittämä ohjelmistonkehitysympäristö Java-ohjelmointikielelle.
SQL	Structured Query Language on tietokantojen käsittelyyn tarkoitettu kieli.
PostgreSQL	Tietokannanhallintajärjestelmä
Parseri	Ohjelman osa, joka pyrkii yhdistämään ja jäsentämään, ”parsimaan”, eri lähteistä saatua tietoa toiseen muotoon.

3 Yleiskatsaus

Tässä luvussa kuvataan yleisesti sovelluksen rakennetta ja sen sisältämiä komponentteja. Luvussa käsitellään myös CONCEPT-projektin toteuttaman sovelluksen sijoittumista jo olemassa olevaan konferenssinhallintajärjestelmään. Sovelluksen arkkitehtuurin yleisnäkymä on esitetty kuvassa 3.1.



Kuva 3.1: Yleiskuvaus sovelluksen arkkitehtuurista.

3.1 Tietokanta

CONCEPT-projektin toteuttaman aikataulutussovelluksen tärkein tietolähde on konferenssinhallintajärjestelmässä sijaitseva PostgreSQL-tietokanta. Vaikka tietokanta on jo olemassa aikataulutussovelluksen kehityksen alkaessa, tulee tietokantaan tehdä muutamia muutoksia, jotta sen tarjoama tieto riittää aikataulutussovelluksen tar-

peisiin. Sovelluksen käyttämät tietokannan taulut on listattu taulukossa 3.1. Taulukkoon on myös merkitty suunnitellut lisäykset tietokannan rakenteeseen.

Taulu	Lisäys
article	
associate	
chairman	
chairmen	
colors	x
congress	
hall	x
halldays	x
hallrestriction	x
keyword	
list	
preference	x
present	
restriction	
schedule	
scheduletype	x
session	
sessiontype	
specialcase	
specialty	
topic	
users	

Taulukko 3.1: Sovelluksen käyttämät tietokannan taulut

3.2 Käyttöliittymä

Käyttöliittymän tarkoituksena on tarjota käyttäjälle monipuoliset ja intuitiiviset työkalut konferenssin aikatauluttamiseen. Koska kehitettävän sovelluksen tarkoituksena on mahdollistaa tehokkaampi aikataulutusprosessi, on käyttöliittymää kehitetty

hyvin asiakaslähtöisesti. Kehitettävä sovellus ei tarjoa automaattista aikataulutusta, vaan kätevän työkalun aikataulutuksen helpottamiseksi. Tämä korostuu etenkin käyttöliittymäsuunnittelussa. Käyttöliittymän tarkoituksena on toimia ”ikkunana” ajonaikaiseen tietorakenteeseen, eikä käyttöliittymä varastoi mitään tietoa.

3.3 Toimintalogiikka

Toimintalogiikka toimii lopullisena rajapintana käyttöliittymälle. Toimintalogiikan tehtävänä on paketoida ajonaikaisen tietorakenteen tarjoamat pienemmät toiminnallisuudet jonkun tietyn toiminnon toteuttavaksi kokonaisuudeksi. Näitä kokonaisuuksia toimintalogiikka tarjoaa metodeina ylöspäin käyttöliittymälle.

3.4 Ajonaikainen tietorakenne

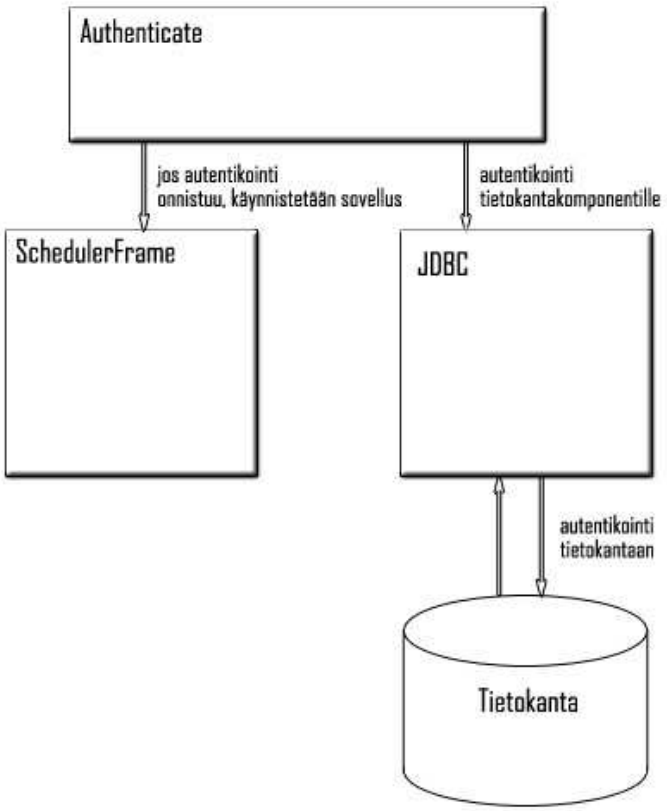
Ajonaikainen tietorakenne pyrkii simuloimaan tietokannan relaatiorakennetta olioiden välisellä viiterakenteella. Näin mahdollistetaan dynaaminen aikataulun muokkaus. Tietokannasta tuotu tieto sijoitetaan oliopohjaiseen tietorakenteeseen korvaten tietokannan taulut olioryhmillä ja relaatiot viiteyhteyksillä. Tämä tietorakenne tarjoaa käyttöliittymälle yhteyden tietokannan tietoihin ja muodostaa käyttöliittymän muokkaaman aikataulutusalustan.

3.5 Tietokantakomponentti

Tietokantakomponentti hoitaa sovelluksen vuorovaikutuksen käytettävään PostgreSQL-tietokantaan. Se tulkkaa käyttöliittymältä ja tietorakenteelta välittyneet kyselyt toimintalogiikan kautta SQL-muotoon, jossa ne välitetään eteenpäin tietokannalle. Tietokantakomponentin rooli korostuu etenkin hakuja tehdessä, sillä käyttöliittymän hakutoiminnot on tarkoitus toteuttaa tietokantaan tehtävillä hauilla. Tietokantakomponentti välittää tietoa myös tietokantaan päin, esimerkiksi tallennettaessa luotua aikataulua.

4 Sovelluksen autentikointi ja käynnistyminen

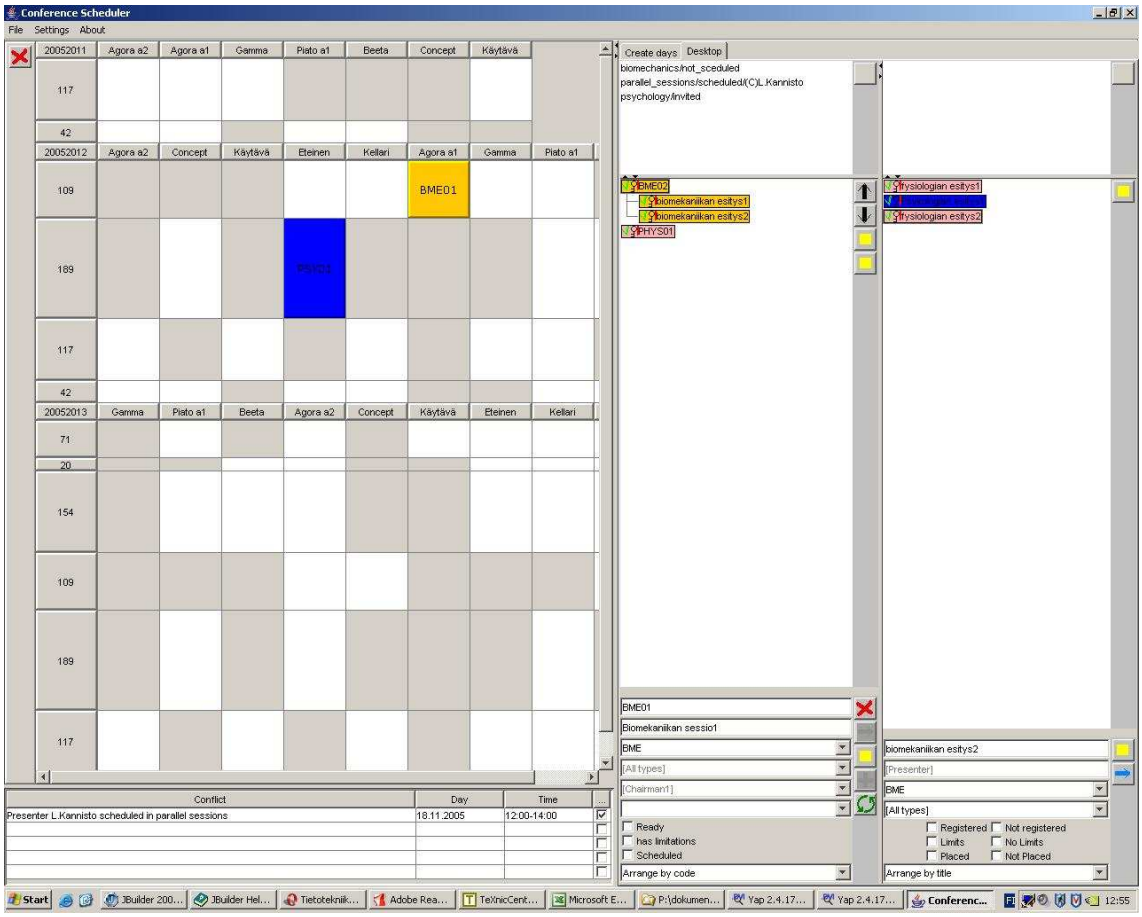
Sovelluksen autentikointia ja käynnistymistä varten on luotu kaksi luokkaa. Scheduler-luokka, joka käynnistää sovelluksen, sisältää sovelluksen varsinaisen pääohjelman. Scheduler-luokka huolehtii autentikointiprosessin ja itse sovelluksen käyttöliittymän käynnistämisestä. Autentikointi tapahtuu Authenticate-luokassa, joka huolehtii käyttäjän antaman käyttäjätunnuksen ja salasanan siirtämisestä tietokantakomponentille, joka puolestaan pyrkii ottamaan yhteyden tietokantaan näitä tunnuksia käyttäen. Mikäli yhteydenmuodostus onnistuu, luetaan tietokannan taulujen sisältö olioiden luomista varten ja käynnistetään sovellus kuvan 4.1 mukaisesti. Yhteys säilyy tietokantakomponentin hallussa kunnes se suljetaan. Mikäli kirjautumistiedoissa oli virheitä, palautuu tästä aiheutunut poikkeus Authenticate-luokan käsittelyyn ja autentikointiprosessi aloitetaan uudelleen.



Kuva 4.1: Autentikointiprosessi.

5 Käyttöliittymä

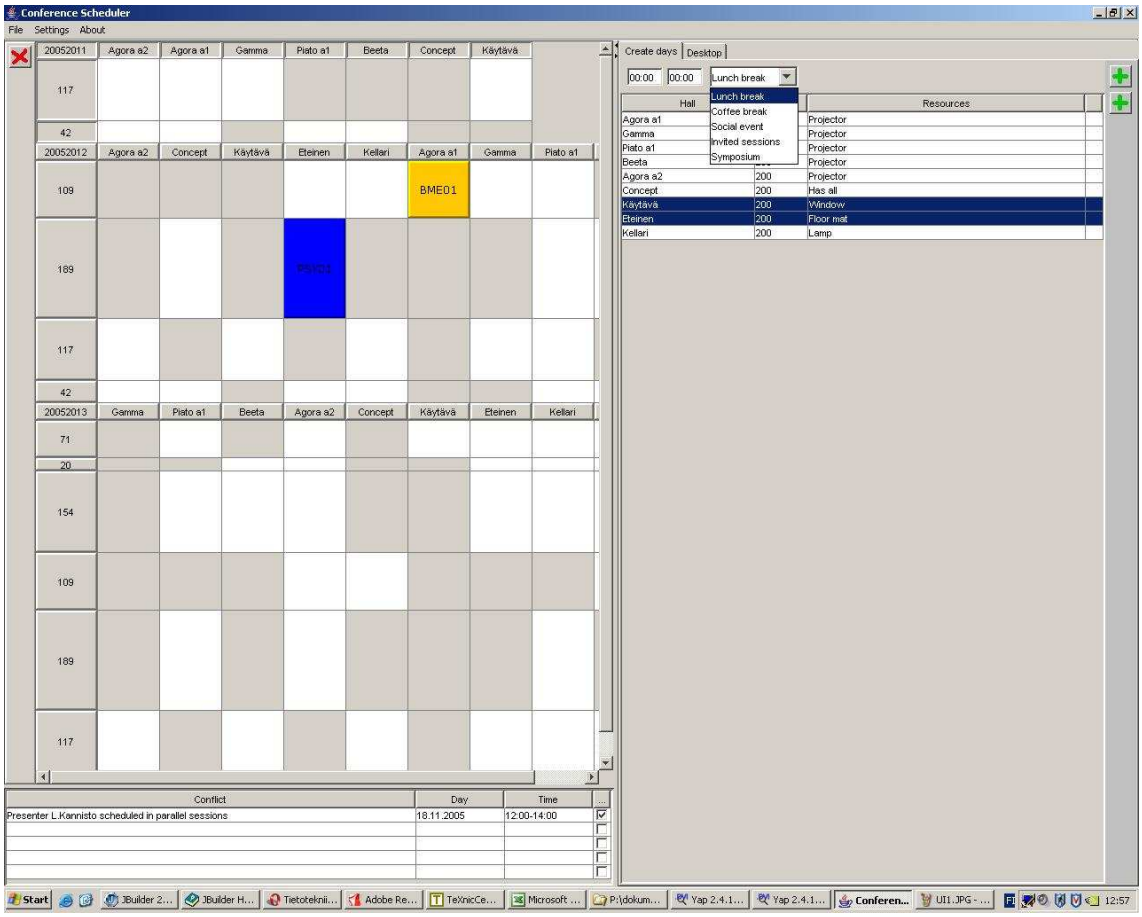
Sovelluksen käyttöliittymä koostuu pääikkunasta ja asetussivusta. Pääikkuna 5.1 on jaettu pystysuunnassa kahteen osioon. Vasemmalla puolella on aikataulu ja konfliktilista. Oikealla puolella on lehtiö, jonka toisella välilehdellä luodaan aikataulu (”Create Days”) 5.2 ja toisella on työpöytä (”Desktop”) 5.1 sessioiden luomiseen.



Kuva 5.1: Pääikkuna: aikataulu ja työpöytä.

5.1 Asetukset

Erillisellä asetussivulla voidaan määrittää tietokanta-asetukset, aihepiirien värit ja blokkityppit. Lisäksi asetuksissa määritetään konfliktitilanteiden asetukset. Konflikti voidaan sallia, estää tai siitä voidaan varoittaa.



Kuva 5.2: Pääikkuna: aikataulu ja aikataulun luominen.

5.2 Aikataulun luominen

Kun tietokantayhteys on saatu, aikataulussa on valmiina alkiot konferenssin päiville. Näistä käyttäjä luo aikataulun lisäämällä niihin saleja ja blokkeja. Päivistä valitaan ne, joille lisäys halutaan tehdä. Blokkeja voidaan lisätä yksitellen, mutta saleja voidaan lisätä useita kerralla. Kun käyttäjä lisää blokin, tai saleja, ne ilmestyvät valittujen päivien aikatauluun. Aikataulussa näkyvät valkeina ne ajat ja paikat, jolloin sali on käytettävissä ja siihen voi sijoittaa sessioita. Jos sali tai blokki halutaan poistaa päivältä, se valitaan aikataulusta ja poistetaan painikkeella. Poistamisen voi tehdä vain yksi sali tai blokki kerrallaan.

5.3 Sessioiden luominen työpöydällä

Työpöydällä käsitellään sessioita ja esityksiä. Työpöydällä tapahtuu sessioiden luominen, muokkaaminen ja poistaminen, sekä esitysten sijoittaminen sessioihin. Lisäksi työpöydällä voidaan tarkastella esitysten ja sessioiden tietoja. Työpöytä on jaettu pystysuunnassa keskeltä kahtia. Vasemmalla puolella käsitellään sessioita ja oikealla puolella esityksiä. Työpöytien objekteissa käytetään symboleita, jotka antavat niistä tietoa. Symboleista nähdään onko sessio merkitty valmiiksi, onko esityksillä esittäjiä jotka eivät ole rekisteröityneet, ja onko sessioilla tai esityksillä aikarajoituksia.

Kukin sessio tai esitys esiintyy työpöydällä vain kerran. Session valitseminen tai poistaminen aikataulusta, esityksen poistaminen sessiosta, session lisääminen sekä hakutoiminnot lisäävät objekteja työpöydille. Session raahaus aikatauluun tai esityksen raahaaminen sessioon poistavat objekteja työpöydiltä. Lisäksi työpöydältä voi tyhjentää valitsemansa objektit tai koko työpöydän. Kukin objekti esiintyy työpöydällä vain kerran ja viimeisimmän toimenpiteen käsittelemät objektit osoitetaan taustavärillä.

Vasemmalla puolella alaosassa on kentät session tietoja varten. Kenttien tietoja voidaan käyttää session lisäämiseen, muokkaamisen ja hakuehtojen antamiseen. Session lisääminen ja muokkaaminen tapahtuu täyttämällä session tiedot kenttiin, ja painamalla painiketta. Uusi sessio lisätään tällöin työpöydälle. Session tiedot näytetään samoissa kentissä, kun työpöydältä valitaan sessio. Kentät toimivat myös hakuehtojen antamiseen hakua tehtäessä. Kentät voidaan tyhjentää painikkeella. Oikealla alhaalla on puolestaan kentät esityksen tietojen näyttämiseen ja hakuehtojen antamiseen. Myös esityksen tiedot nähdään kentissä, kun esitys valitaan työpöydältä. Myös nämä kentät voi tyhjentää painikkeella. Esitysten haku tapahtuu hakuehtojen antamisen jälkeen painikkeella.

Työpöydän keskiosassa ovat varsinaiset työpöydät, johon käyttäjä voi tuoda tiedon jota haluaa käsitellä. Sessiotyöpöydällä session alla näkyvät kyseiseen sessioon sijoitetut esitykset. Esitykset saa näkyviin klikkaamalla sessiota. Sessiotyöpöydälle saadaan sessioita haun lisäksi valitsemalla aikataulusta sessio katseltavaksi tai editoitavaksi, tai poistamalla sessio aikataulusta.

5.4 Sessioiden aikatauluttaminen

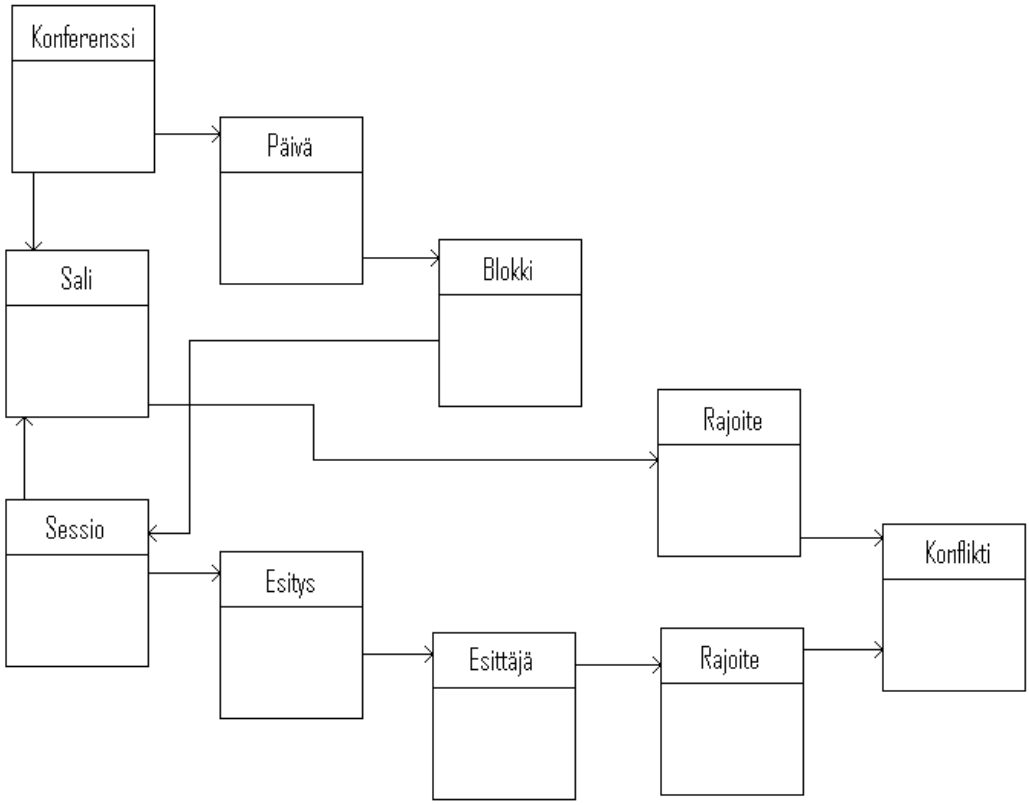
Sessiot aikataulutetaan raahaamalla ne sessiotyöpöydältä aikatauluun blokin ja salin muodostamaan vapaaseen soluun. Solu, johon sessio on sijoitettu määrää täten session ajan ja paikan. Session voi poistaa aikataulusta raahaamalla sen takaisin työpöydälle, tai poistamalla sen aikataulusta painikkeella. Session voi sijoittaa aikataulussa uuteen paikkaan raahaamalla sen toiseen vapaaseen soluun. Session voi raahata aikataulun solusta edelleen toiseen soluun. Kahden session paikkaa voi vaihtaa keskenään raahaamalla aikataulussa sessio toisen päälle.

5.5 Hakujen tekeminen

Sovelluksella voidaan hakea työpöydille esille esittäjiä ja sessioita. Hakuehdot annetaan työpöydän alaosassa oleviin kenttiin. Sessioita voidaan hakea lyhenteen, otsikon, tyypin, aiheen ja puheenjohtajien perusteella. Lisäksi sessioita haettaessa voidaan valita, haetaanko valmiiksi merkittyjä sessioita, aikataulutettuja sessioita vai sessioita, joissa jollain esityksellä on rajoituksia. Näistä lyhenteissä ja otsikoissa voidaan käyttää vapaata sanahakua. Esityksiä voidaan hakea otsikon, esittäjän, tyypin, aiheen, avainsanojen ja arvosanojen perusteella. Lisäksi esityksiä haettaessa hakuehdoiksi voidaan valita, onko esittäjä rekisteröitynyt, onko esityksellä tai esittäjällä aikarajoitteita, tai onko esitys sijoitettu sessioon. Esitysten hakuehdoista otsikossa ja esittäjässä voidaan käyttää vapaata sanahakua. Hakuja tehtäessä voidaan myös määritellä, minkä hakuehdon mukaisesti tulokset halutaan järjestää.

6 Ajonaikainen tietorakenne

Tietorakenne koostuu Session, Presentation, Presenter, ConferenceDay, Block, Conference, Hall ja ScheduleRestriction -luokista sekä niiden säiliöluokista. Luokissa määritellään rakenteen keskinäiset viitteet kuvan 2 mukaisesti. Säiliöluokkien nimeämisessä on käytetty yksittäisen elementin nimen monikkoa.



Kuva 6.1: Yksinkertaistettu kuvaus sovelluksen luokkarakenteesta viitteineen.

6.1 Conference

Tietorakenne voi sisältää yhden konferenssin kerrallaan. Mikäli toinen konferenssi halutaan ottaa käsittelyyn, on edellinen konferenssi suljettava ja tietorakenne alustettava toisen konferenssin tiedoilla. Sovelluksen käynnistyessä rakenne on tyhjä. Ensimmäisenä luodaan tietokantakomponentin avustuksella Conference-olio vastaamaan käyttäjän valitsemaa tietokannan Congress-taulua. Conference-luokka ra-

kentaa sille syntyessään annetun aikavälin mukaan itselleen oikean määrän konferenssipäiviä.

Conference-luokka määrittelee ilmentymilleen nimen, lyhenteen ja viitteen ConferenceDays-säiliöluokan ilmentymään, jossa sijaitsevat Congress-taulun Time_from ja Time_to kenttien perusteella luodut konferenssin päivät.

6.2 ConferenceDay

ConferenceDay on konferenssipäivää kuvaava luokka. Luokan ilmentymä sisältää Date-luokan ilmentymän kyseessä olevan päivän päivämäärästä, viitteen Halls-luokan ilmentymään sisältäen viitteet kyseisenä päivänä käytössä oleviin saleihin, viitteen Blocks-luokan ilmentymään sisältäen viitteet kyseiseen päivään liittyvistä blokeista ja viitteen kyseisen päivän säiliöluokan ilmentymään.

6.3 Block

Block on aikataulutusblokkia kuvaava luokka. Blokki on aikataulusuunnittelun apukeino, joka mahdollistaa koko aikataulutettavan päivän jonkin aikavälin aikataulutuksen sitomisen samaksi kaikkiin saleihin. Luokan ilmentymä sisältää alku- ja loppuajan millisekunteina, viitteen Sessions-luokan ilmentymään sisältäen blokkiin sijoitetut sessiot ja viitteen kyseisen blokin säiliöluokan ilmentymään.

6.4 Session

Session on konferenssin sessiota kuvaava luokka. Sessio on sovelluksen keskeisin aikataulutusosa muodostaen yhteyden esitysten ja aikataulun välille. Luokan ilmentymä sisältää tyypin, aiheen, nimen, lyhenteen, viitteen Presenters-luokan ilmentymään sisältäen kaikki kyseiseen sessioon liittyvät puheenjohtajat, viitteen Presentations-luokan ilmentymään sisältäen kaikki kyseiseen sessioon liittyvät esitykset ja viitteen kyseisen session säiliöluokan ilmentymään. Session-luokka tarjoaa metodit session tietojen muokkaamiseen sekä esitysten ja puheenjohtajien lisäämiseen sessioon linkitettyjen Presenters- ja Presentations-luokkien kautta. Lisäksi tarjotaan

metodia session rajoitekokonaisuuksien vertailusta johonkin annettuun rajoitteeseen.

6.5 Hall

Hall on konferenssin salia kuvaava luokka. Salitiedot tuodaan sovellukseen tietokannasta. Luokan ilmentymään sisältyy salin kapasiteetti, nimi, varustus, viite ScheduleRestrictions- säiliöluokan ilmentymään sisältäen kyseistä salia koskevat aikataulurajoitukset ja viite kyseisen salin säiliöluokan ilmentymään. Hall-luokka tarjoaa metodit salin linkittämiseksi sessioihin. Salin tietoja ei sovelluksessa muuteta.

6.6 Presentation

Presentation on konferenssin esitystä kuvaava luokka. Sovellus hakee esitysten tiedot tietokannasta, eikä niitä muokata sovelluksessa. Luokan ilmentymään sisältyy esityksen aihe, nimi, kesto minuutteina, annettu arvosana, viite Presenters-luokan ilmentymään sisältäen kaikki esitykseen liittyvät esittäjät ja viite kyseisen esityksen säiliöluokan ilmentymään. Presentation-luokka tarjoaa metodit esitysten muodostamiseen tietokannan tietojen perusteella ja esitysten linkittämiseen sopiviin sessioihin. Esitysten tietoja ei sovelluksessa muuteta.

6.7 Presenter

Presenter on konferenssin esittäjää kuvaava luokka. Konferenssin esittäjät tuodaan tietokannasta ja ne on jo sovelluksen käytön alkaessa sidottu vastaaviin esityksiin. Sovelluksen yhteydessä sessioiden puheenjohtajaviitteet osoittavat Presenter-luokan ilmentymiin. Luokan ilmentymä sisältää esittäjän nimen, tiedon onko esittäjä puheenjohtaja ja viitteen kyseisen esittäjän säiliöluokan ilmentymään. Presenter-luokka tarjoaa metodit esittäjän luomiseen ja puheenjohtajuuden linkittämiseen. Esittäjät on sovelluksen käytön alkaessa jo linkitetty esityksiin, eikä esittäjien tietoja muuteta sovelluksessa.

6.8 ScheduleRestriction

ScheduleRestriction-luokka kuvaa kaikkia rakenteen olioihin kohdistuvia ja aikataulutukseen vaikuttavia rajoitteita. Luokan ilmentymä sisältää aikataulurajoitteen ajallisen vaikutusalueen sekä sen kuvauksen. Näitä rajoiteolioita vertaamalla tietorakenne havaitsee mahdolliset aikataulutuskonfliktit. Aikataulurajoitteita sisältävät Session-, Presenter-, Presentation- ja Hall-luokat. Rajoiteolio muodostetaan kahdesta aikarajasta, jotka järjestetään oikeaan järjestykseen. Näiden välinen aika on rajoitteen kattama aika. Mikäli joku toinen verrattava rajoite leikkaa tämän aikavälin kanssa, syntyy vertailutilanteessa konflikti. Kahden rajoitteen välisessä vertailussa löytynyt konflikti synnyttää Conflict-luokan esiintymän. ScheduleRestriction-luokka tarjoaa metodit rajoitteiden vertailemiseen ja luomiseen. Rajoitteita ei luomisen jälkeen muokata. ScheduleRestriction-luokan ilmentymät varastoidaan kantaan.

6.9 Conflict

Conflict-luokka kuvaa tietorakenteen elementtien aikataulutuksessa syntyneitä konflikteja. Sen ilmentymät syntyvät rajoiteolioihin kohdistuvan vertailun ilmoittaessa konfliktin olemassaolosta. Konfliktit sisältävät sen synnyttäneen rajoitteen tai rajoitteiden viitteet sekä viestin, joka antaa tietoa konfliktin luonteesta. Conflict-luokka tarjoaa metodit konfliktin luomiseen sekä kuvaavan konfliktiviestin muodostamiseen konfliktin muodostavien rajoitteiden ominaisuuksien pohjalta. Conflict-luokan esiintymiä ei varastoida tietokantaan, vaan ne luodaan dynaamisesti sovelluksen aikataulutuselementtien rajoitteista.

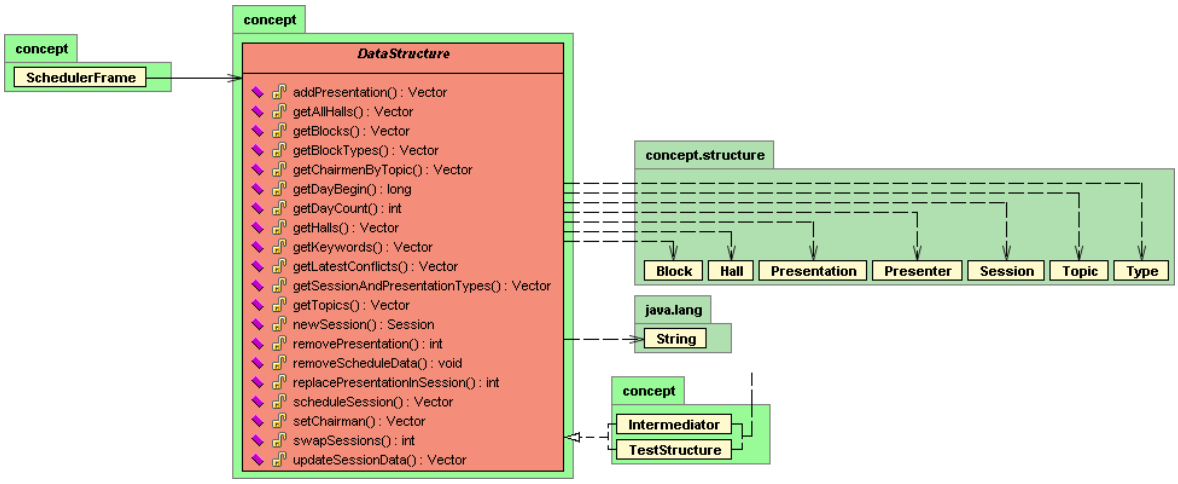
6.10 Tietokannan ja sovelluksen kommunikaatiota helpottavat luokat

Tietokannan tauluissa ja niitä yhdistävissä relaatioissa on varastoituna tietoa, jota sovelluksen tulee käyttää toiminnassaan ja huomioida tietokantaa päivitettäessä. Tämän tiedon ylläpitämiseksi on ajonaikaiseen tietorakenteeseen tehty kaksi erillistä luokkaa. Näitä luokkia ovat Type ja Topic. Type-luokan esiintymät määräävät esitysten ja sessioiden tyypin. Session ja siihen kuuluvan esityksen tyypin erotes-

sa syntyy konflikti. Topic-luokan esiintymät kuvaavat eri aihealueita. Sovelluksessa session aihealue vaikuttaa siihen osallistuviin esityksiin ja puheenjohtajiin. Mikäli session ja sen esitysten tai puheenjohtajien aihealueet eroavat, syntyy konflikti. Aihealue vaikuttaa myös suoritettaviin hakuihin, esimerkiksi puheenjohtajia voi hakea aihealueen mukaan. Topic- ja Type-luokat on suunniteltu samankaltaisiksi tietokannan vastaavien taulujen kanssa.

7 Toimintalogiikka

Toimintalogiikka muodostaa yhteyden ajonaikaisesta tietorakenteesta käyttöliittymään. Toimintalogiikan tehtävänä on rakentaa jokin tietty toiminto, esimerkiksi session lisääminen aikatauluun, yhdeksi kokonaisuudeksi, joka tässä tapauksessa ilmenee metodina. Toimintalogiikka käsittelee suoraan ajonaikaisen tietorakenteen olioiden tarjoamia metodeja. Näiden metodien kautta rakennetaan kokonaisuuksia, joita käyttöliittymä voi kutsua havaitessaan kokonaisuutta vastaavan toiminnon. Toimintalogiikka toteutetaan luokalla Intermediator, joka toteuttaa sovelluksen tarpeisiin erikseen määritellyn rajapinnan DataStructure. Tätä rajapintaa käyttämällä voidaan sovelluksen toimintoja hyödyntävää käyttöliittymäkomponenttia vaihtaa suhteellisen yksinkertaisesti. Toimintalogiikkaan rakenteellisesti liittyvät luokat on esitetty kuvassa 7.1.



Kuva 7.1: Toimintalogiikan muodostava rajapinta ja siihen liittyvät luokat

Ajonaikaisen tietorakenteen tarjoamat metodit on suunniteltu toteuttamaan sisäisiä muutoksia tietorakenteeseen paikallisesti. Jotta suoritetun toiminnon aiheuttamat muutokset koko tietorakennetta ajatellen saadaan tallennettua, täytyy paikalliset toiminnot paketoida koko rakennetta käsittelevään kokonaisuuteen. Sovellus on suunniteltu erillisinä komponentteina, joten toimintalogiikan kaltaisen rajapinnan merkitys etenkin integraatiovaiheessa korostuu.

Toimintalogiikan pohjana toimiva rajapinta DataStruct on koottu metodeista (Taulukko 7.1), jotka vastaavat käyttöliittymän toteuttamia tehtäväkokonaisuuksia. Rajapinnan toteuttava Intermediator-luokka kerää tietorakenteesta tarvittavat metodit jonkin tehtäväkokonaisuuden toteuttamiseen.

Toimintalogiikan toiminnasta esimerkkinä on sekvenssikaavio (Liite A), joka kuvaa esityksen lisäämistä aikataulutettuun sessioon. Kuvassa esitetään metoditason viestinkuljetusta eri komponenttien välillä yli määritellyn DataStructure-rajapinnan.

Metodi	Kuvaus
public Vector getTopics();	Palauttaa aiheen.
public Vector getKeywords(Topic topic);	Palauttaa aiheeseen liittyvät avainsanat.
public Vector getBlockTypes();	Palauttaa Blokkityypit.
public Vector getSessionAndPresentationTypes();	Palauttaa Sessio- ja esitystyypit.
public Vector getLatestConflicts();	Palauttaa konfliktit.
public int getDayCount();	Palauttaa päivien lukumäärän.
public long getDayBegin(int index);	Palauttaa päivän ensimmäisen millisekunnin.
public Vector getAllHalls();	Palauttaa tietokannan salit.
public Vector getHalls(int day);	Palauttaa päivän salit.
public Vector getBlocks(int day);	Palauttaa päivän blokit.
public Vector getChairmenByTopic(Topic topic);	Palauttaa aihepiirin käytettävissä olevat puheenjohtajat.
public Vector setChairman(Session s, Presenter p, int orderNumber);	Lisää sessioon s puheenjohtajan p järjestysnumeroineen.
public int removePresentation(Session s, Presentation p);	Poistaa esityksen p sessiosta s.
public Session newSession(String abbrv, String Title, Topic topic, Type type);	Luo uuden session.
public Vector updateSessionData(Session s, String abbrv, String Title, Topic topic, Type type, Presenter cman1, Presenter cman2, boolean ready);	Päivittää session tiedot.

Metodi	Kuvaus
public Vector scheduleSession(Session s, Hall h, Block b);	Asettaa sessiolle s uudet aikataulutiedot.
public int swapSessions(Session s1, Session s2);	Vaihtaa sessioiden s1 ja s2 aikataulutiedot.
public void removeScheduleData(Session s);	Poistaa session s aikataulutustiedot.
public int replacePresentationInSession(Presentation p, int newPlacement);	Siirtää esityksen p sessiossaan.
public Vector addPresentation(Session s, Presentation p);	Lisää esityksen p sessioon s.

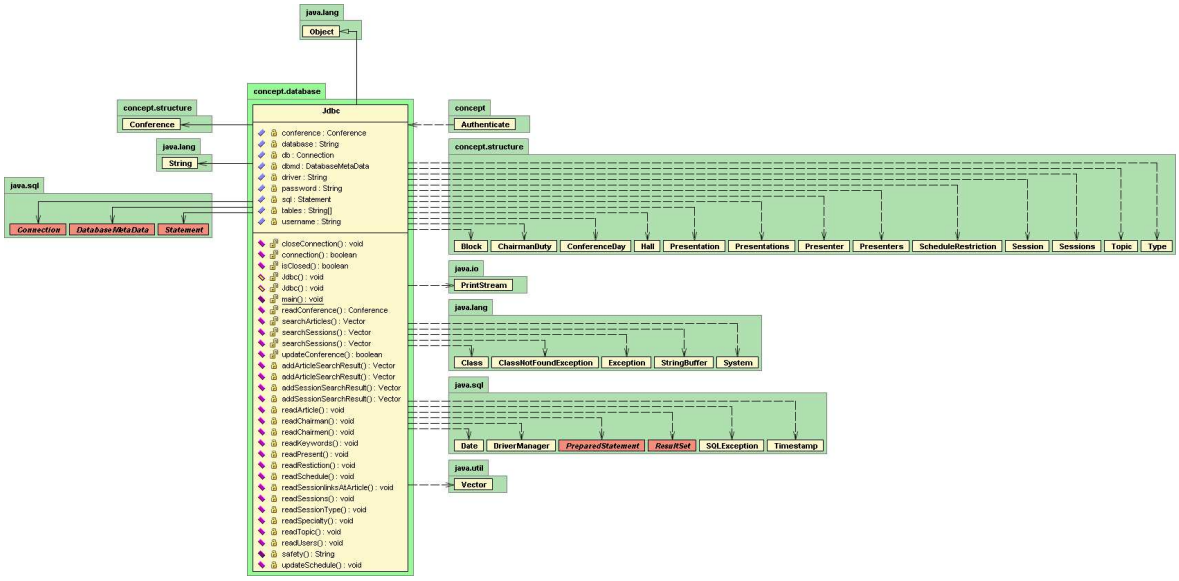
Taulukko 7.1: DataStruct-rajapinnan toiminta.

8 Tietokantakomponentti

Tietokantakomponentin tehtävänä on mahdollistaa muun sovelluksen kommunikointi PostgreSQL-tietokannan kanssa. Tietokantakomponentti rakennetaan Javan JDBC-tietokanta-alustan päälle. Tietokantakomponentti käyttää PostgreSQL-ajuria tietokantayhteyden luomiseen. Sovelluksen käynnistyessä käyttäjä antaa tietokannan osoitteen ja se välitetään tietokantakomponentille. Käyttäjä syöttää myös käyttäjätunnuksensa sekä salasanansa, joilla tietokantakomponentti pyrkii tietokantaan. Tietokantakomponentti toimii siis autentikoivana osana järjestelmässä. Tietokantakomponentin yhteydet muihin luokkiin ilmenevät kuvasta 8.1.

Komponentti koostuu pääasiallisesti tietokantaa oliorakenteeksi kirjoittavasta parserista sekä vastaavasti oliorakenteen tietoja tietokantaan kirjoittavasta parserista. Kun sovellus käynnistetään, tietokantakomponentti ottaa yhteyden käyttäjän osoittamaan tietokantaan käyttäjän antamilla tunnuksilla. Tietokantakomponentti hakee tietokannan tiedot SQL-kyselylausekkeilla ja muodostaa tietojen pohjalta olioita ajonaikaiseen käyttöön. Tämän helpottamiseksi on ohjelmiston kehityksessä pyritty yhdenmukaisuuteen tietokannan ja olioiden attribuuttien välillä. Kun tietokannan rivejä vastaavat oliot on luotu, muodostetaan viiterakenne vastaamaan tietokannan relaatiotaulujen kuvaamaa rakennetta. Tämän jälkeen on tietokantakomponentti luonut ajonaikaisen oliorakenteen ja käyttäjä pääsee käsiksi tietoihin käyttöliittymän välityksellä.

Kun suoritetaan haku sovelluksessa, se välitetään tietokannan hakuresursseille tietokantakomponentin kautta. Ensin sovelluksen ylläpitämä oliorakenne kirjoitetaan kantaan, jotta käyttäjän näkemä tilanne vastaisi hakutuloksia. Olioiden tietojen pohjalta muodostetaan SQL-lauseita luomaan oliorakennetta vastaava kanta. Tämän jälkeen tietokantakomponentti muodostaa hakua vastaavan SQL-lauseen ja tietokanta palauttaa hakua vastaavan tiedon. Tietokantakomponentti tunnistaa kannan antamasta hakutuloksesta vastaavat oliot ja palauttaa tietorakenteen kautta viitteet käyttöliittymälle.



Kuva 8.1: JDBC-komponentin yhteydet.

9 Ohjelmakoodi

Sovellus toteutetaan Java-ohjelmointikielellä. Näin siitä saadaan alustariippumaton ja sovellus toimii missä tahansa Java-virtuaalikoneella (v1.4) varustetussa ympäristössä. Ohjelmisto toteutetaan käyttäen Borlandin kehittämää JBuilder 2005 Enterprise ohjelmistonkehitysympäristöä. Ohjelmakoodin nimeäminen ja kommentointi tehdään englanniksi. Sovelluksen rajapinnoissa ja teknisessä toteutuksessä tavoitellaan hyvää jatkokehitettävyyttä. Ohjelmakoodin nimeämisestä ja kommentoinnista on esimerkki liittessä B.

9.1 Nimeäminen

Java-luokkien nimet kirjoitetaan isolla alkukirjaimella esimerkiksi Session. Muuttujien ja metodien nimet kirjoitetaan pienellä alkukirjaimella ja ensimmäistä seuraavat sanat aloitetaan isolla kirjaimella, esimerkiksi getSessionCount().

9.2 Kommentointi

Ohjelmakoodin kommentoinnissa käytetään Javadoc- dokumentointia. Erillistä selvitystä vaativat ohjelmakoodilohkot kommentoidaan vielä erikseen toiminnan selkiyttämiseksi. Kommentointi tullaan toteuttamaan huolellisesti ja siten tarjoamaan hyvät lähtökohdat jatkokehitykselle.

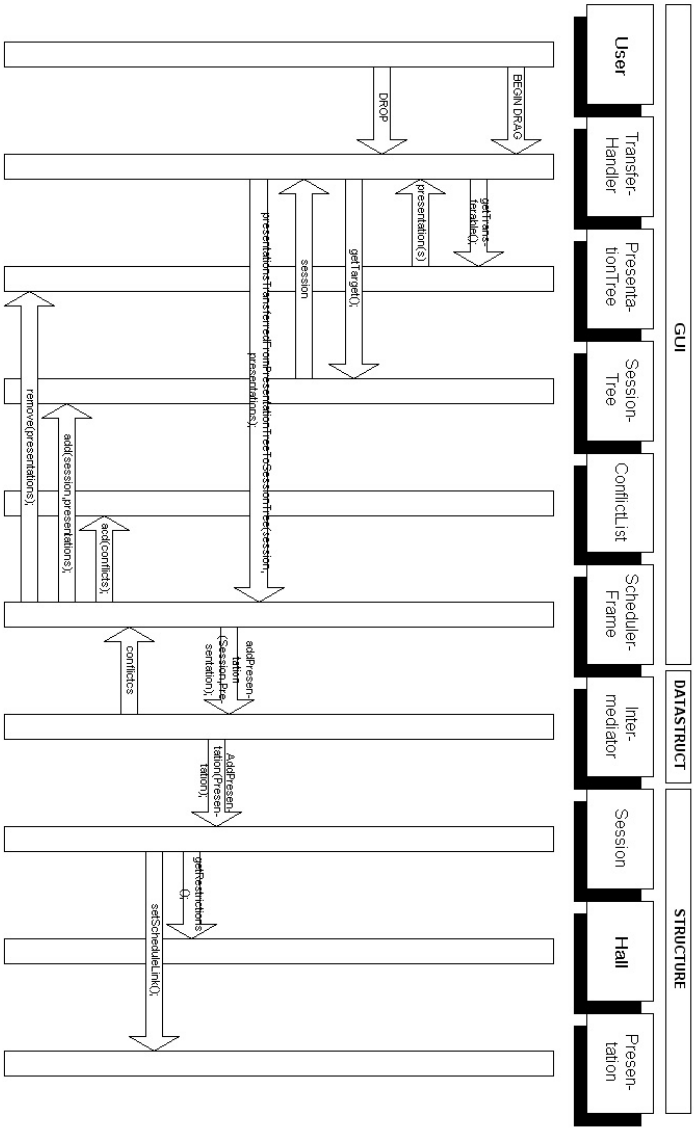
10 Yhteenveto

Sovelluksen suunnittelu on toteutettu useassa iteratiivisessa vaiheessa. Sovellussuunnitelman rakentamista onkin oleellisesti vaikeuttanut sovelluksen suunnittelun dynaaminen luonne. Tämä dokumentti pyrkii esittämään yhden suunnitteluvaiheen tuloksena saadun sovellussuunnitelman. Sovelluksen rakenteen hahmotuttua se on pääpiirteittäin pysynyt ennallaan. Sovellus koostuu käytännössä neljästä komponentista, joita on pyritty erottamaan toisistaan tarpeen vaatiessa. Käyttöliittymäkomponentin erottuminen toiminnallisesta rakenteesta oli eräs suunnittelun lähtökohdista. Tämä on sovelluksen selkein rajapinta ja mikä tahansa käyttöliittymäkomponentti, joka toteuttaa kyseisen rajapinnan, voi toimia sovelluksen käyttöliittymänä. Lisäksi tietokantaoperaatioiden kokoamista saman komponentin alle on pidetty tärkeänä. Suunnittelun tavoitteena oli myös pitää sovelluksen arkkitehtuuri mahdollisimman selkeänä ja yksinkertaisena. Näin toteutettavat komponentit säilyttävät omat tehtäväkokonaisuutensa, eikä sovelluksessa ilmene päällekkäisyyksiä. Sovelluksen rakenteelliset elementit, etenkin tietorakenteen muodostavat luokat, on luotu vastaamaan mahdollisimman hyvin tietokannan tauluja. Tämä vastaavuus helpottaa oleellisesti tietokannan tietojen ja ajonaikaisen tietorakenteen välisiä muunnoksia.

11 Lähteet

[1] PK TR PS AS, ”CONCEPT-projektin vaatimusmäärittely”, 2005

A Sekvenssikaavio



Kuva A.1: Esityksen lisääminen aikataulutettuun sessioon.

B Esimerkki ohjelmakoodin kommentoinnista ja nimeämisestä

```
/**
 *
 * <p>Title: Esimerkkiluokka</p>
 *
 * <p>Description: Luokka, jolla kuvataan projektin
 * kommentointi- ja nimeämiskäytäntöjä.</p>
 *
 * <p>Copyright: Copyright (c) 2005</p>
 *
 * @author Tatu Repo
 * @version 1.0
 */
public class EsimerkkiLuokka {
    public EsimerkkiLuokka() {
    }

    /**
     * Esimerkkimetodi, jolla kuvataan nimeämiskäytäntöjä
     * @return int palautusarvo, jonka avulla kuvataan
     * kommentointia.
     */
    public int esimerkkiMetodi() {  //Metodin nimeäminen.
    String esimerkkiMuuttuja = "Muuttujan nimeäminen.";
        return 1;
        //Näin merkitään lisäkommentointi.
    }

}
```