CONCEPT-Sovellusprojekti

Sovellusraportti

Pekka Kuuva
Tatu Repo
Pasi Saari
Anna Seppänen

Versio: 1.02
Julkinen
10. maaliskuuta 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, Sovellusraportti
Sivumäärä: 45
Tiedosto: sovellusraportti.tex

Tiivistelmä: Sovellusraportissa kuvataan Concept-projektin tuottaman aikataulun laatimissovelluksen toteutus ja tekniset valinnat.
Avainsanat: aikataulu, konferenssi, sovellusprojekti.

Versiohistoria

Versio	Päivämäärä	Muutokset	Tekijät
0.1	1.1.2006	Ensimmäinen hahmotelma rungosta.	TR
0.2	3.1.2006	Tekstiä lisätty, taulukko tietokannan muutoksista lisätty.	TR
0.3	4.1.2006	Tekstiä lisätty tietokantakomponentti-osioon.	TR
0.4	17.1.2006	Taulukoita luokkien attribuuteista lisätty.	TR
0.5	22.1.2006	Taulukoita täydennetty ja käyttöliittymäosio lisätty	AS, TR
0.6	23.1.2006	Tarkennuksia tietokantakomponentin ja tietorakenteen toimintaan	TR
0.7	24.1.2006	Jatkokehitystietoja lisätty	TR
0.8	2.2.2006	Jatkokehitystietoja tarkennettu, listätty konfliktinhallinta	TR
0.9	6.2.2006	Dokumentin rakennetta viimeistelty, kuvat tarkastettu ja päivitetty	TR
0.91	8.2.2006	Dokumentin viimeistely	TR
1.0	13.2.2006	Dokumentin viimeistely	TR
1.01	1.3.2006	Taulukko uusista tauluista lisätty	TR
1.02	5.3.2006	Virheellinen päiväys korjattu	TR

Tietoa projektista

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

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 toteutti syksyn 2005 aikana työasemakäyttöön aikataulunlaatimissovelluksen, joka tulee ensisijaisesti liitettäväksi ECSS07-kongressinhallintajärjestelmään. Tässä dokumentissa kuvataan CONCEPT-projektin tuottaman aikataulunlaatimissovelluksen toteutuksen tekniset yksityiskohdat, rakenne ja toiminta sekä lopullisen toteuksen yhteneväisyys sovellussuunnitelman kanssa. Luvussa 2 kuvataan sovellusraporttiin 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 toteutettujen komponenttien toimintaa. Luvussa 9 tarkastellaan sovellukselle asetettuja vaatimuksia ja niiden toteuttamista sekä jatkokehitystä ja tunnettuja ongelmia. Luku 10 sisältää yhteenvedon dokumentista.

2 Termit

Dokumentin aihealueen termejä ovat seuraavat:

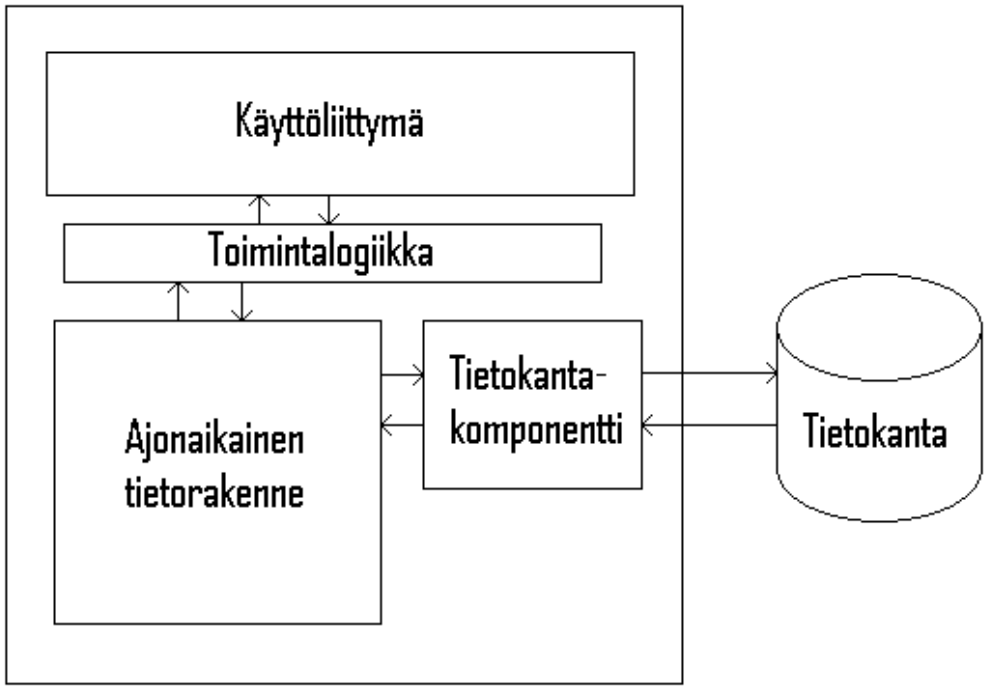
Konferenssi	Muutamasta päivästä viikkoon kestävä tapahtuma, jossa tutkijat tai heidän edustajan tulevat esittelemään tutkimustuloksiaan ja kuuntelemaan muiden tuloksia.
Kongressi	Ks. Konferenssi
Blokki	Blokki tarkoittaa tiettyä ajanjaksoa, esim. ma 10.12. klo 12:30 - 14:00.
ECSS	European College of Sport 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.

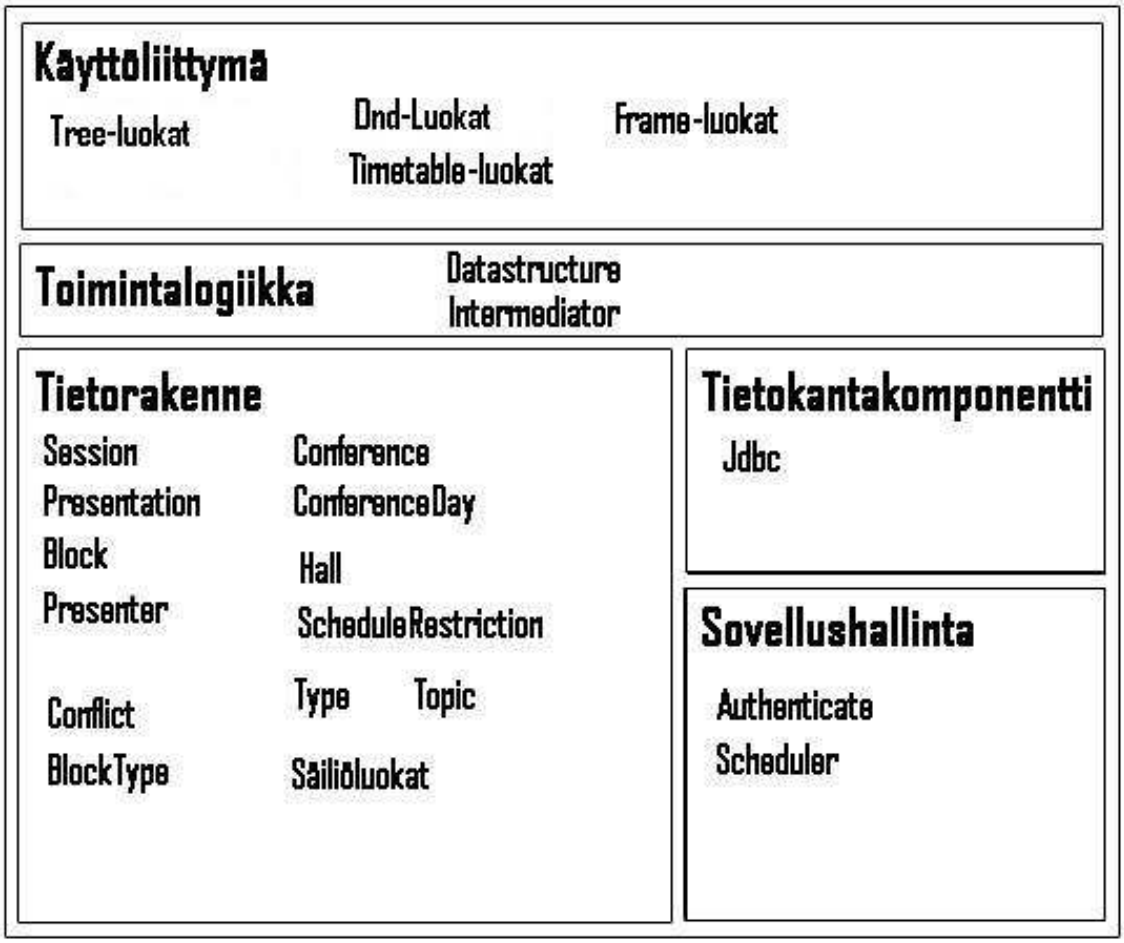
3 Yleiskatsaus

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

Kuvassa 3.2 esitetään sovelluksen arkkitehtuuri ja luokkajakauma. Käyttöliittymässä on runsaasti luokkia jokaisen taulukon ominaisuuksien määrittämistä varten, joten ne on kuvassa esitetty vain eri taulukoiden ryhminä. Tietorakenteessa on lisäksi säiliöluokat ConferenceDay, Block, Session, Hall, Presentation, Presenter ja ScheduleRestriction-luokille ja nämä on kuvassa 3.2 esitetty ryhmänä säiliöluokat.



Kuva 3.2: Kuvaus sovelluksen kokonaisrakenteesta ja luokkajakaumasta

3.1 Tietokanta

Kehitetyn aikataulutussovelluksen pohjana toimii myös Coma- ja ECSS07-projektien muodostavan konferenssinhallintajärjestelmän osana oleva tietokanta. Tietokannan rakenne ei kuitenkaan sellaisenaan riittänyt vastaamaan aikataulutussovelluksen asettamia vaatimuksia. Tietokantaan jouduttiin tekemään joitakin muutoksia ja lisäyksiä, joista osan toteutti tilaaja ja osan projektiryhmä. Tietokannan muutokset tai lisäykset eivät saaneet vaikuttaa muiden konferenssinhallintajärjestelmän osien toimintaan. Kehitetty sovellus käyttää tietokantaa aktiivisesti, mutta sovellus ei muokkaa kuin osaa tietokannan tauluista. Taulukossa 3.1 kuvataan sovelluksessa käytetyt tietokannan taulut sekä tieto, onko taulu lisätty tai onko sitä muutettu sovelluksen kehitysvaiheessa ja lisäksi tieto siitä, muokkaako sovellus tietokannan taulua. Taulukkoon merkittyjen projektiryhmän lisäysten lisäksi tilaajan toimesta kantaan

lisättiin taulut chairman, chairmen, specialcase, specialty ja restriction. CONCEPT-sovellusprojektin lisäämien taulujen yksityiskohtainen kuvaus on liitteessä B.

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

Taulukko 3.1: Sovelluksen käyttämät tietokannan taulut

Tietokannan tauluihin article, chairman, present, restriction, session, schedule, specialcase ja specialty tehtiin muutoksia. Article-taulun constraint-ominaisuutta `article_fk_s` muutettiin niin, että article-taulun `presented_in`-sarake muuttuu null-arvoon, mikäli siinä viitattu sessio poistetaan. Chairman-taulun constraint-ominaisuutta `chairman_fk_s` muutettiin siten, että chairman-taulun puheenjohtajan ja session yhteyden muodostava rivi poistuu, jos sessio poistetaan. Present-tauluun lisättiin integer-arvoinen `running_number`-sarake, joka ilmoittaa esityk-

sen esitysvuoron sessiossa. Restriction taulun `order_number`-saraketta muutettiin niin, että se sallii myös null-arvon. Lisäksi restriction-taulun constraint-ominaisuutta `restriction_fk_s` muutettiin niin, että poistettaessa rajoita specialcase-taulusta, poistetaan myös siihen rajoitteeseen viittaavat rivit restriction-taulusta. Session-tauluun lisättiin boolean-arvoinen `isready`-sarake, joka varastoi session valmiusmerkinnän sekä text-arvoinen `abbrev`-sarake, joka ilmoittaa session lyhenteen. Lisäksi session-taulun constraint-ominaisuutta `session_fk_b` muutettiin niin, että session ja blokin yhteyden muodostava arvo belongs muutetaan null-arvoon, mikäli siinä viitattu blokki poistetaan. Schedule-taulun sarake `title` poistettiin ja consrict-ominaisuutta `schedule_fk_s` muutettiin siten, että scheduletype taulun riviä ei voi poistaa, mikäli jokin schedule viittaa siihen. Specialcase-tauluun lisättiin sarake `session`, joka kuvaa sitä, minkä session aiheuttama kyseinen rajoitus mahdollisesti on. Sarake voi saada arvon null, jos rajoitus ei aiheudu sessiosta. Tauluun lisättiin myös kuvattua saraketta koskeva constraint-ominaisuus `specialcase_fk_ses`, joka poistaa kaikki sessioon viittavat rajoitukset, mikäli itse sessio poistetaan. Specialty-taulun `order_number`-saraketta muutettiin niin, että se sallii myös null-arvon. Lisäksi taulun constraint-ominaisuutta `specialty_fk_s` muutettiin siten, että jos taulussa viitattu rajoite poistuu, poistuvat myös siihen viittaavat specialty-taulun rivit.

3.2 Käyttöliittymä

Käyttöliittymän tarkoituksena on tarjota käyttäjälle monipuoliset ja intuitiiviset työkalut konferenssin aikatauluttamiseen. Kehitetty sovellus ei tarjoa automaattista aikataulutusta, vaan kätevän työkalun aikataulutuksen helpottamiseksi. Käyttöliittymä erotettiin muusta sovelluksesta toimintalogiikkaa hyödyntävän rajapinnan ansiosta. Näin käyttöliittymä ja sovelluksen toiminnallinen rakenne voidaan erottaa toisistaan, mikäli käyttöliittymä halutaan korvata toisella komponentilla tai liittää johonkin toiseen toiminnalliseen tietorakenteeseen. Käyttöliittymä toimii ”ikkunana” ajonaikaiseen tietorakenteeseen, eikä käyttöliittymä varastoi mitään tietoa.

3.3 Toimintalogiikka

Toimintalogiikka toimii lopullisena rajapintana käyttöliittymälle. Toimintalogiikan tehtävänä on paketoida ajonaikaisen tietorakenteen tarjoamat pienemmät toimin-

nallisuudet jonkun tietyn toiminnon toteuttavaksi kokonaisuudeksi. Näitä kokonaisuuksia toimintalogiikka tarjoaa metodeina ylöspäin käyttöliittymälle.

3.4 Ajonaikainen tietorakenne

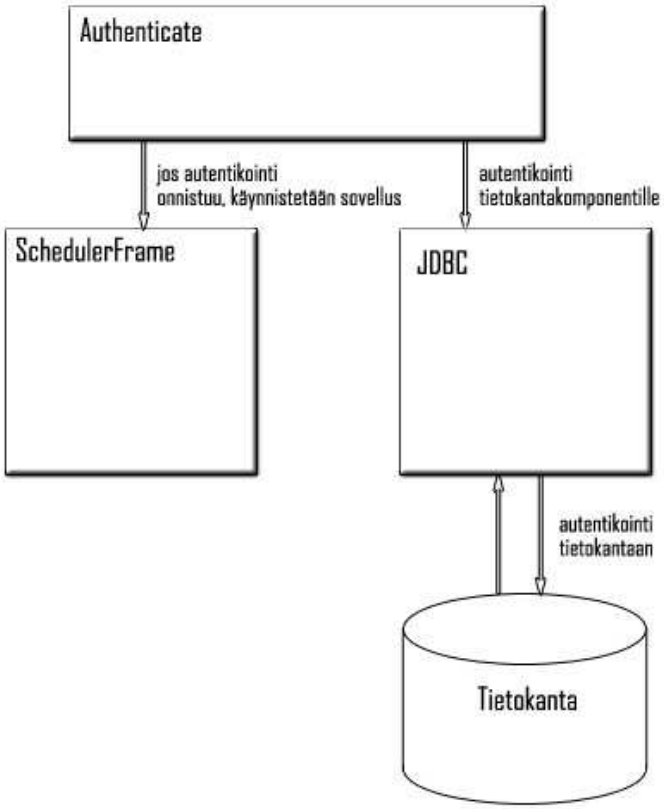
Ajonaikainen tietorakenne simuloi tietokannan relaatiorakennetta olioiden välisellä viiterakenteella. Näin mahdollistetaan dynaaminen aikataulun muokkaus. Tietokannasta tuotu tieto sijoitetaan oliopohjaiseen tietorakenteeseen korvaten tietokannan taulut olioryhmillä ja relaatiot viiteyhteyksillä. Tämä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

3.5 Tietokantakomponentti

Tietokantakomponentti hoitaa sovelluksen vuorovaikutuksen käytettävään PostgreSQL-tietokantaan. Se tulkitsee käyttöliittymältä välittyneet kyselyt SQL-muotoon, jossa ne välitetään eteenpäin tietokannalle. Tietokantakomponentin rooli korostuu etenkin hakuja tehdessä, sillä käyttöliittymän hakutoiminnot on toteutettu tietokantaan tehtävillä hauilla. Tietokantakomponentti välittää tietoa myös tietokantaan päin tallennettaessa luotu aikataulu.

4 Sovelluksen autentikointi ja käynnistyminen

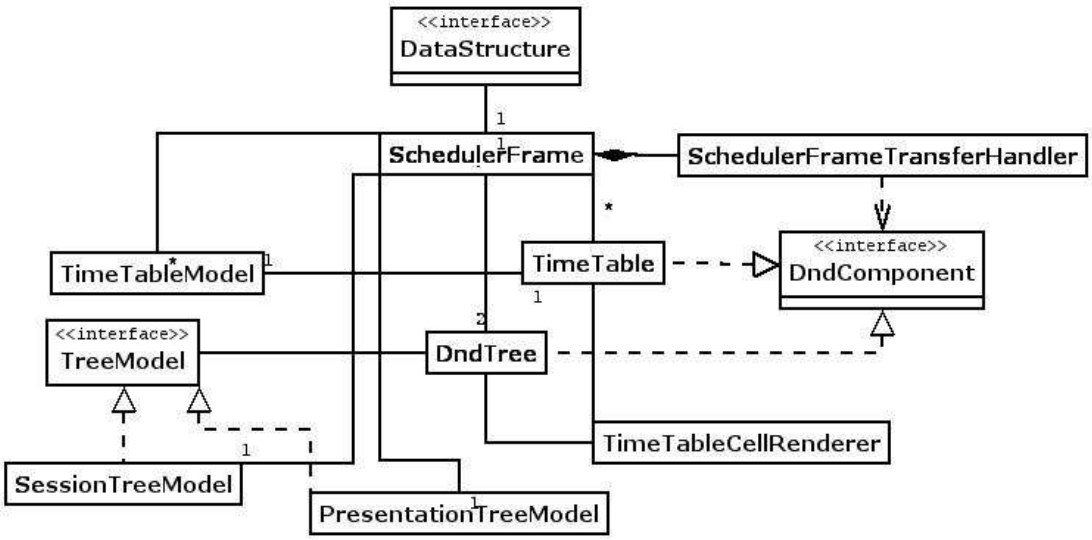
Sovelluksen autentikointia ja käynnistymistä varten on luotu kaksi luokkaa. Varsinaisen sovelluksen pääohjelman sisältää Scheduler-luokka, joka käynnistää sovelluksen. Scheduler-luokka huolehtii autentikointiprosessin ja itse sovelluksen käyttöliittymän käynnistämisestä. Autentikointi tapahtuu Authenticate-luokassa, joka huolehtii käyttäjän antaman käyttäjätunnuksen ja salasanan siirtämisestä tietokantakomponentille, joka puolestaan pyrkii ottamaan yhteyden tietokantaan näitä tunnuksia käyttäen. Authenticate-luokka avaa ikkunan, jossa kysytään käyttäjältä tietokannan osoite ja käytettävä käyttäjätunnus sekä salasana kyseiseen tietokantaan. Autentikointi-ikkunassa käytetään tietojen esitäyttöä ennalta säädetyn tekstitiedoston sisällön mukaisesti. Tekstitiedostoon voi kirjata esimerkiksi useimmin käytetyn tietokannan osoitteen, jolloin käyttäjä välttyy toistuvalta työltä yhteyttä uudelleen otettaessa. Mikäli yhteys onnistuu, luetaan tietokannan taulujen sisältö olioiden luomista varten ja käynnistetään sovellus kuvan 4.1 mukaisesti. Yhteys säilyy tietokantakomponentin hallussa kunnes se suljetaan. Mikäli käyttäjätunnus tai salasana olivat virheellisiä, tietokantaa ei löydetty annetusta osoitteesta tai yhteys annettuun tietokantaan on poikki, palautuu tästä aiheutunut poikkeus Authenticate-luokan käsittelyyn, joka avaa virhetilanteesta kertovan dialogin käyttäjälle. ja autentikointiprosessi aloitetaan uudelleen. Salasana kulkee tietokantakomponentin ja Javan valmiin Jdbc-komponentin kautta salattuna tietokantaan.



Kuva 4.1: Autentikointiprosessi

5 Käyttöliittymä

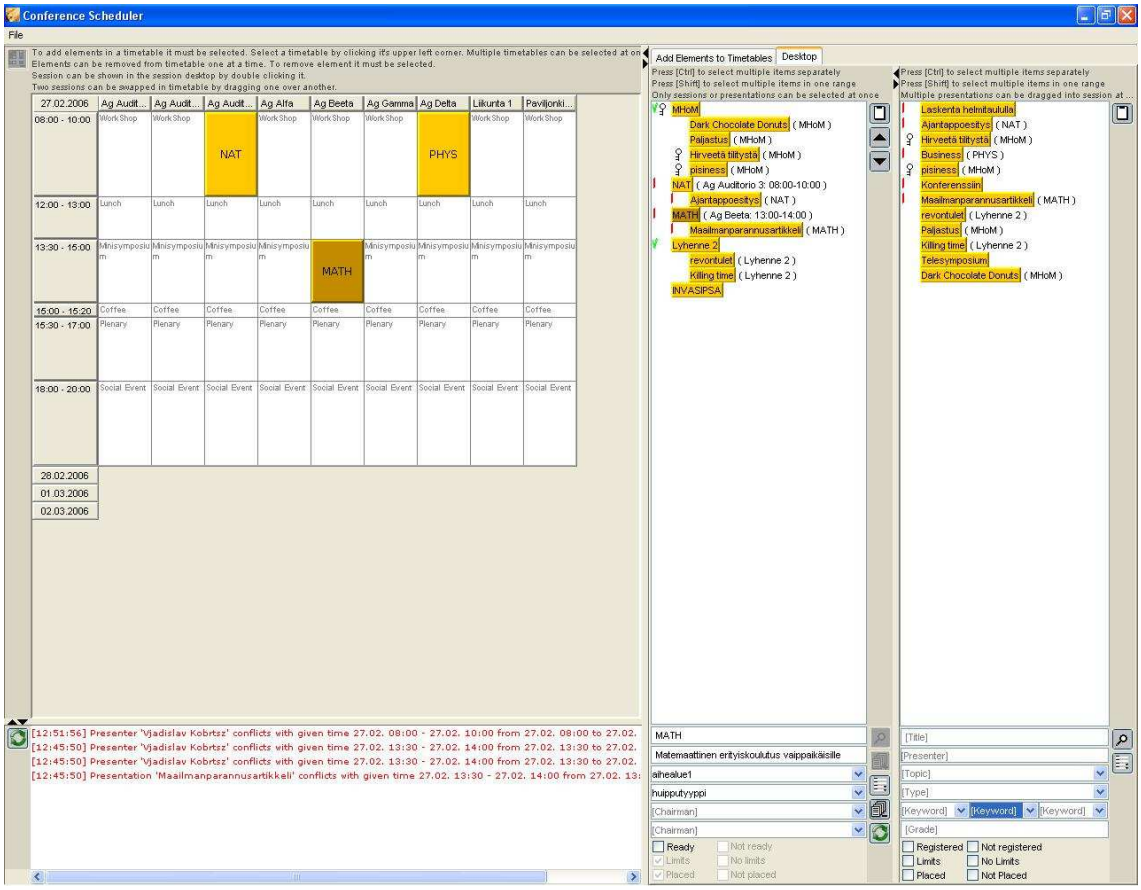
Käyttöliittymän toiminta on rakennettu suurimmaksi osaksi SchedulerFrame-luokkaan, joka määrittelee sovelluksen pääikkunan. SchedulerFrame-luokka muodostaa myös käyttöliittymän yhteyden tietorakenteeseen DataStructure-rajapinnan kautta. Kuvassa 5.1 on esitetty SchedulerFramen lisäksi muut tärkeimmät käyttöliittymän luokat, jotka edustavat aikatauluja (TimeTable) sekä sessiopuuta ja esityslistaa (DndTree). Lisäksi kaaviossa on esitetty aikataulujen, sessiopuun ja esityslistan tietosisällöstä (TimeTableModel, SessionTreeModel, PresentationTreeModel) ja piirtämisestä(TimeTableCellRenderer) vastaavaa luokat. Kaaviossa on esitetty myös SchedulerFrame:n sisäinen luokka SchedulerFrameTransferHandler, joka vastaa drag&drop -toiminnoista käyttöliittymässä. SchedulerFrameTransferHandler kerää siirrettävän datan ja huolehtii siirron oikeellisuusta komponenttien DndComponent-rajapintaa käyttämällä. Käyttöliittymän luokkien metodeista tarkemman kuvauksen tarjoaa Javadoc-dokumentaatio liittessä C.



Kuva 5.1: Käyttöliittymän luokkakaavio

Sovelluksen käyttöliittymä on jaettu keskeltä kahtia (5.2). Vasemmassa osassa ovat aikataulut ja konfliktilista. Oikealla puolestaan on lehtiö, jonka toiselta välilehdeltä (”Add elements to timetables”) voidaan lisätä elementtejä aikatauluun ja toisella välilehdellä (”Desktop”) on työpöytä, jossa luodaan sessiot ja lisätään esitykset

niihin. Työpöydällä tapahtuu myös sessioiden ja esitysten haku. Käyttöliittymässä sessiot ja esitykset esitetään värikoodeilla, jotka kuvaavat aihepiirejä, joten käyttäjän on helppo tunnistaa samaan aihepiiriin kuuluvat esitykset ja sessiot.

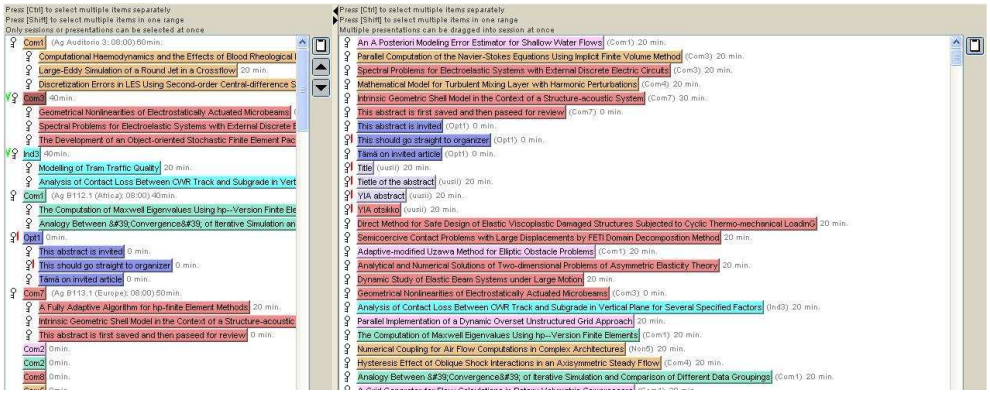


Kuva 5.2: Pääikkuna: aikataulu ja aikataulun luominen

5.1 Työpöytä

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. Sen vasemmalla puolella käsitellään sessioita ja oikealla puolella esityksiä. Sessiot esitetään työpöydällä puurakenteessa, jossa kunkin session alla ovat kyseiseen sessioon sijoitetut esitykset. Esityksiä käsitellään sessiopuun vieressä olevan esityslistan avulla.(5.3) Työpöydän alaosassa on kentät yksittäisten sessioiden ja esitysten tietoja varten. Työpöytä antaa käyttäjälle mahdol-

lisuuden rajata tietomäärästä haluamansa osan käsiteltäväksi, sillä työpöydillle voi hakea haluamansa sessiot ja esitykset, sekä poistaa tarvittaessa työpöydiltä haluamansa objektit. Valittujen objektien poistaminen työpöydiltä tapahtuu niiden yläosissa, oikealla puolella, olevilla painikkeilla. Hakutoimonnoista kerrotaa tarkemmin kohdissa ”Sessiokentät ja toiminnot” sekä ”Esityskentät ja toiminnot”. Käsittelyn avustamiseksi sessiopuussa ja esityslistassa on myös aputoiminto, jonka avulla niistä voidaan helposti löytää objekteja alkukirjaimen perusteella. Painamalla Shift-näppäintä ja haluttua kirjainnäppäintä sessiopuussa tai esityslistassa, valinta siirtyy listassa alsapäin seuraavaan objektiin, joka alkaa kyseisellä kirjaimella.



Kuva 5.3: Sessiopuu ja esityslista

5.1.1 Esitysten ja sessioiden sijoitus raahaamalla

Esitysten sijoittaminen sessioihin, sessioiden sijoittaminen aikatauluun, sekä näiden käänteisoperaatiot, eli sijoitusten poistamiset, tapahtuvat raahaamalla objekteja työpöytien ja aikataulun välillä. Sessioita voi raahata aikatauluun tai pois aikataulusta vain yhden kerrallaan, mutta esityksiä voi raahata sessioon sessiopuussa tai pois sessiosta useita kerralla. Sessiopuussa esityksiä voi siirtää myös sessiosta toiseen. Raahattaessa kursori muuttuu kieltomerkiksi, jos objektien pudottaminen alla olevaan kohteeseen ei ole sallittua. Onnistuneen raahauksen jälkeen raahattu objekti poistuu lähtöpaikastaan.

Kun esitys sijoitaan raahaamalla sessioon, se sijoitetaan tällöin sessioon kyseisen session viimeiseksi esitykseskis. Esityksen paikkaa sessiossa voi vaihtaa valitsemalla esitys sessiopuussa ja siirtämällä sitä ylös- tai alaspäin sessiopuun oikealla puolilla olevilla nuolipainikkeilla.

Työpöydän objektien raahaamista tai poistamista varten haluttujen objektien täytyy

olla valittuna. Työpöydältä voi valita useita objekteja pitämällä alhaalla Ctrl (erilliset valinnat) tai Shift (yksi jatkuva valinta) -näppäimiä. Sessiopuusta voi valita kerrallaan joko esityksiä tai sessioita, mutta ei molempia yhtäaikaa, jotta raahaustoimintojen merkitys voidaan tulkita oikein.

5.1.2 Symbolit työpöydällä

Työpöytien objekteissa käytetään symboleita, jotka antavat niistä tietoa (5.3). Symboleista punainen huutomerkki kertoo, että session tai esityksen sijoittamisella aikatauluun on rajoituksia, vihreä merkintä kuvaa session valmiusmerkintää ja avainsymboli kertoo rekisteröitymättömistä käyttäjistä. Valmiusmerkintää käytetään siis vain sessioiden yhteydessä, mutta huutomerkki ja avain esiintyvät myös esitysten yhteydessä. Esitysten kohdalla huutomerkki tarkoittaa, että esituksellä tai jollain sen esittäjistä on rajoituksia aikataulutuksen suhteen. Avainsymboli tarkoittaa, että jokin sen esityksen esittäjistä ei ole rekisteröitynyt järjestelmään. Sessioilla voi olla myös rajoitteita, mutta niiden yhteydessä huutomerkki voi kertoa myös sessioon sijoitettujen esitysten rajoitteista. Esityksen rajoitemerkintä siis siirtyy myös sessiolle, johon se sijoitetaan. Avainsymboli käyttäytyy myös samalla tavoin, eli esityksen avainsymboli siirtyy myös sessioon.

5.1.3 Sessiokentät ja toiminnot

Vasemmalla puolella alaosassa on kentät session tietoja varten. Session tietoja ovat lyhenne, otsikko, aihe, tyyppi, puheenjohtajat ja valmiusmerkintä. Kenttiä käytettän session tietojen täyttämiseen sessiota lisättäessä, session tietoja päivitettäessä ja sessioita haettaessa. Hakua tehtäessä hakuehdoksi voidaan antaa kenttiin edellä mainittujen tietojen lisäksi tieto, onko sessio sijoitettu aikatauluun, tai kuuluuko sen esityksiin esittäjiä, jotka eivät ole rekisteröityneita käyttäjiä. Hakua tehtäessä lyhenteessä ja otsikossa voi käyttää vapaata sanahakua ja puheenjohtajista huomioidaan hakuehtona vain ensimmäinen. Kenttien vieressä olevilla painikkeilla voidaan suorittaa haku, lisätä uusia sessioita, tyhjentää kentät ja poistaa kentissä esillä oleva sessio tai muokata sen tietoja. Käyttöliittymässä toimintoja vastaavat painikkeet on sijoitettu kyseisessä järjestyksessä ylhäältä lukien (5.4).

Kun uusi sessio lisätään, se ilmestyy sessiopuuhun. Sessiopuusta taas voidaan valita hiiren oikealla painikkeella sessio, jonka tiedot halutaan näytettäväksi tai muokattavaksi kenttiin. Valittu sessio myös aukaistaan, jolloin sen esitykset tulevat myös

näkyviin session alle. Esitykset saadaan piilotettua, kun suljetaan avattu sessio valitsemalla se uudelleen. Avaamisen ja sulkemisen voi tehdä myös oikealla ja vasemmalla nuolinäppäimellä.

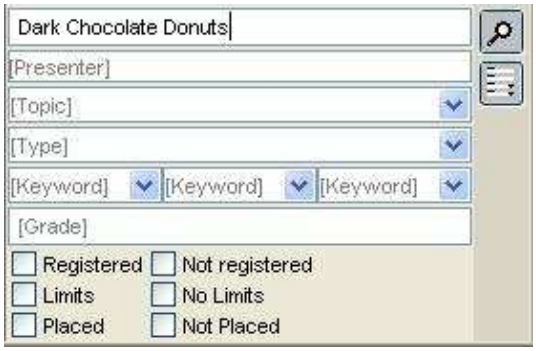
Kuva 5.4: Sessiokentät ja painikkeet

5.2 Esityskentät ja toiminnot

Oikealla työpöydän alaosassa on puolestaan kentät esityksen tietojen näyttämiseen ja hakuehtojen antamiseen, kun halutaan hakea esityksiä. Esityksen tietoja ovat otsikko, esittäjä(t), aihe, tyyppi, kolme avainsanaa, arvosana ja tiedot onko esitys sijoitettu sessioon, onko esityksellä tai sen esittäjillä rajoituksia aikataulutuksen suhteen tai onko esityksellä käyttäjiä, jotka eivät ole rekisteröityneet. Myös esityksen tiedot nähdään kentissä, kun esitys valitaan työpöydältä hiiren oikella painikkeella. Kenttien vieressä, oikealla puolella, on painikkeet, joilla suoritetaan haku ja tyhjennetään kentät. Haku suoritetaan ylemmällä ja tyhjennys alemmalla painikkeella (5.5).

5.3 Aikataulun luominen

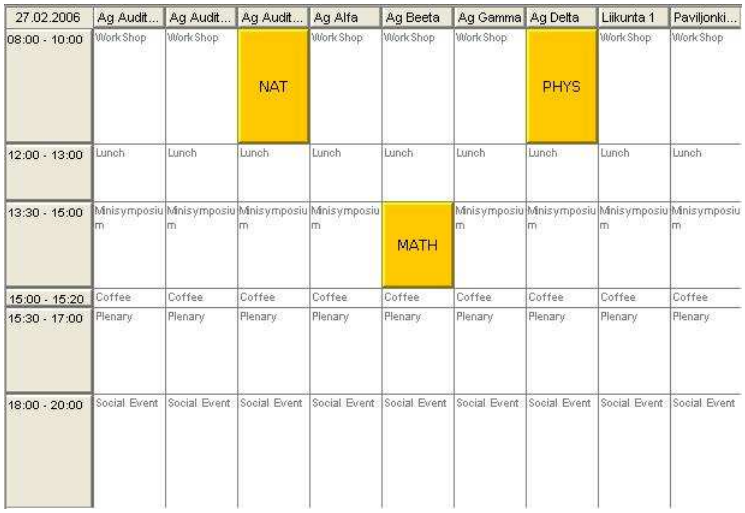
Kun tietokantayhteys on saatu, aikataulussa on valmiina aikataulujen alkiot (5.6) kaikille konferenssin päiville. Näistä käyttäjä luo aikataulut (5.7) lisäämällä elementtejä, eli saleja ja blokkeja. Elementin lisäämiseksi päivän aikataulu täytyy valita hiiren oikealla painikkeella aikataulun vasemman ylänurkan solu, jossa näkyy päivämäärä. Valitun aikataulun tunistaa erilaisista reunoista, jolloin aikataulu näyttää



Kuva 5.5: Esityskentät ja painikkeet

painuneen alas. Koska päiviä voidaan valita useita kerralla, voidaan myös elementtejä lisätä useille päiville yhdellä kertaa. Lisäys tapahtuu aina kaikille valituille päiville. Jos jokin elementti halutaan poistaa päivältä, se valitaan aikataulusta ja poistetaan painikkeella, joka on aikataulujen yläpuolella, vasemmalla (5.6). Elementtejä voi poistaa vain yhden kerrallaan.

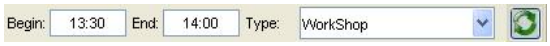
Kuva 5.6: Alkiot



Kuva 5.7: Aikataulu yhdelle konferenssin päivälle

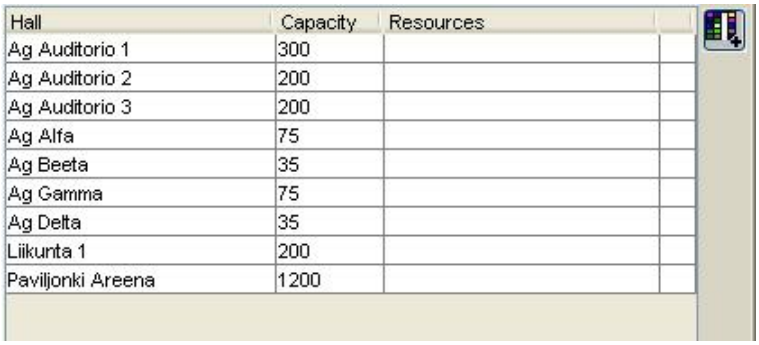
Blokkeja voidaan lisätä aikatauluun yksi kerrallaan. Blokin lisäämiseksi sille täytyy antaa aloitus- ja lopetusaika muodossa hh:mm, sekä tyyppi. Blokki lisätään painikkeella, jolloin se ilmestyy aikatauluun. (5.8.) Aikataulussa blokit järjestyvät aloitusajan mukaisesti pienimmästä suurimpaan (5.7). Blokin tietoja voidaan päivittää valitsemalla blokki aikataulusta, jolloin sen tiedot ilmestyvät muokattavaksi samoihin kenttiin, joihin myös lisättävien blokkien tiedot syötetään. Tietojen päivittäminen tapahtuu myös samalla painikkeella kuin blokkien lisääminenkin. (5.9.)

Kuva 5.8: Blokin lisääminen



Kuva 5.9: Blokin tietojen päivittäminen

Saleja voidaan lisätä useita kerralla. Lisättävät salit valitaan salilistasta ja ne lisätään aikatauluihin listan vieressä olevalla painikkeella (5.10). Aikatauluun voi lisätä kunkin salin vain kerran. Jos aikatauluun yritetään lisätä sali, joka on jo lisätty, lisäystä ei tehdä. Aikataulussa näkyvät harmaina ne solut, jolloin sali ei ole käytettävissä.



Hall	Capacity	Resources
Ag Auditorio 1	300	
Ag Auditorio 2	200	
Ag Auditorio 3	200	
Ag Alfa	75	
Ag Beeta	35	
Ag Gamma	75	
Ag Delta	35	
Liikunta 1	200	
Paviljonki Areena	1200	

Kuva 5.10: Salilista

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. 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. Aikataulutetun session tietoja voidaan tarkastella kaksoisklikkaamalla hiiren oikealla painikkeella haluttua sessiota. Tällöin sessio ilmestyy sessiopuuhun avattuna, eli myös sen esitykset ovat nähtävissä, ja lisäksi session muut tiedot esitetään sessiopuun alla olevissa kentissä.

5.5 Konfliktilista

Aikataulutuksesta aiheutuneet konfliktit näkyvät alhaalla ikkunan vasemmassa osassa konfliktilistassa. Listan oikealla puolella on painike, jolla konfliktilista voidaan päivittää ajantasaiseksi. (5.11.) Päivityksessä listasta poistuvat ne konfliktit, joita ei ole enää aikataulussa.

Kuva 5.11: Konfliktilista

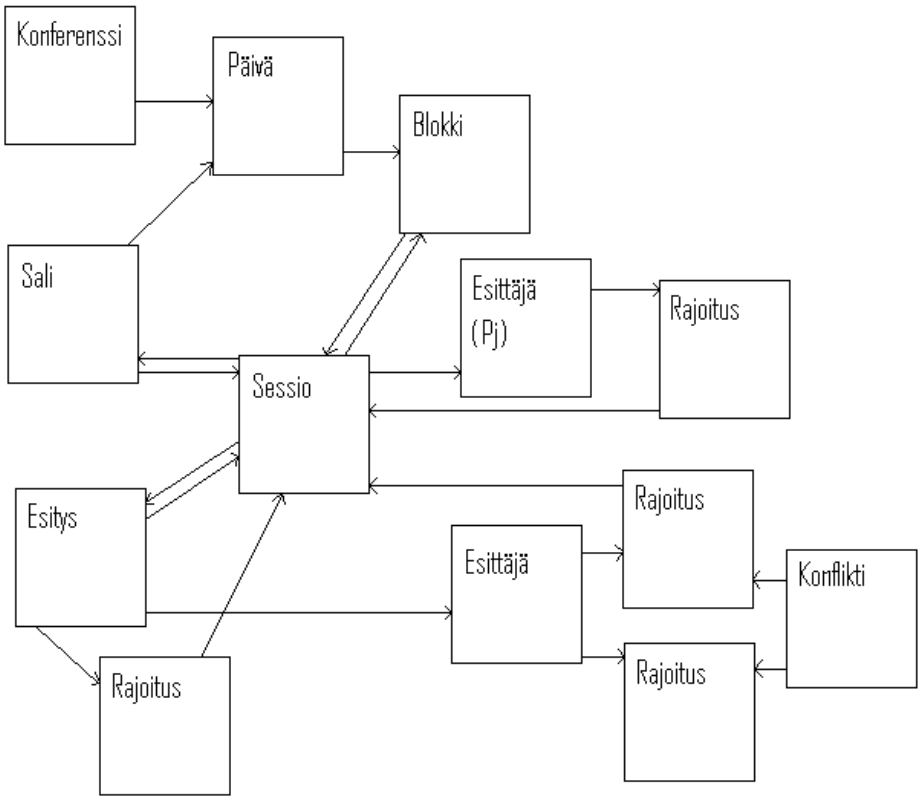
6 Ajonaikainen tietorakenne

Tietorakenne koostuu Session, Presentation, Presenter, ConferenceDay, Block, Conference, Hall, ScheduleRestriction, Conflict, Type, Topic, Preference ja BlockType luokista sekä Sessions, Presentations, ConferenceDays, Blocks, Halls ja ScheduleRestrictions säiliöluokista. Säiliöluokkien nimeämisessä on käytetty yksittäisen elementin nimen monikkoa. Säiliöluokat pitävät sisällään viiterakenteita toiminnallisten luokkien olioista ja tarjoavat lisäys- ja poistopalveluita rakenteiden ylläpitämistä varten.

Tietorakenteen luokkarakennetta on esitelty kuvassa 6.1. Konferenssiin kuuluu konferenssin kestosta riippuva määrä päiviä, jotka käyttäjällä on mahdollisuus jakaa valitsemansa pituisiin blokkeihin. Päiville voidaan myös määrittää saleja. Sessiolle voidaan määrittää blokki ja sali, jolloin sessio on aikataulussa. Tällöin kaikille session esityksille ja esittäjille syntyy rajoiteolio session aikataulutustietojen mukaan. Kuvassa 6.1 on tilanne, jossa sessioon sijoitetulla esityksellä on esittäjä, jolla on olemassa henkilökohtainen aikataulurajoitus. Lisäksi esittäjällä on session aikataulutuksesta aiheutunut rajoitus, jolla on myös linkki sen aiheuttaneeseen sessioon. Näiden kahden rajoituksen ajat leikkaavat ja muodostuu konflikti. Kuvan 6.1 mukaisesti, blokilla ja salilla on listat niihin liittyvistä sessioista. Sessiolla puolestaan on lista siihen liittyvistä esityksistä ja puheenjohtajista. Esityksellä on lista siihen liittyvistä esittäjistä ja viite sessioon, mikäli esitys on sellaiseen sijoitettu. Samasta kuvasta huomataankin selvästi, että sessio on keskeisin elementti sovelluksen aikataulutustoiminnoissa. Tietorakenteen luokkien metodeista tarkemman kuvauksen tarjoaa Javadoc-dokumentaatio liittessä C.

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. Sovellus käynnistettäessä rakenne on tyhjä ja ensimmäisenä luodaan tietokantakomponentin avustuksella Conference-olio vastaamaan käyttäjän valitsemaa tietokannan Congress-taulua. Conference-luokka rakentaa sille syntyessään annetun aikavälin mukaan itselleen oikean määrän konferenssipäiviä.



Kuva 6.1: Tietorakenteen luokkarakenne

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. Conference-luokassa määritellään myös DateFormat-luokan esiintymä, joka formatoi päivien Date-luokkien esiintymien tulostusasun. Conference-luokka määrittää konferenssin aihealueet Topic-luokan esiintymiä sisältävällä allTopics-rakenteella, blokkityypit BlockType-luokan esiintymiä sisältävällä allBlockTypes-rakenteella, kaikki avainsanat merkkijonoja sisältävällä keywords-rakenteella sekä sessio- ja esitystyypit Type-luokan esiintymiä sisältävällä allTypes-rakenteella. Kaikki edellä mainitut rakenteet on toteutettu käyttäen Javan Vector-tietorakennetta.

Attribuutti	Tyyppi	Tehtävä
days	ConferenceDays	Konferenssin päivät sisältävä rakenne
conferencePreference	Preference	Viite konferenssin asetuksiin
title	String	Konferenssin otsikko
abbrev	String	Konferenssin lyhenne
keywords	Vector	Konferenssin avainsanat sisältävä rakenne
allTypes	Vector	Konferenssin sessio-/esitystyypit sisältävä rakenne
allTopics	Vector	Konferenssin aihealueet sisältävä rakenne
allBlockTypes	Vector	Konferenssin blokkityypit sisältävä rakenne
allSessions	Sessions	Konferenssin sessiot sisältävä rakenne
allPresentations	Presentations	Konferenssin esitykset sisältävä rakenne
allPresenters	Presenters	Konferenssin esittäjät ja puheenjohtajat sisältävä rakenne
allHalls	Halls	Konferenssin salit sisältävä rakenne
conferenceStart	long	Konferenssin alkuaika millisekunteina

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. Konferenssipäivälle ei ole vastinetta tietokannan tauluissa vaan päivämäärät sisältyvät kaikkien aikataulutettavien elementtien omiin aikatietoihin. Konferenssipäivää kuvaavan olion attribuutit ja Conference-luokan metodit tarjoavat mahdollisuuden sekä indeksipohjaiseen että millisekuntipohjaiseen päivien käsittelyyn. ConferenceDay-luokassa määritellään myös Dateformat-luokan esiintymä, joka formatoi päivien Date-luokkien esiintymien tulostusasun ConferenceDay-luokan toString-metodia tai getDateString-metodia käytettäessä.

Attribuutti	Tyyppi	Tehtävä
dateinms	long	Päiväys millisekunteina
date	Date(java.sql.Date)	Päivämäärää vastaava olio
blocks	Blocks	Päivän blokit sisältävä rakenne
halls	Vector	Päivän salit sisältävä rakenne
parent	ConferenceDays	Viite päivän sisältävään rakenteeseen
dateFormat	DateFormat	Päivämäärien muodon määrävä DateFormat-instanssi

6.3 Block

Attribuutti	Tyyppi	Tehtävä
startTime	long	Blokin alkuaika millisekunteina
endTime	long	Blokin päättymisaika millisekunteina
id	int	Blokin id-numero tietokannan rivien tunnistamista varten
blockType	BlockType	Viite blokin tyyppi vastaavaan BlockType-instanssiin

Block on aikataulutusblokkia kuvaava luokka. Blokki on aikataulusuunnittelun apukeino, joka mahdollistaa koko aikataulutettavan päivän jonkin aikavälin aikataulutuksen sitomisen samaksi kaikkiin saleihin. Luokan ilmentymä sisältää yksilöllisen kokonaislukutunnuksen id, alkuajan ja loppuajan millisekunteina, viitteen Sessions-luokan ilmentymään sisältäen blokkiin sijoitetut sessiot ja viitteen kyseisen blokin säiliöluokan ilmentymään. Blokkien tiedot vastaavat tietokannan schedule-taulun tietoja. Block-luokassa määritelty toString-metodi tulostaa blokissa käytetyn BlockType-luokan esiintymän toString-metodin palauttaman merkkijonon.

6.4 Session

Session on konferenssin sessiota kuvaava luokka. Sessio on sovelluksen keskeisin aikataulutusosa muodostaen yhteyden esitysten ja aikataulun välille. Luokan ilmen-

Attribuutti	Tyyppi	Tehtävä
isReady	boolean	Session valmiusmerkinnästä kertova boolean-muuttuja
scheduleLink	Block	Viite session blokkiin mikäli sessio aikataulutettu, muutoin null
id	int	Session id-numero tietokannan rivien tunnistamista varten
parent	Sessions	Viite session sisältävään rakenteeseen
hallLink	Hall	Viite session saliin mikäli sessio aikataulutettu, muutoin null
startTime	long	Session alkuaika millisekunteina
endTime	long	Session päättymisaika millisekunteina
type	Type	Viite session tyyppiä vastaavaan Type-instanssiin
topic	Topic	Viite session aihealuetta vastaavaan Topic-instanssiin
title	String	Session otsikko
abbrev	String	Session lyhenne
chairmen	Presenters	Session puheenjohtajat sisältävä rakenne
presentations	Presentations	Session esitykset sisältävä rakenne
accumulatedrestrictions	ScheduleRestrictions	Session kertyneet rajoitukset sisältävä rakenne

tymä 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. Sessioiden tiedot vastaavat tietokannan session-taulua. Sessiotasolle on koottu rajoitteiden vertailurajapinta Block-luokan esiintymien ja sessioiden sisältämien rajoitteiden välille. Vertaamalla session sisältämien rajoitteiden aikavälejä sessioon linkitetyn blokin aikavälin kanssa saadaan aikataulutusrajoitteiden aiheuttamat konfliktit. Verrattaessa session kertyneitä rajoitteita sessioon linkitetyn salin

kanssa saadaan salirajoitteiden aiheuttamat konfliktit. Nämä konfliktit muodostavat sovelluksen koko mitattavan konfliktiavaruuden. Session vertailumetodit vertailevat rajoituksia käyttäen ScheduleRestrictions-säiliöluokan vertailumetodeja ja itse aikavälien vertailu tehdään ScheduleRestriction-luokan compare-metodilla.

6.5 Hall

Attribuutti	Tyyppi	Tehtävä
capacity	int	Salin kapasiteetti
name	String	Salin nimi
accessories	String	Salin varustus
restrictions	ScheduleRestrictions	Salin rajoitukset sisältävä rakenne
id	int	Salin id-numero tietokannan rivien tunnistusta varten
sessions	Sessions	Salin sessioviitteet sisältävä rakenne.
info	String	Salin lisätiedot

Hall on konferenssin salia kuvaava luokka. Salit tuodaan sovellukseen tietokannasta, eikä niitä muokata sovelluksesta. Luokan ilmentymään sisältyy salin kapasiteetti, nimi, varustus, viiterakenne salissa järjestettävistä sessioista, 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. Salien tiedot vastaavat tietokannan hall-taulua.

6.6 Presentation

Presentation on konferenssin esitystä kuvaava luokka. Esitykset sovellus hakee tietokannasta, eikä niitä muokata sovelluksessa. Luokan ilmentymään sisältyy esityksen aihe, nimi, kesto minuutteina, annettu arvosana, viite Presenters-luokan ilmentymään sisältäen kaikki esitykseen liittyvät esittäjät, viitteen esityksen aihealuetta vastaavaan Topic-luokan ilmentymään, viite esityksen tyyppiä vastaavaan Type-luokan vastaavaan ilmentymään rakenne esityksen avainsanoista, ja viite kyseisen

Attribuutti	Tyyppi	Tehtävä
presenters	Presenters	Esityksen esittäjien viitteet sisältävä rakenne
scheduleLink	Session	Linkki esityksen sisältävään sessioon
keywords	Vector	Esityksen avainsanat sisältävä rakenne
isInvited	boolean	Boolean merkintä siitä onko esitys kutsuttu
topic	Topic	Viite esityksen aihealuetta vastaavaan Topic-instanssiin
type	Type	Viite esityksen tyyppiä vastaavaan Type-instanssiin
title	String	Esityksen otsikko
duration	int	Esityksen kesto minuutteina
grade	int	Esityksen arvosana
parent	Presentations	Viite esityksen sisältävään rakenteeseen
id	int	Esityksen id-numero tietokannan rivien tunnistusta varten
restrictions	ScheduleRestrictions	Esityksen omat rajoitukset sisältävä rakenne
accumulatedrestrictions	ScheduleRestrictions	Esityksen kerääntyneet rajoitukset sisältävä rakenne

esityksen säiliöluokan ilmentymään. Presentation-luokka tarjoaa metodit esitysten muodostamiseen tietokannan tietojen perusteella ja esitysten linkittämiseen sopiviin sessioihin. Esitysten tietoja ei sovelluksessa muuteta, ainoastaan esitysten viitesuhteet muuttuvat sovelluksen käytön seurauksena. Esitysten tiedot vastaavat tietokannan article-taulua.

6.7 Presenter

Presenter on konferenssin esittäjää kuvaava luokka. Konferenssin esittäjät tuodaan tietokannasta ja ne on sovelluksen käytön alkaessa jo sidottu vastaaviin esityksiin. Sovelluksen yhteydessä sessioiden puheenjohtajaviitteet osoittavat Presenter-luokan ilmentymiin. Luokan ilmentymä sisältää esittäjän nimen, tiedon onko esittäjä puheenjohtaja, viiterakenteen esittäjän esitysvelvollisuuksista, viiterakenteen esittäjän

Attribuutti	Tyyppi	Tehtävä
name	String	Esittäjän nimi
isChairman	boolean	Boolean-muuttuja, joka kertoo onko esittäjä puheenjohtaja
parent	Presenters	Viite esittäjän sisältävään rakenteeseen
id	int	Esittäjän id-numero tietokannan rivien tunnistusta varten
restrictions	ScheduleRestrictions	Esittäjän rajoitukset sisältävä rakenne
presenterduties	Vector	Esittäjän esitysvelvollisuudet sisältävä rakenne
chairmanduties	Vector	Esittäjän puheenjohtajuudet sisältävä rakenne
maximumcmduties	int	Esittäjän puheenjohtajuuksien maksimimäärä
cmtopic	Topic	Viite esittäjän puheenjohtajuuden aihealuetta vastaavaan Topic-instanssiin
registered	boolean	Boolean-muuttuja, joka kertoo onko esittäjä rekisteröitynyt konferenssiin

puheenjohtajuusvelvollisuuksista, viitteen esittäjän puheenjohtajuuden aihealuetta vastaavaan Topic-luokan ilmentymään, tiedon onko esittäjä rekisteröitynyt konferenssiin, puheenjohtajuusvelvollisuuksien maksimimäärän 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. Ainoastaan puheenjohtajien viitesuhteet muuttuvat sovelluksen käytön seurauksena. Esittäjien tiedot löytyvät tietokannan users-taulusta. Taulusta luetaan ainoastaan puheenjohtajat ja esittäjät, joita molempia mallinnetaan Presenter-luokan ilmentymillä. Presenter-luokan cmtopic-attribuutti ilmaisee esittäjän puheenjohtajuuden aihealueen ja on sidottu Presenter-luokan isChairman-attribuuttiin, joka kertoo onko esittäjä puheenjohtaja. Mikäli puheenjohtajuuden aihealue muutetaan eroamaan null-arvosta, vaihtuu myös isChairman-muuttuja true-arvoon.

6.8 ScheduleRestriction

Attribuutti	Tyyppi	Tehtävä
parent	ScheduleRestrictions	Viite rajoituksen sisältävään rakenteeseen
begins	long	Rajoituksen alkuaika millisekunteina
ends	long	Rajoituksen päättymisaika millisekunteina
id	int	Rajoituksen id-numero tietokannan rivien tunnistusta varten
dummy	boolean	Boolean-muuttuja, joka kertoo onko rajoitus synnytetty vain vertailuprosessin helpottamiseksi
rcbsLink	Session	Linkki rajoituksen aiheuttaneeseen sessioon, mikäli rajoite on session aiheuttama
datef	DateFormat	Rajoitteen päivämäärän muodon määräävä DateFormat-instanssi

ScheduleRestriction-luokka kuvaa kaikkia rakenteen olioihin kohdistuvia ja aikataulutukseen vaikuttavia rajoitteita. Luokan ilmentymä sisältää aikataulurajoitteen ajallisen vaikutusalueen sekä 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. Tietokannan specialcase-taulussa kuvataan sovelluksessa käytettävien aikataulurajoitusten tiedot. Aikataulurajoitusten linkitys hoidetaan esityksiin specialty-taulun ja esittäjiin sekä puheenjohtajiin restriction-taulun kautta. Rajoitteet saavat säiliöluokaltaan merkkijonotiedon siitä, mille oliolle ne kuuluvat. Tätä merkkijonotietoa käytetään konfliktin viestinmuodostuksessa.

6.9 Conflict

Attribuutti	Tyyppi	Tehtävä
message	String	Konfliktin käyttäjälle tulostama viesti
initiators	Vector	Konfliktin aiheuttaneet rajoitteet sisältävä rakenne
dateformat	DateFormat	Konfliktin päivämäärämuodon määräävä DateFormat-instanssi

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 konfliktista. Viesti sisätltää konfliktin synnyttämän olion nimen tai otsikon, konfliktoivan aikajakson sekä aikajakson ja sen kanssa konfliktoivan rajoitteen ajallisen leikkauksen. 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.

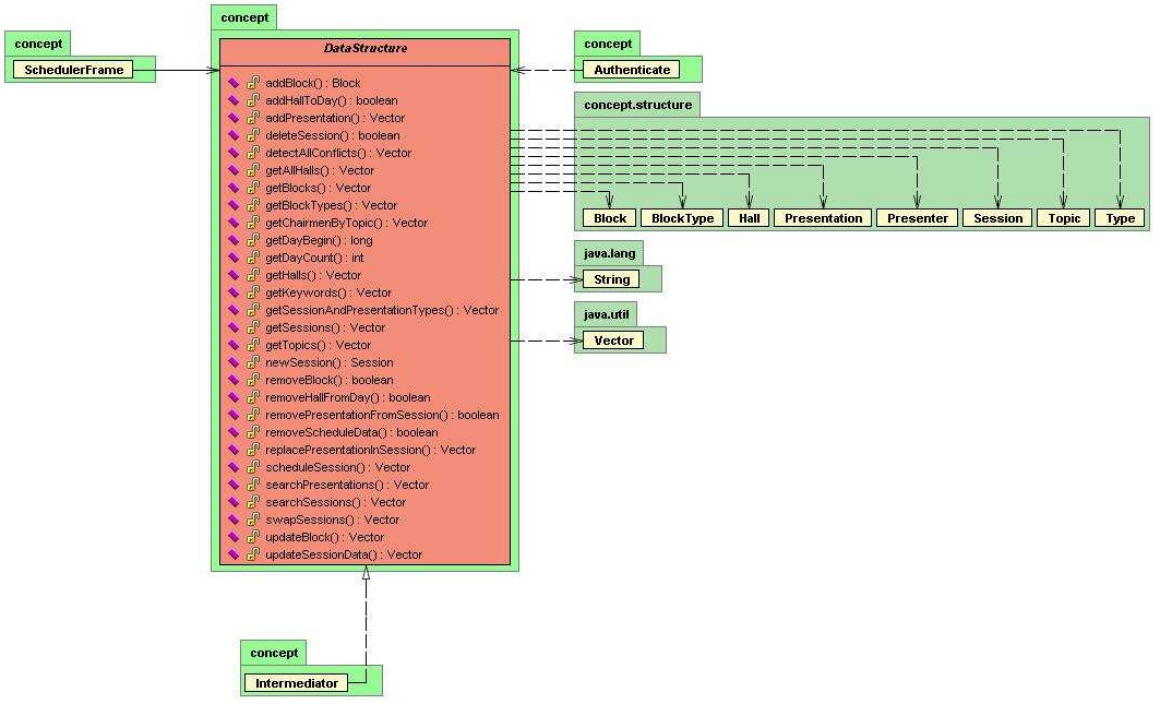
6.10 Tietokannan ja sovelluksen kommunikaatiota helpottavat luokat

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 erotessa syntyy konflikti. Topic-luokan esiintymät kuvaavat eri aihealueita. Sovelluksessa session aihealue vaikuttaa siihen osallistuviin esityksiin ja puheenjohtajiin. Session puheenjohtajia valittaessa, näytetään ainoastaan session aihealueen puheenjohtajat. Aihealue vaikuttaa myös suoritettaviin hakuihin, esimerkiksi puheenjohtajia voi hakea aihealueen mukaan. Topic- ja Type-luokat on suunniteltu samankaltaisiksi tietokannan sessiontype-taulun ja topic-taulun kanssa.

Tietorakenne sisältää myös Preference-luokan aikataulutussovelluksen asetuksia varten. Tämän luokan ja sen täydellisen toiminnan lopullinen toteutus jää kuitenkin jatkokehityksen varaan.

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. Toimintalogiikka huolehtii sovelluksen käytön aikana tietorakenteen linkityksen eheydestä. Kun sovellus käynnistetään ja sen tarvitsemat tiedot ladataan tietokannasta, likityksestä huolehtii tietorakenteen muodostajat ja lisäysmetodit. 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 aiheutta-

mat 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, jotka vastaavat käyttöliittymän toteuttamia tehtäväkokonaisuuksia. Rajapinnan toteuttava Intermediator-luokka kerää tietorakenteesta tarvittavat metodit jonkin tehtäväkokonaisuuden toteuttamiseen. Datastruct-rajapinnan määrittelemät ja ajonaikaista tietorakennetta koskevat metodit on esitetty taulukoissa 7.1, 7.2 ja 7.3.

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

Toimintalogiikan aikataulutustoimenpiteitä suorittavat metodit palauttavat käyttöliittymälle Vector-tietorakenteen, johon varastoidaan toimenpiteen aikatauluun mahdollisesti aiheuttamat konfliktit. Tämä käytäntö palauttaa käyttöliittymälle ainoastaan suoritetun operaation näkökulmasta aiheutuneet konfliktit. Järjestelmän kaikki konfliktit on etsittävä käyttäen detectAllConflicts-metodia.

Rajapinnan toteuttavat toimintalogiikan metodit, jotka ovat yhteydessä tietokantakomponentin metodeihin, käsittelevät tietokannan metodien mahdollisesti heittämiä poikkeuksia. Poikkeuksia ilmenee, mikäli sovelluksen yhteys tietokantaan katkeaa odottamattomasti kesken sovelluksen käytön. Toimintalogiikka käsittelee poikkeuksen suorittamalla jo mahdollisesti tietorakenteeseen kohdistuneita muutoksia tehneiden toimenpiteiden vastaoperaatiot. Näin varmistetaan, että tietokannan tiedot vastaavat ajonaikaisen tietorakenteen tietoja. Kaikki toimintalogiikan käsittelemät ja tietokantakomponentin aiheuttamat poikkeukset ovat SQLException-tyyppisiä. Toimintalogiikan metodeista tarkemman kuvauksen tarjoaa Javadoc-dokumentaatio liittessä C.

Käyttöliittymän ja ajonaikaisen tietorakenteen välisten metodien lisäksi toimintalogiikka sisältää kaksi käyttöliittymän ja tietokantakomponentin välistä hakumetodia. Nämä metodit suorittavat haun sekä sessio- että esityskohtaisena. Metodien tarkka kuvaus taulukossa 7.3.

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 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 getSessions();	Palauttaa päivälle aikataulutetut sessiot.
public Vector getChairmenByTopic(Topic topic);	Palauttaa aihepiirin käytettävissä olevat puheenjohtajat.
public Vector addPresentation(Session s, Presentation p);	Lisää esityksen p sessioon s.
public boolean removePresentation(Session s, Presentation p);	Poistaa esityksen p sessiosta s
public Vector scheduleSession(Session s, Hall h, Block b);	Sijoittaa session s blokkiin b ja saliin h
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, Presenter cman1, Presenter cman2);	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.

Taulukko 7.1: DataStruct-rajapinnan metodit.

Metodi	Kuvaus
public Vector detectAllConflicts();	Palauttaa kaikki kyseisellä hetkellä olemassaolevat konfliktit.
public Vector swapSessions(Session s1, Session s2);	Vaihtaa sessioiden s1 ja s2 paikkoja.
public boolean removeScheduleData(Session s);	Poistaa session s aikataulusta.
public Vector replacePresentationInSession(Presentation p, int newPlacement);	sijoittaa esityksen p sijaintiin newPlacement session esitysjärjestyksessä.
public Block addBlock(long begin, long end, BlockType type);	Luo uuden blokkityyppiä type olevan blokin alkuajalla begin, loppuajalla end.
public Vector updateBlock(Block block, long begin, long end, BlockType type);	Päivittää blokin tiedot annettujen parametrien mukaisiksi.
public boolean addHallToDay(Hall hall, long day);	Lisää salin hall päivään joka alkaa millisekunnilla day.
public boolean removeBlock(Block block);	Poistaa blokin block.
public boolean removeHallFromDay(Hall hall, long day);	Poistaa salin hall päivältä, joka alkaa millisekunnilla day.
public boolean deleteSession(Session s);	Poistaa session s.

Taulukko 7.2: DataStruct-rajapinnan metodit.

Metodi	Kuvaus
public Vector searchSessions(String abbrv, String Title, Topic topic, Type type, Presenter cman, boolean isReady, boolean notReady, boolean hasRestrictions, boolean noRestrictions, boolean isScheduled, boolean notSceduled);	Etsii sessioita annetuilla parametreillä.
public Vector searchPresentations(String title, Topic topic, Type type, String presenter, String keyword1, String keyword2, String keyword3, int grade, boolean isRegistered, boolean notRegistered, boolean isPlaced, boolean notPlaced, boolean hasRestrictions, boolean noRestrictions);	etsii esityksiä annetuilla parametreillä.

Taulukko 7.3: DataStruct-rajapinnan hakumetodit.

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. Käyttäjä syöttää tietokannan osoitteen ja se välitetään tietokantakomponentille. Käyttäjä syöttää myös käyttäjätunnuksensa sekä salasanansa, joilla tietokantakomponentti pyrkii tietokantaan. Tietokantakomponentti toimii siis autentikoivana osana järjestelmässä. Tietokantakomponentin tuntee tietorakenteen luokat, sillä komponentti joutuu rakentamaan lukumetodeillaan tietorakenteen sovellus käynnistettäessä. Lisäksi tietokantakomponenttia käytetään Authenticate-luokan sekä Intermediator-luokan kautta.

Tietokantakomponentti muodostuu Jdbc-luokasta, joka sisältää tietokantaa oliorakenteeksi kirjoittavan parserin sekä vastaavasti oliorakenteen tietoja tietokantaan kirjoittavan parserin. Kun sovellus käynnistetään, tietokantakomponentti saa Authenticate-luokalta käyttäjän syöttämät tunnukset, joilla se pyrkii ottamaan yhteyden käyttäjän osoittamaan tietokantaan. Mikäli yhteydenotto ei onnistu, palauttaa tietokantakomponentti Authenticate-luokalle poikkeuksen. Poikkeus käsitellään Authenticate-luokassa ja käyttäjältä kysytään uudestaan tietokannan osoite sekä yhteydenottotunnukset. Tietokantakomponentti hakee tietokannan tiedot eri tehtäväkokonaisuuksia toteuttavilla valmistelluilla SQL-kyselylausekkeilla, jotka on jaoteltu vastaavia toimintoja kuvaaviin metodeihin. Tietokantakomponentti saa tietokannasta tekemiään kyselyjä vastaavaa tietoa ja muodostaa näistä tiedoista ResultSet-olion. ResultSet-oliolta haetaan tietokannan kyselytuloksen tietoja, joiden pohjalta muodostetaan tietorakenteen luokkien määrittelemiä olioita ajonaikaiseen käyttöön. Eri toimintoja vastaavat metodit palauttavat luomansa oliot.

Tietokantakomponentin metodeihin on sisällytetty kannan lukitus operaatioiden väliseksi ajaksi. Tämä ylläpitää tietokannan ja sen käyttäjälleen tarjoaman tiedon eheyttä. Tietokannan lukitus on toteutettu PostgreSQL-tietokannan tarjoamilla resursseilla. Kannan lukitseminen estää sovelluksen samaan tietokantaan kohdistuvan yhtäaikaisen käytön useasta sovellusinstanssista. Näin voidaan taata, että sovelluksen tarjoama tieto ja sovelluksen aikaansaamat muutokset välittyvät kantaan virheettömästi ja tietokannan varastoiman tiedon eheys säilyy. Kannan lukitusta hallitaan tietokantakomponentin Jdbc-luokan lockDatabase()- ja unlockDatabase()-metodeilla.

Tietokantakomponentin lukuoperaatiot on yhdistetty yksittäisen metodin, readConference():n alle. Kyseinen metodi tekee muiden tietokantakomponentin metodien avulla tarvittavat SQL-kyselyt tietokantaan ja muodostaa ResultSet-olioiden vastauksista olioita sekä linkittää oliot tietokannan relaatioiden mukaisesti. readConference()-metodi kysyy tietokannalta kaikki sovelluksen ajonaikaisen tietorakenteen toiminnassa tarvittavat tiedot ja muodostaa tiedoista valmiin Conference-luokan ilmentymän, joka sisältää kokonaisen tietokannan tietoja vastaavan tietorakenteen linkityksineen sovelluksen käyttöön. Tämä ilmentymä palautetaan metodin palautusarvona ylempien luokkien käyttöön.

readConference()-metodi ja muut sen kutsumat lukumetodit käyttävät olioiden muodostamiseen ja niiden välisen viiterakenteen muodostamiseen tietorakenteen määrittelemiä luokkia, niiden konstruktoreita sekä lisäys-metodeja. readConference()-metodia lukuunottamatta lukumetodit ovat private-määreellä rajattu vain luokan sisäiseen käyttöön. Tietokantakomponentin kutsumat konstruktorit sekä lisäys-metodit linkittävät read-metodien luomat oliot. Kuvattujen lukuoperaatioiden ulkopuolella sovelluksen ajonaikainen linkittäminen jakautuu toimintalogiikan Intermediator-luokan metodeille sekä säiliöluokkien osalta tietorakenteen luokkien määrittämille konstruktoreille.

Tietokantakomponentin kaikki sovelluksen tarjoamaa informaatiota kantaan kirjoittavat metodit aiheuttavat kutsuvalle luokalle välitettävän poikkeuksen tietokannan käsittelyn tuottaessa virheen, esimerkiksi silloin, jos yhteys tietokantaan on poikki. Tämä poikkeus käsitellään toimintalogiikassa, joka samalla huolehtii, että tietokannan ja ajonaikaisen tietorakenteen tiedot ovat yhtenevät. Toimintalogiikka palauttaa käyttöliittymälle tiedon epäonnistuneesta toimenpiteestä, josta käyttöliittymä puolestaan informoi käyttäjää.

Kun suoritetaan haku sovelluksessa, se välitetään tietokannan hakuresursseille tietokantakomponentin kautta. Tämän jälkeen tietokantakomponentti muodostaa hakua vastaavan SQL-lauseen ja tietokanta palauttaa hakua vastaavan tiedon. SQL-kyselyn tuloksista muodostetusta ResultSet-oliosta tietokantakomponentti tunnistaa tietorakenteessa sijaitsevat vastaavat oliot ja palauttaa toimintalogiikan kautta viitteet käyttöliittymälle. Käyttöliittymä saa hakua vastaavien olioiden viitteet Vector-tietorakenteessa, jonka sisällön se voi kätevästi esittää käyttäjälle. Myös tietokantakomponentin metodeista tarkemmat kuvaukset liitteen C osoittamassa Javadoc-dokumentaatiossa.

9 Sovellukselle asetetut vaatimukset ja jatkokehitys

Sovellus on suunniteltu siten, että sen osia voidaan tarvittaessa helposti laajentaa ja kehittää. Projektin aikana ilmenikin useita jatkokehitystarpeita ja ideoita. Sovelluksen kehityksen loppuvaiheessa ilmeni testauksessa sovelluksen joissakin toiminnoissa puutteita, jotka päätettiin siirtää jatkokehitykseen. Jatkokehityskohteiksi lisättiin sovelluksen tarjoamien hakutulosten järjestäminen haluttuun järjestykseen sekä puheenjohtajien puheenjohtajuusmäärien ilmoittaminen. Lisäksi sovelluksen hylättyjen ja hyväksyttyjen esitysten erottelu siirrettiin jatkokehitykseen. Projektin tuottaman vaatimusmäärittelyn [1] vaatimuksista osa siirrettiin jatkokehitykseen projektin ajallisten resurssien käytyä vähiin. Sovellukseen voi vaatimusmäärittelyn vaatimuksesta 5.5.1 poiketen lisätä kaksi sessiota samalla nimellä ja lyhenteellä. Lisäksi esitystä sessioon lisättäessä ei tarkisteta aihealueiden yhteensopivuutta vaatimusmäärittelyn vaatimuksesta 5.6.1 poiketen. Saman vaatimuksen ilmoittama esittäjien rajoitteiden havainnollinen esittäminen siirrettiin myös jatkokehitykseen. Sovellukselle asetetuista vaatimuksista jatkokehitykseen siirrettiin myös session keston tarkistaminen blokin suhteen, vaatimusmäärittelyssä vaatimus 5.5.5. Sovelluksen kehityksen loppuvaiheessa ilmenneistä tietokannan ja sovelluksen rakenteen yhteensopivuusongelmista johtuen jatkokehitykseen siirrettiin myös vaatimus 5.8.2 Esittäjien rajoitteet, vaatimus 5.8.3 Puheenjohtajien rajoitteet sekä vaatimus 5.8.1 Henkilökohtaiset rajoitteet. Konfliktijärjestelmään suoraan vaikuttavista yhteensopivuusongelmista enemmän myöhemmin. Lisäksi sovelluksen vaatimusmäärittelyn matalan prioriteetin vaatimukset siirrettiin sovitusti jatkokehityskohteiksi.

Sovelluskehityksen suurin ongelma paljastui projektin loppuvaiheessa, kun projektiryhmä havaitsi, ettei alkuperäisen konferenssinhallintajärjestelmän toteuttama tietokantarunko toiminutkaan projektiryhmän kuvittelemalla tavalla. Tietokannan users-taulussa oli ryhmän käsitysten vastaisesti sekä yksilöityjä käyttäjiä, että täysin yksilöimättömiä esittäjiä. Yhdellä esittäjällä havaittiin olevan ainakin yhtä monta instanssia kuin esityksiäkin. Näitä instansseja yhdisti ainoastaan sama etu- ja sukunimi sekä sähköpostiosoite. Instanssien välillä esittäjien tiedot kuitenkin vaihtelivat muun muassa kirjoitusvirheiden takia. Projektiryhmä oli pitänyt sovelluksen sunnittelussa keskeisenä tekijänä oliosuunnittelun periaatetta, jonka mukaan yksilöitävistä järjestelmäelementeistä muodostetaan omat yksilölliset instanssinsa, joita kutsutaan olioiksi. Koska tietokanta tarjosi nyt tietoa esittäjistä, joita ei voitu luotettavasti yksilöidä, kohtasi sovelluksen oliorakenteessa toteutettu konfliktinhallintajärjestelmä haasteen johon ei löydetty selkeää ratkaisua. Ongelma aiheuttaa sen, ettei

Vaatimus	Osittain toteutettu
Salitietojen tallentaminen	
Salien muokkaaminen	
Session asettaminen ajallisesti sessiota lyhyempään blokkiin	x
Varoajat	
Puhujan varoajan asettaminen	
Rajoitteiden konfliktoinnin estäminen	
Sovelluksen värien valinta	
Konflikteista varoittaminen	
Aikatauludumppi	
Laajennettu haku	
Salin lisääminen manuaalisesti	
Esittäjien rajoitteet	
Puheenjohtajien rajoitteet	
Henkilökohtaiset rajoitteet	
Esityksen lisääminen sessioon	x
Session lisääminen	x
Session sali- ja aikatietojen määrittäminen	x

Taulukko 9.1: Vaatimuksista muodostetut jatkokehityskohteet

sovellukseen toteutettu konfliktin havainnointi toimi tietokannan toisteisen datan kanssa. Jos esittäjillä on oma ilmentymä kutakin esitystään kohden, ei kahden saman esittäjän sisältävän esityksen ajallisesti rinnakkaisesta aikataulutuksesta synny konfliktia. Projektiryhmä kehitteli useita ideoita ongelman ratkaisemiseksi, mutta projektin loppuvaiheen olemattomien ajallisten resurssien sekä ongelman vaikeuden takia, ratkaisu sovelluksen konfliktinhallinnan lopullisesta toteutuksesta jätettiin jatkokehitykseen.

Projektiryhmän mielestä ongelma juonsi juurensa aiemman konferenssihallintajärjestelmää kehittäneen sovellusprojektin kohtaamasta vaikeasta ongelmasta koskien esittäjien yksilöimistä konferenssin ilmoittautumistasolla. Tästä syystä toimivaksi testattua oliorakennetta ei haluttu lähteä taivuttamaan epäluotettavasti toimivan merkkijonovertailun kautta toimivaksi järjestelmäksi. Projektiryhmän kesken mie-

tittiin useita eri ratkaisuvaihtoehtoja ja lopulta päädyttiin mielipiteeseen, jonka mukaan ongelma olisi parasta lopullisesti ratkaista järjestämällä esittäjät yksilöivää dataa sovelluksen käyttöön. Väliaikaiseksi ratkaisuksi projektiryhmä ideoi järjestelmää, jossa sovelluksen toteuttamiseksi muodostettuun oliorakenteeseen ei tehtäisi muutoksia, vaan oliorakenteeseen nojaava konfliktinhallinta ohitettaisiin erilliseksi paketiksi kootulla tietokantakomponentin kautta toteutetulla merkkijonovertailuun pohjaavalla konfliktinhallinnalla. Toimintalogiikan konfliktinetsintäkutsun kautta siirryttäisiin siis tietokantakomponentin puolelle, jossa toteutettaisiin SQL-kysely, joka paljastaisi saman nimen ja mahdollisesti sähköpostiosoitteen omaavien esittäjien kaikki instanssit ja niiden rajoitukset. Näitä rajoituksia tutkittaisiin aikataulutietojen valossa ja syntyneet konfliktit raportoitaisiin alkuperäisen konfliktinetsinnän käynnistäneelle käyttöliittymän metodille. Tällaisella muutoksella kehitetty aikataulutussovellus säilyttäisi valmiutensa toteuttaa tehokasta konfliktinhallintaa oliotasolla, mikäli ilmoittautumistason yksilöitävyysongelmat joskus saataisiin ratkaistua. Muutoksen ansiosta sovelluksella voitaisiin tehdä aikataulutustyötä rajoitetulla luotettavuudella toimivan konfliktinhallinnan avustuksella tuhoamatta alkuperäistä suunnitteluinnovaatiota.

10 Yhteenveto

Sovelluksen suunnittelu on toteutettu useassa iteratiivisessa vaiheessa. Sovellus koostuu käytännössä neljästä komponentista, joita on pyritty erottamaan toisistaan tarpeen vaatiessa. Käyttöliittymäkomponentin erottuminen toiminnallisesta rakenteesta oli eräs suunnittelun ja myöhemmin toteutuksen lähtökohdista. Tämä on sovelluksen selkein rajapinta ja mikä tahansa käyttöliittymäkomponentti, joka toteuttaa kyseisen rajapinnan, voi toimia sovelluksen käyttöliittymänä. Lisäksi tietokantaoperaatioiden kokoamista saman komponentin alle pidettiin tärkeänä. Sovelluksen kehityksessä 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 ja olioiden viitteet relaatiotauluja. Tämä vastaavuus helpottaa oleellisesti tietokannan tietojen ja ajonaikaisen tietorakenteen välisiä muunnoksia. Samankaltaista vastaavuutta on noudatettu myös tietokantaan tehdyissä muutoksissa. Niiden tarve on lähtenyt sovelluksen vaatimista lisätiedoista ja näitä tietoja tallentavat tietokantaan lisätyt taulut ja relaatiot on suunniteltu sovelluksen toiminnan vaatimien olio-ja viiterakenteiden mukaan.

Sovellussuunnitelmassa [2] kuvattiin sovelluksen suunniteltua toteutusta erään suunnitteluiteraation jälkeen. Sovellussuunnitelmassa esitettyyn sovellusrakenteeseen tulikin joitakin muutoksia. Etenkin tietokantakomponentin toteutus kävi sovellussuunnitelmassa esitetyn mallin jälkeen läpi lukuisia mittaviakin muutoksia. Suurin muutos oli suunnitellun jaksottaisen erissä suoritettavan tallennuksen vaihtuminen jokaisen operaation jälkeen suoritettavaksi tallennukseksi. Tämä muutos yksinkertaisti tietokannan ja sovelluksen oliorakenteen kommunikaatiota ja käyttöä huomattavasti, vaikka se toikin sovelluksen toteuttamille operaatioille lisätyötä. Tehty muutos kehittää sovellusta myös käyttäjäystävällisempään suuntaan, sillä pahimmassa tapauksessa sovelluksen suorituksen keskeydyttyä odottamattomasti, tietokannasta menetetään korkeintaan viimeksi tehdyn operaation aiheuttamat muutokset. Näin sovelluksen odottamaton keskeytyminen tuhoaa minimaalisesti sovelluksella tehtyä aikataulutustyötä. Myös sovelluksen tietorakenteen koostumusta muutettiin ja parannettiin yhteistoimintaa sekä käyttöliittymän että tietokantakomponentin kanssa. Muutokset kohdistuivat säiliöluokkien konstruktoreihin ja lisäys- sekä poisto-metodeihin. Muutos sovellussuunnitelmaan nähden tehtiin myös sovelluksen viitehallinnan kohdalla. Sovelluksen viitteet muodostetaan ajonaikaisesti

nyt Intermediator-luokan metodeilla suunnitellusta tietorakenteen lisäys- ja poistometodien hoitamasta viitehallinnasta poiketen. Tietokantakomponentin luomien olioiden viitteytys hoidetaan kuitenkin tietorakenteen sisällä. Tietorakenteessa on erikseen tietokantakomponentin käyttöön tarkoitetut viitteyttävät lisää- ja poista-metodit, jotka on nimetty FromDB-loppuisilla nimillä.

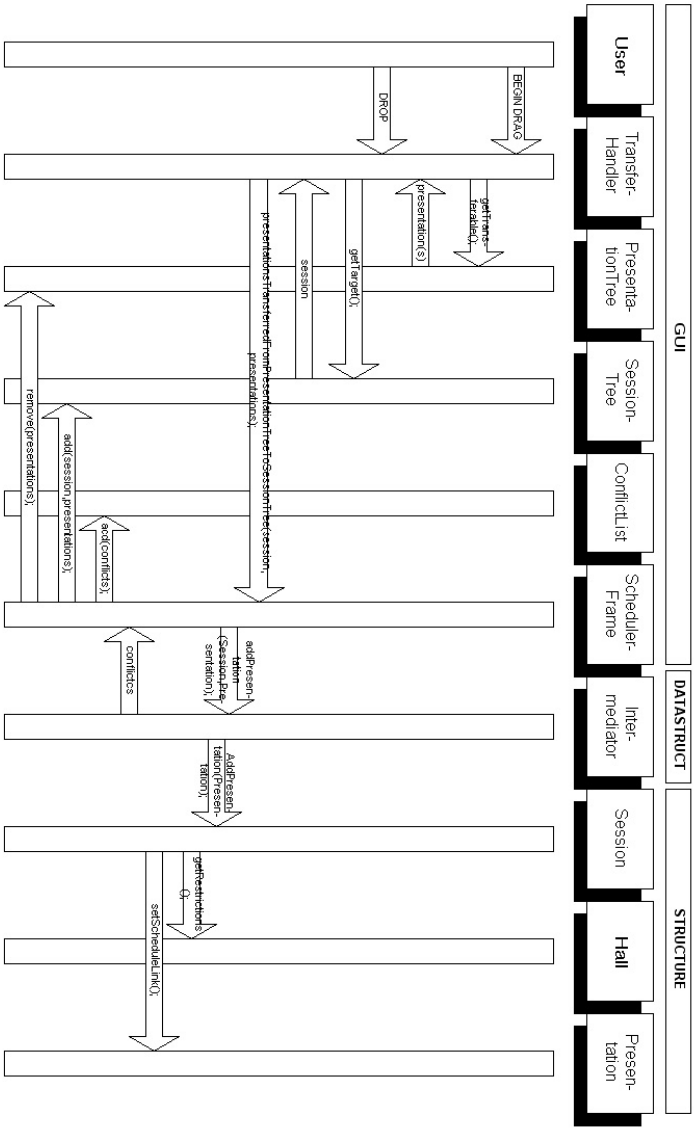
Sovelluksen kehittäminen osoittautui jopa ryhmän pahimpia pelkoja työläämmäksi. Sovelluksen kehityksen alkuvaiheessa pidettiin liiaksi kiinni lukuisista kunnianhimoisista vaatimuksista, joista useat sittemmin osoittautuivat toissijaisiksi tai liian työläiksi toteuttaa sovellusprojektin aikatauluun ja resursseihin nähden. Sovelluksen toteuttamisen priorisointia vaikeutti myös tuntuvasti vaatimusmäärittelyn venyminen miltei koko sovelluksen julkaisuversion kehityskaaren mittaiseksi. Tämä pakotti nopeaan iteraatioon sovelluksen suunnitteluvaiheessa, ja aiheutti puolestaan toteutusvaiheen viivästymisen ja nopean kehitystahdin. Ongelmista huolimatta projektiryhmä uskoo toteutetun sovelluksen toimivan jatkokehityksen mahdollistavana runkona tulevaisuuden konferenssinhallintajärjestelmän kehityksessä.

11 Lähteet

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

[2] PK TR PS AS, ”CONCEPT-projektin sovellussuunnitelma”,2005

A Sekvenssikaavio



Kuva A.1: Esityksen lisääminen aikataulutettuun sessioon.

B Tietokantaan lisätyt taulut

Sarake	Tyyppi	Pakollinen	Rajoitteet
hallid	serial	kyllä	PRIMARY_KEY
title	text		
capacity	integer		

Taulukko B.1: hall-taulu

Sarake	Tyyppi	Pakollinen	Rajoitteet
restrictionid	serial	kyllä	PRIMARY_KEY
hall	integer	kyllä	
”start”	timestamp		
finish	timestamp		

Taulukko B.2: hallrestriction-taulu

Sarake	Tyyppi	Pakollinen	Rajoitteet
userid	serial	kyllä	PRIMARY_KEY
username	text		
tolerance	integer	kyllä	
conflict_r	integer	kyllä	
conflict_g	integer	kyllä	
conflict_b	integer	kyllä	

Taulukko B.3: preference-taulu

Sarake	Tyyppi	Pakollinen	Rajoitteet
”user”	integer	kyllä	PRIMARY_KEY, preference(userid) UPDATE CASCADE, DELETE CASCADE
topic	integer	kyllä	PRIMARY_KEY, topic(topicid) UPDATE CASCADE, DELETE RESTRICT
r	integer	kyllä	
g	integer	kyllä	
b	integer	kyllä	

Taulukko B.4: colors-taulu

Sarake	Tyyppi	Pakollinen	Rajoitteet
scheduletypeid	serial	kyllä	PRIMARY_KEY
title	text	kyllä	

Taulukko B.5: scheduletype-taulu

Sarake	Tyyppi	Pakollinen	Rajoitteet
hall	integer	kyllä	PRIMARY_KEY, hall(hallid) UPDATE CASCADE, DELETE CASCADE
day	integer	kyllä	PRIMARY_KEY

Taulukko B.6: halldays-taulu

C Javadoc dokumentaatio

Sovelluksen javadoc-dokumentaatio on saatavilla tämän dokumentin paperiversion liitteenä sekä osoitteessa:
`http://sovellusprojektit.it.jyu.fi/concept/concept/index2.html`