

## **Tahiti — tähtitieteellisten havaintojen tietokanta**

Tomi Hänninen  
Juho Muhonen  
Ismo Puustinen  
Kai Pääsky  
Pekka Simola  
Nuutti Varis

Helsinki 31.3.2003

Suunnitteludokumentti

HELSINGIN YLIOPISTO

Tietojenkäsittelytieteen laitos

Tiedekunta/Osasto — Fakultet/Sektion — Faculty		Laitos — Institution — Department	
Matemaattis-luonnontieteellinen		Tietojenkäsittelytieteen laitos	
Tekijä — Författare — Author			
Tomi Hänninen		Juho Muhonen	Ismo Puustinen Kai Pääsky Pekka Simola Nuutti Varis
Työn nimi — Arbetets titel — Title			
Tahiti — tähtitieteellisten havaintojen tietokanta			
Oppiaine — Läroämne — Subject			
Tietojenkäsittelytiede			
Työn laji — Arbetets art — Level		Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Suunnitteludokumentti		31.3.2003	84 sivua
Tiivistelmä — Referat — Abstract			
<p>Tämä dokumentti on Tahiti-projektin suunnitteludokumentti. Dokumentin tarkoituksena on selvittää järjestelmän rakenne, jotta järjestelmän suoraviivainen toteutus ja Tahiti-ryhmän sisäinen tehtävänjako olisivat mahdollisia. Dokumentin toissijainen tehtävä on tehdä tulevaisuudessa valmiin järjestelmän yleisrakenteen ymmärtämisestä helpompaa järjestelmän ylläpidettävyyden vuoksi. Myös erilaiset ohjelmointisuorituksen yhtenäisyyteen liittyvät sopimukset, kuten koodikonventio, tullaan määrittelemään suunnitteludokumentissa.</p> <p>Uusin versio tästä dokumentista tulee aikanaan saatavilla Tahiti-ryhmän kotisivuille osoitteeseen <a href="http://www.cs.helsinki.fi/group/tahiti/">http://www.cs.helsinki.fi/group/tahiti/</a> .</p> <p>Versiohistoria:</p> <ol style="list-style-type: none"> <li>1. Versio 1.0 (katselmoitava versio)</li> <li>2. Versio 1.1 (korjattu versio)</li> <li>3. Versio 1.2 (muokattu versio)</li> </ol>			
Avainsanat — Nyckelord — Keywords			
Tahiti, fotometria, Standard Asteroid Photometric Catalogue			
Säilytyspaikka — Förvaringsställe — Where deposited			
Muita tietoja — övriga uppgifter — Additional information			
Versio 1.2			

# Sisältö

<b>1 Johdanto</b>	<b>1</b>
1.1 Toteutuksessa käytettäviä standardeja ja kirjallisuutta . . . . .	1
1.2 Tuote . . . . .	1
1.3 Sanasto . . . . .	1
1.4 Yleiskatsaus dokumenttiin . . . . .	2
<b>2 Yleiskuvaus</b>	<b>2</b>
<b>3 Container-rajapinta</b>	<b>4</b>
3.1 Container-rajapinnat . . . . .	4
3.2 AsteroidQueryContainer . . . . .	5
3.3 UserQueryContainer . . . . .	7
3.4 LogQueryContainer . . . . .	7
3.5 LightcurveFormContainer . . . . .	7
3.6 UserFormContainer . . . . .	8
3.7 TrajectoryFormContainer . . . . .	8
3.8 SettingsFormContainer . . . . .	9
3.9 Lightcurve . . . . .	9
3.10 DataPoint . . . . .	10
3.11 Trajectory . . . . .	11
3.12 Asteroid . . . . .	11
3.13 User . . . . .	12
3.14 Settings . . . . .	12
3.15 LogEntry . . . . .	13
3.16 TahitiLibrary . . . . .	13
<b>4 Tahiti-API</b>	<b>18</b>
4.1 Yleistä Apista . . . . .	20
4.2 Kutsurakenteesta . . . . .	20
4.3 Käsittelijäluokkien yhteisiä piirteitä . . . . .	25
4.4 Lokikirjoituksesta . . . . .	26
<b>5 Tahiti-Apin luokat</b>	<b>27</b>
5.1 Luokan DBControl kuvaus . . . . .	36

5.2	Luokan LightcurveHandler kuvaus . . . . .	37
5.3	Luokan LogHandler kuvaus . . . . .	40
5.4	Luokan MailHandler kuvaus . . . . .	41
5.5	Luokan OutputCreator kuvaus . . . . .	42
5.6	Luokan SystemHandler kuvaus . . . . .	43
5.7	Luokan TrajectoryHandler kuvaus . . . . .	43
5.8	Luokan UserHandler kuvaus . . . . .	44
<b>6</b>	<b>Tietokanta</b>	<b>48</b>
6.1	Numerosarjat . . . . .	48
6.2	Käyttäjät . . . . .	49
6.3	Taulut . . . . .	50
6.3.1	TahitiUser . . . . .	50
6.3.2	Asteroid . . . . .	51
6.3.3	Observation . . . . .	51
6.3.4	Lightcurve . . . . .	52
6.3.5	DataPoint . . . . .	53
6.3.6	Trajectory . . . . .	53
6.3.7	LogEntry . . . . .	54
6.3.8	Settings . . . . .	54
6.4	Näkymät . . . . .	55
6.4.1	AsteroidOverview . . . . .	55
6.4.2	LightcurveOverview . . . . .	55
6.4.3	LightcurveUpdate . . . . .	56
6.5	Funktiot . . . . .	56
6.5.1	default_trajectory . . . . .	57
6.5.2	observation_time . . . . .	57
6.5.3	number_of_datapoints . . . . .	57
<b>7</b>	<b>Käyttöliittymäluokat</b>	<b>57</b>
7.1	Kontrolliluokat . . . . .	58
7.1.1	AsteroidQueryController . . . . .	59
7.1.2	AsteroidDataController . . . . .	59
7.1.3	LightcurveController . . . . .	59
7.1.4	CustomElementController . . . . .	59

7.1.5	LightcurveSubmitController . . . . .	60
7.1.6	UserInfoUpdateController . . . . .	60
7.1.7	UserSearchController . . . . .	60
7.1.8	UserInfoGetController . . . . .	60
7.1.9	LightcurveModifyController . . . . .	60
7.1.10	AsteroidDataSubmitController . . . . .	60
7.1.11	MailController . . . . .	61
7.1.12	SiteConfigurationSubmitController . . . . .	61
7.1.13	LoginController . . . . .	61
7.1.14	EventHistoryDataController . . . . .	61
7.1.15	UserRegisterController . . . . .	61
7.2	Bean-tietoluokat . . . . .	62
7.2.1	AsteroidQueryBean . . . . .	62
7.2.2	LightcurveSubmitBean . . . . .	62
7.2.3	LightcurveDataBean . . . . .	62
7.2.4	AsteroidTrajectoryDataBean . . . . .	62
7.2.5	EarthTrajectoryDataBean . . . . .	62
7.2.6	UserInfoDataBean . . . . .	62
7.2.7	UserSearchBean . . . . .	62
7.2.8	UserSearchDataBean . . . . .	63
7.2.9	LoginBean . . . . .	63
7.2.10	LoginDataBean . . . . .	63
7.2.11	EventHistoryDataBean . . . . .	63
7.2.12	SiteConfigurationDataBean . . . . .	63
<b>8</b>	<b>Jsp-käyttöliittymäsivut</b>	<b>63</b>
8.1	header.jsp – Järjestelmän otsikkosivu . . . . .	65
8.2	index.jsp – aloitussivu . . . . .	65
8.3	register.jsp – rekisteröitymissivu . . . . .	68
8.4	browseasteroid.jsp – Asteroidin tiedot-sivu . . . . .	70
8.5	cesubmit.jsp – Omien ratatietojen syöttö . . . . .	74
8.6	datafile.jsp – Datatiedoston näyttö . . . . .	74
8.7	lcsubmit.jsp – Valokäyrän syöttö . . . . .	74
8.8	asteroidsubmit.jsp – Asteroidin ratatietojen syöttö . . . . .	76

	v
8.9 userinfo.jsp – Käyttäjän omien tietojen päivitys . . . . .	76
8.10 usersearch.jsp – Käyttäjien etsintä . . . . .	79
8.11 edituserinfo.jsp – Käyttäjän tietojen päivitys . . . . .	79
8.12 siteadmin.jsp – Järjestelmän asetusten päivitys . . . . .	81
8.13 eventhistory.jsp – Järjestelmän tapahtumahistoria . . . . .	82
<b>9 Atlas-syötin</b>	<b>83</b>
9.1 Toiminta . . . . .	83
9.2 Toteutus . . . . .	83
<b>Lähteet</b>	<b>84</b>

# 1 Johdanto

Tämä dokumentti on Tahiti-ohjelmistotuotantoprojektiryhmän suunnitteludokumentti. Dokumentin tarkoituksena on selvittää järjestelmän rakenne niin arkkitehtuurillisesti kuin luokkatasollakin. Järjestelmä tullaan luomaan tämän dokumentin pohjalta, joten sisällön luonne on hyvin tekninen. Myös järjestelmän toteutuksen jälkeinen ylläpito helpottuu, kun järjestelmän suunnittelu on tehty hyvissä ajoin ja dokumentoitu huolella.

Dokumentin kohderyhmänä on ensisijaisesti Tahiti-ohjelmistotuotantoprojektiryhmä, mutta myös asiakkaan tulee tarpeen tullen saada tietoonsa järjestelmän arkkitehtuuri sekä komponentit. Järjestelmän projektin jälkeinen ylläpito, päivitys ja korjaukset helpottuvat myös, kun tämän dokumentin tiedot ovat saatavilla.

## 1.1 Toteutuksessa käytettäviä standardeja ja kirjallisuutta

Järjestelmän ohjelmoinnissa noudatetaan Java Code Conventions [Sun03] -ohjelmointityyliä. Ohjelmakomponenttien sisäinen kommentointi toteutetaan Javadoc-menetelmää hyväksikäyttäen. Tähtitieteellinen sanasto on lainattu kirjasta Tähtitieteen perusteet [Kar00]. Osa sanastosta on selitetty Tahiti-ohjelmistotuotantoprojektiryhmän määrittelydokumentissa [Tah03].

## 1.2 Tuote

Tuotettava järjestelmä on fotometrinen havaintojen tietokanta. Fotometrisissa havainnoissa tallennetaan kohteesta — tässä tapauksessa asteroidista — heijastuvan valon intensiteetti eri ajan hetkinä. Järjestelmän pääasialliset toiminnot ovat havaintojen haku ja tallettaminen. Järjestelmän on tilannut Helsingin yliopiston tähtitieteen laitos, jonka edustajana projektin aikana toimii dosentti Mikko Kaasalainen.

## 1.3 Sanasto

- *Api* on lyhenne sanoista Application Programming Interface eli suomeksi ohjelmointirajapinta. Järjestelmän yhteydessä Api tarkoittaa rajapintaluokkaa, jonka kautta päästään käsiksi varsinaisiin järjestelmäpalveluihin kuten tiedonkäsittelyyn.
- *Intensiteetti* tarkoittaa projektin puitteissa säteilyn l. näkyvän valon vuontiheyttä. Tähtitieteellisessä kirjallisuudessa käsite voi saada muitakin merkityksiä.
- *Java Beans* -luokat ovat Javan luokkia, joita käytetään pääasiassa tiedon välittämiseen komponenttien välillä. Järjestelmän Beanit sisältävät myös syötteiden tarkistamislogiikkaa.
- *JSP* on lyhenne sanoista Java Server Pages. JSP on tekniikka, jolla saadaan yhdistettyä HTML-koodia ja Java-ohjelmakoodia. JSP:llä voidaan toteuttaa www-sovelluksia hieman tavallisia Java-luokkia joustavammin.
- *JSP-moottori* on ohjelmisto, joka suorittaa ja tulkitsee JSP-sivuja (kts. edellinen kohta). Järjestelmän JSP-moottorina toimii Apache Tomcat. Sama moottori toimii myös Java Servlet-

moottorina. Ohjelmisto lyhyesti sanottuna suorittaa järjestelmän komponentteja tarvittavassa laajuudessa.

- *Magnitudi* on mittayksikkö, joka kehitettiin ennen ajanlaskun alkua tähtien kirkkauksien vertailemiseen. Alunperin tähdet jaettiin kirkkauden mukaan kategorioihin yhdestä kuuteen, suuremman luvun tarkoittaessa himmeämpää tähteä. Kun kehitettiin kaukoputkia ja ihmisilmää tehokkaampia havaintoinstrumenttejä, kirkkaimmat kohteet jouduttiin luokitsemaan negatiivisilla arvoilla. Aiempien havaintojen vertailukelpoisuuden vuoksi luokittelu pidettiin samana. Magnitudiasteikko on ihmissilmän erottelukyvyn logaritmisuuden vuoksi logaritminen. Tämän lisäksi edellä lueteltu asteikko on näennäinen magnitudiasteikko, sillä tähtien etäisyyksiä saatika havaintopaikan ja kohteen välillä olevia sumuja ja vastaavia valoa himmentäviä tekijöitä ei luokittelussa huomioida. Kun kyseiset seikat huomioidaan, saadaan kohteen absoluuttinen magnitudi laskettua yksinkertaisella kaavalla.
- *MVC* tulee sanoista Model View Controller. Kyseessä on suunnittelumalli, joka koostuu kolmesta eri kokonaisuudesta. Model eli malli esittää varsinaista dataa järjestelmässä. View eli näkymät ovat tiedon eri visualisointeja järjestelmässä. Controllerin eli käsittelijöiden avulla tieto voidaan muokata sitä kulloinkin tarvitsevan kokonaisuuden, mallin tai näkymien, tarvitsemaan muotoon.
- *Pakkaus* (package) on Javan tapa jaotella ohjelmien nimiavaruus pienempiin osiin. Yksi ohjelma voi koostua monesta pakkauksesta, joista jokainen muodostaa oman nimiavaruutensa.
- *Servlet* on Java-sovellus, joka ottaa palvelupyynnön joko www-lomakkeilta tai http-protokollan mukaisesti (eli www-osoitteen muodossa)

## 1.4 Yleiskatsaus dokumenttiin

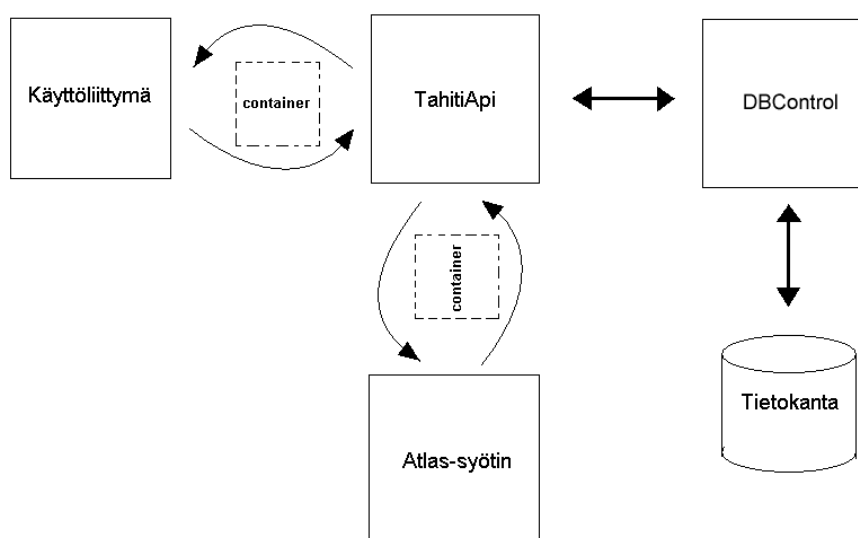
Tässä luvussa esiteltiin lyhyesti tämä dokumentti, toteutettava järjestelmä sekä keskeisimmät määritelmät. Luvussa kaksi luodaan yleiskatsaus järjestelmään. Kolmannessa luvussa perehdytään järjestelmän tietokomponentteihin eli containereihin. Neljännessä luvussa esitellään järjestelmän api eli ohjelmistorajapinta, jonka palveluilla varsinainen operointi tietokannan kanssa toimii. Viidennessä luvussa täsmennetään edellämäinitun apin käyttämät luokat. Kuudennessa luvussa esitellään käytettävä tietokantarakenne tauluineen, näkymineen ja fuktioineen. Seitsemännessä luvussa luodaan silmäys käyttöliittymän rakenteeseen. Kahdeksas luku esittelee käyttöliittymän www-sivut. Yhdeksännessä luvussa esitellään järjestelmän Atlas-syötin.

## 2 Yleiskuvaus

Järjestelmässä on kuvan 1 mukaisesti neljä suurta rakenne-elementtiä: käyttöliittymä, Tahiti-API, tietokanta ja Atlas-syötin. Näistä elementeistä Atlas-syötin ei varsinaisesti kuulu järjestelmään, vaan on ulkoinen osa. Järjestelmä jaetaan pakkauksiin (package) näiden osien mukaan, paitsi Container-luokat:

1. Käyttöliittymä pakkaukseen `fi.helsinki.cs.group.tahiti.ui`





Kuva 1: Järjestelmän yleiskuvaus

2. Tahiti-API pakkaukseen `fi.helsinki.cs.group.tahiti.api`
3. Containerit pakkaukseen `fi.helsinki.cs.group.tahiti.containers`
4. Atlas-syötin pakkaukseen `fi.helsinki.cs.group.tahiti.atlas-importer`

Koska tietokantaa ei toteuteta Java-kielellä, se ei myöskään tarvitse pakkausta.

Käyttöliittymä jakautuu seuraaviin osiin:

- JSP-sivuihin, jotka generoivat käyttäjän selaamien www-sivujen html-koodin
- Container-rajapinnan toteuttaviin JavaBean-luokkiin, jotka kuljettavat tietoa JSP-sivujen ja kontrolliluokkien välillä sekä
- kontrolliluokkiin, jotka tulkitsevat ja tarkastavat käyttäjän palvelupyynnön ennen sen toimittamista Tahiti-Apille.

Kun käyttäjä tekee järjestelmän www-sivulla tietojenkäsittelyllisen toimenpiteen, JSP-sivu luo toimenpiteestä ja sen syötteestä JavaBean-olion, joka toimitetaan kontrolliluokalle. Kontrolliluokka tutkii syötteen oikeellisuuden ja palauttaa tarvittaessa JSP-sivulle ilmoituksen datan virheellisyydestä. Syötteen ollessa kelvoinen kontrolliluokka toimittaa containerin edelleen Tahiti-Apille.

Tahiti-API on käsittelijöitä yhdistävä yhteinen rajapinta, joka huolehtii kommunikaatiosta käyttöliittymän ja käsittelijöiden välillä. Käsittelijät tarjoavat palveluita, joilla tietokantaan viedään dataa ja tietokannasta haetaan dataa Tahiti-Apin pyytämällä tavalla. Virheidenkäsittely suoritetaan palvelupyynnön tyyppin mukaisella tasolla: esimerkiksi virheelliset parametrit voidaan saada kiinni kontrolliluokissa, käsittelijöissä tai vasta tietokannassa.

Tietokanta on siinä mielessä autonominen osa, että käsittelijät joutuvat käyttämään sen omaa SQL-kieltä tietokantatoimintoja varten. Tietokanta koostuu kuvan 12 mukaisista tauluista, joissa järjestelmän sisältämää tietoa säilytetään.

Atlas-syötin on järjestelmän ulkopuolinen osa: sitä ei tarvita järjestelmän normaalin toiminnan aikana. Syötin suorittaa tehtävänsä silloin, kun vielä Atlas-tiedostomuodossa olevia valokäyriä siirretään ensi kertaa järjestelmään. Syötin koostuu yksinkertaisesta jäsentäjästä, joka jäsentää Atlas-tiedoston ja vie Tahiti-Apiä käyttäen Atlas-muotoisen valokäyrätiedon tietokantaan.

### 3 Container-rajapinta

Järjestelmä käyttää tiedon siirtämiseen käyttöliittymän ja TahitiApin välillä tätä tarkoitusta varten suunniteltuja rajapintaluokkia. Tässä dokumentissa näitä rajapintoja kutsutaan *container*-rajapinnoiksi. Container-rajapinnat sisältävät ainoastaan get- ja set-metodeita niiden sisälle talletettavien attribuuttien käsittelyä varten. Kussakin container-rajapinnassa on määritelty get- ja set-metodi kutakin talletettavaa attribuuttia kohti. Set-metodilla attribuutille asetetaan arvo ja get-metodilla attribuutin arvo haetaan. Siis esimerkiksi attribuutin AsteroidName omaavassa rajapinnassa ovat metodit `getAsteroidName()` ja `setAsteroidName()`.

Mitä tahansa luokkia, jotka toteuttavat container-rajapinnan, voidaan käyttää tiedon siirtämiseen järjestelmän ja järjestelmää käyttävän sidosryhmän välillä. Container-rajapintojen lisäksi järjestelmästä näkyy ulospäin luokka TahitiLibrary, joka sisältää järjestelmän käyttämät vakiot sekä järjestelmän käyttämiä laskentamenetelmiä.

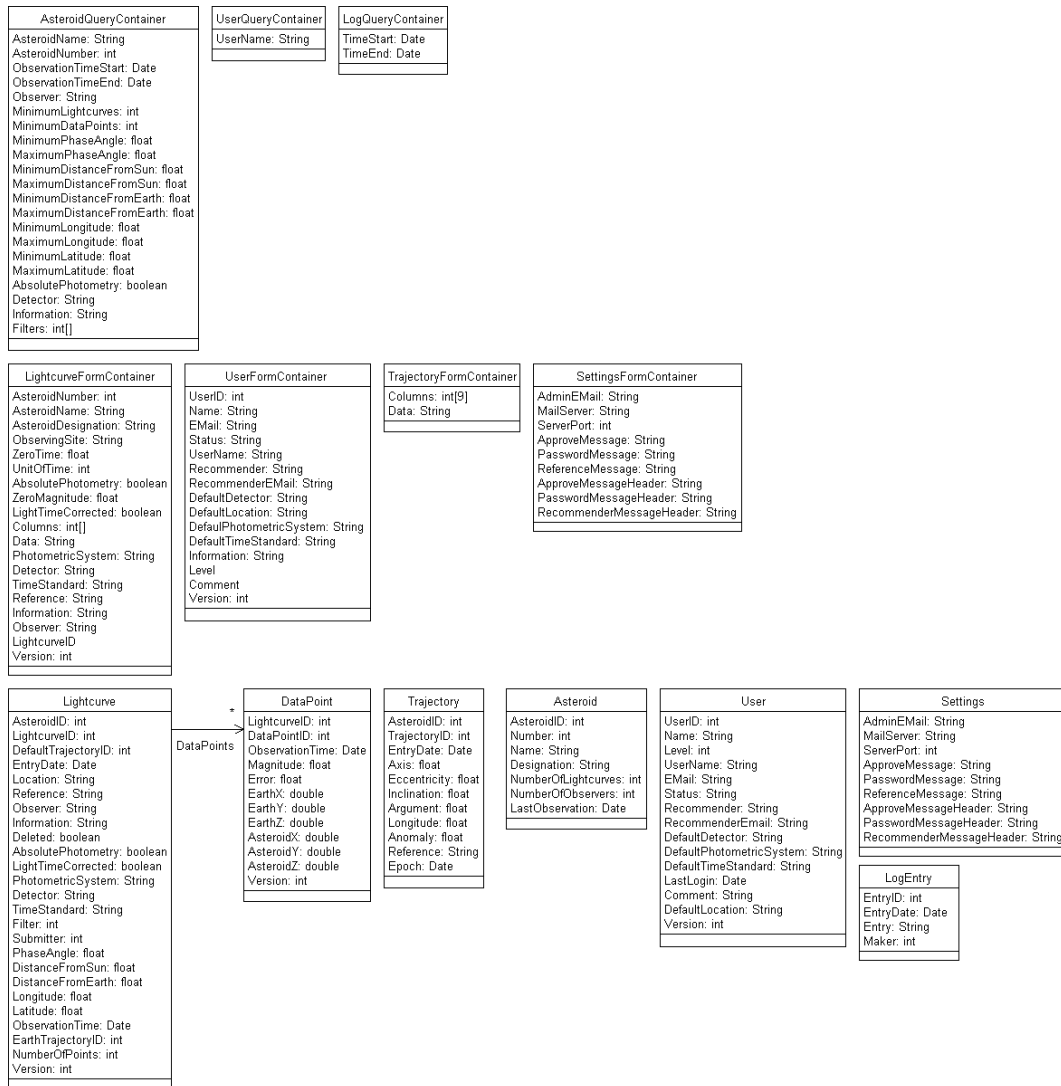
Joihinkin container-rajapintojen attribuutteihin liittyy rajoitteita. Rajoitteilla tarkoitetaan arvoväljejä, joiden rajoissa attribuutin arvon tulee olla. Nämä attribuutit tulisi tarkistaa käytettäessä container-rajapinnan toteuttavan luokan attribuutteja. Attribuutteihin liittyvät rajoitteet esitetään aliluvuissa lueteltujen attribuuttien perässä käyttäen merkintöjä `>`, `<`, `>=` ja `<=`. Merkinnän perässä on vastaava rajoittava arvo. Esimerkiksi merkintä `>0` tarkoittaa, että attribuutin arvo on suurempi kuin nolla.

Tässä luvussa on ensin kuvattu kaikki järjestelmän sisältämät container-rajapinnat kuvan avulla, jonka jälkeen jokaisesta rajapinnasta on oma alilukunsa, jossa luetellaan rajapintaan liittyvät attribuutit, ja selitetään, mitä kukin attribuutti sisältää. Selitys on suluisa attribuutin jälkeen. Kuva poikkeaa toteutuksesta siten, että niiden container-luokkien, joiden nimessä esiintyy sana Form tai Query, attribuutit ovat rajapinnassa merkkijonotyyppiä. Tämä on tarkoituksenmukaista siksi, että käyttöliittymä voi helposti täyttää vain merkkijonomuotoa olevia kenttiä ja että tyhjän kentän esittäminen helpottuisi. Kuvassa Form- tai Query-luokkien attribuuttien tyypit kertovat, mitä tyyppiä merkkijonomuodossa annettu attribuutti oikeasti on. Lopuksi esitellään luokka TahitiLibrary, ja sen sisältämät metodit ja vakiot.

#### 3.1 Container-rajapinnat

Kuvassa 2 ovat kaikki järjestelmän container-rajapinnat. Kukin rajapinta on kuvattu omana luokkana, jossa get- ja set-metodien sijaan on lueteltu kaikki luokan attribuutit. Jokaista attribuuttia vastaa toteutettaessa get- ja set-metodi. Attribuutin nimen jälkeen on kerrottu attribuutin

tyyppi, joka on samalla get-metodin palautustyyppi ja set-metodin parametrityyppi, lukuunottamatta Form- ja Query-luokkia, joissa attribuutit palautetaan merkkijonotyyppisinä. Lightcurve-containerista DataPoint-containeriin menevä nuoli tarkoittaa, että Lightcurve-container sisältää attribuuttina taulukon DataPoint-containereita.



Kuva 2: Container-rajapinnat

### 3.2 AsteroidQueryContainer

AsteroidQueryContainer on suunniteltu välittämään tietoa, jonka perusteella valokäyriä voidaan hakea järjestelmästä. Attribuutit ovat talletettu containeriin merkkijonoina. Se sisältää attribuutit:

- AsteroidName (Asteroidin nimi)
- AsteroidNumber (Asteroidin numero)  $\geq 0$
- ObservationTimeStart (Havaintoaikahaarukan alkuhetki)

- ObservationTimeEnd (Havaintoaikahaarukan loppuhetki)
- Observer (Havainnoija)
- Location (sijainti)
- MinimumLightcurves (Asteroidiin liittyvien valokäyrien vähimmäismäärä)  $> 0$
- MinimumDataPoints (Valokäyrässä olevien havaintopisteiden vähimmäismäärä)  $> 0$
- MinimumPhaseAngle (Asteroidin vaihekulman vähimmäisarvo valokäyrän havaintohetkellä)  $\geq 0, \leq 180$
- MaximumPhaseAngle (Asteroidin vaihekulman enimmäisarvo valokäyrän havaintohetkellä)  $\geq 0, \leq 180$
- MinimumDistanceFromSun (Asteroidin etäisyyden auringosta vähimmäisarvo valokäyrän havaintohetkellä)  $\geq 0$
- MaximumDistanceFromSun (Asteroidin etäisyyden auringosta enimmäisarvo valokäyrän havaintohetkellä)  $\geq 0$
- MinimumDistanceFromEarth (Asteroidin etäisyyden maasta vähimmäisarvo valokäyrän havaintohetkellä)  $\geq 0$
- MaximumDistanceFromEarth (Asteroidin etäisyyden maasta enimmäisarvo valokäyrän havaintohetkellä)  $\geq 0$
- MinimumLongitude (Asteroidin ekliptikaalisen leveyden vähimmäisarvo valokäyrän havaintohetkellä)
- MaximumLongitude (Asteroidin ekliptikaalisen leveyden enimmäisarvo valokäyrän havaintohetkellä)
- MinimumLatitude (Asteroidin ekliptikaalisen pituuden vähimmäisarvo valokäyrän havaintohetkellä)  $\geq -90, \leq 90$
- MaximumLatitude (Asteroidin ekliptikaalisen pituuden enimmäisarvo valokäyrän havaintohetkellä)  $\geq -90, \leq 90$
- AbsolutePhotometry (Tieto siitä, onko valokäyrän mittauksissa käytetty suhteellisia vai absoluuttisia arvoja)
- LighttimeCorrected (valoaikakorjattu)
- Detector (Havainnointilaite)
- Information (Lisätietoa)
- Filters (Käytetyt suotimet) Jokin TahitiLibrary luokan COL-vakioista

### 3.3 UserQueryContainer

UserQueryContainer on suunniteltu välittämään tietoa, jonka perusteella käyttäjiä voidaan hakea järjestelmästä. Attribuutit ovat talletettu containeriin merkkijonoina. Se sisältää seuraavat attribuutit:

- UserName (Käyttäjän nimi)

### 3.4 LogQueryContainer

LogQueryContainer on suunniteltu välittämään tietoa, jonka perusteella lokimerkintöjä voidaan hakea järjestelmästä. Attribuutit ovat talletettu containeriin merkkijonoina. Se sisältää seuraavat attribuutit:

- TimeStart (Lokikirjausten tapahtumahetken alkamisajankohta)
- TimeEnd (Lokikirjausten tapahtumahetken loppumisajankohta)

### 3.5 LightcurveFormContainer

LightCurveFormContainer on suunniteltu välittämään järjestelmään talletettavaan valokäyrään liittyvät tiedot. Attribuutit ovat talletettu containeriin merkkijonoina. Se sisältää seuraavat attribuutit:

- LightcurveID (Valokäyrän järjestelmän sisäinen numero)  $> 0$
- AsteroidNumber (Valokäyrään liittyvän asteroidin numero)  $> 0$
- AsteroidName (Valokäyrään liittyvän asteroidin nimi)
- AsteroidDesignation (Valokäyrään liittyvän asteroidin tunnus)
- ObservingSite (Havainnointipaikka)
- Observer (Havainnoitsija)
- ZeroTime (Datatiedoston havaintoaikojen nollassaika)  $> 0$
- UnitOfTime (Datatiedoston havaintoaikojen yksikkö) Jokin TahitiLibraryn UNIT-vakioista
- AbsolutePhotometry (Tieto siitä, ovatko mittaukset absoluuttisia)
- ZeroMagnitude (Datatiedoston havaintojen kirkkauden nollassaika)
- LightTimeCorrected (Tieto siitä, onko data valoajakorjattu)
- Columns (Datatiedoston sarakkeiden järjestys) Jokin TahitiLibraryn COL-vakioista
- Data (Datatiedosto)
- PhotometricSystem (Fotometrinen järjestelmä)

- Detector (Havainnointilaite)
- TimeStandard (Käytetty ajan mittausväline)
- Reference (Julkaisut)
- Information (Lisätietoa)
- Version (valokäyrätietojen versionumero)

### 3.6 UserFormContainer

UserFormContainer on suunniteltu välittämään järjestelmään tietoa, kun käyttäjätietoihin tehdään muutoksia tai järjestelmään kirjataan uusi rekisteröintipyyntö. Attribuutit ovat talletettu containeriin merkkijonoina. Se sisältää seuraavat attribuutit:

- UserID (Järjestelmän sisäinen käyttäjätunnus)  $> 0$
- Name (Käyttäjän nimi)
- EMail (Käyttäjän sähköpostiosoite)
- Status (Käyttäjän status)
- UserName (Käyttäjän kirjautumistunnus)
- Recommender (Suositteleva)
- RecommenderEMail (suosittelevan sähköpostiosoite)
- DefaultDetector (Käyttäjän oletushavainnointilaite)
- DefaultLocation (Käyttäjän oletussijainti)
- DefaultPhotometricSystem (Käyttäjän oletettu fotometrinen järjestelmä)
- DefaultTimeStandard (Käyttäjän oletusaikajärjestelmä)
- Information (Lisätietoa käyttäjästä)
- Level (Käyttäjän käyttäjätaso) Jokin TahitiLibraryn USER-vakioista
- Comment (Ylläpitäjän käyttäjästä kirjoittama kommentti)
- Version (käyttäjätietojen versionumero)

### 3.7 TrajectoryFormContainer

TrajectoryFormContainer on suunniteltu välittämään järjestelmään talletettavat rataelementtitiedot. Attribuutit ovat talletettu containeriin merkkijonoina. Se sisältää seuraavat attribuutit:

- Columns (Datatiedoston sarakkeiden järjestys) Jokin TahitiLibraryn COL-vakio
- Data (Datatiedosto)

### 3.8 SettingsFormContainer

SettingsFormContainer on suunniteltu välittämään järjestelmään tietoa, kun asetuksiin tehdään muutoksia. Attribuutit ovat talletettu containeriin merkijoina. Se sisältää seuraavat attribuutit:

- AdminEMail (Ylläpitäjän sähköpostiosoite)
- MailServer (Järjestelmän postipalvelin)
- ServerPort (Järjestelmän postipalvelimen portti)  $\geq 0$
- ApproveMessage (Käyttäjälle lähetettävä hyväksymissähköposti)
- ApproveMessageHeader (hyväksymispostin otsake)
- PasswordMessage (Käyttäjälle lähetettävä salasananollousposti)
- PasswordMessageHeader (salasanapostin otsake)
- ReferenceMessage (Suositelijalle lähetettävä tiedustelusähköposti)
- ReferenceMessageHeader (tiedustelupostin otsake)

### 3.9 Lightcurve

Lightcurve on suunniteltu sisältämään tietoa valokäyrästä, jota järjestelmä palauttaa sille tehdyn haun tuloksena. Se sisältää seuraavat attribuutit:

- AsteroidID (Valokäyrään liittyvän asteroidin järjestelmän sisäinen numero)  $>0$
- LightcurveID (Valokäyrän järjestelmän sisäinen numero)  $>0$
- DefaultTrajectoryID (Valokäyrään liittyvän asteroidin rataelementin järjestelmän sisäinen numero. Sen rataelementin numero, jonka voimassaoloaika on lähinnä ennen valokäyrän havainnointiaikaa)  $>0$
- EarthTrajectory (Maan rataelementin tunnus, jonka voimassaoloaika on lähinnä ennen valokäyrän havainnointiaikaa)  $>0$
- EntryDate (Valokäyrän järjestelmään kirjaamisen ajankohta)
- Location (Sijainti)
- Reference (Julkaisut)
- Observer (Havainnoitsijat)
- Information (Lisätietoa)
- Deleted (Poistomerkintä)
- AbsolutePhotometry (Absoluuttinen fotometria)

- LightTimeCorrected (Valoaikakorjaus)
- PhotometricSystem (Fotometrinen järjestelmä)
- Detector (Havainnointiväline)
- TimeStandard (Aikajärjestelmä)
- Filter (Suodin) Jokin TahitiLibraryn COL-vakioista
- Submitter (Valokäyrän järjestelmään lisännen käyttäjän järjestelmän sisäinen numero)
- PhaseAngle (Valokäyrään liittyvän asteroidin vaihekulma havaintohetkellä)  $\geq 0$ ,  $\leq 180$
- DistanceFromSun (Valokäyrään liittyvän asteroidin etäisyys auringosta havaintohetkellä)  $> 0$
- DistanceFromEarth (Valokäyrään liittyvän asteroidin etäisyys maasta havaintohetkellä)  $> 0$
- Longitude (Valokäyrään liittyvän asteroidin ekliptikaalinen pituus havaintohetkellä)
- Latitude (Valokäyrään liittyvän asteroidin ekliptikaalinen leveys havaintohetkellä)  $\geq -90$ ,  $\leq 90$
- ObservationTime (Valokäyrän havainnointiaika, eli valokäyrän ensimmäisen havainnon havainnointiaika)
- NumberOfPoints (Valokäyrään liittyvien havaintojen lukumäärä)  $\geq 0$
- DataPoint (Taulukko valokäyrän havaintoja)
- Version (valokäyrätietojen versionumero)

### 3.10 DataPoint

DataPoint on suunniteltu sisältämään valokäyrän yksittäisen havaintopisteen tiedot. Se sisältää attribuutit:

- LightcurveID (Havaintopisteeseen liittyvän valokäyrän järjestelmän sisäinen numero)  $> 0$
- DataPointID (Havaintopisteen järjestelmän sisäinen numero)  $> 0$
- ObservationTime (Havaintoaika)
- Magnitude (Kirkkaus)
- EarthX (Maan x-koordinaatti aurinkokeskeisessä avaruudessa laskettuna oletusarvoisille rataelementeillä)
- EarthY (Maan y-koordinaatti aurinkokeskeisessä avaruudessa laskettuna oletusarvoisille rataelementeillä)
- EarthZ (Maan z-koordinaatti aurinkokeskeisessä avaruudessa laskettuna oletusarvoisille rataelementeillä)



- AsteroidX (Asteroidin x-koordinaatti aurinkokeskeisessä avaruudessa laskettuna oletusarvoisilla rataelementeillä)
- AsteroidY (Asteroidin y-koordinaatti aurinkokeskeisessä avaruudessa laskettuna oletusarvoisilla rataelementeillä)
- AsteroidZ (Asteroidin z-koordinaatti aurinkokeskeisessä avaruudessa laskettuna oletusarvoisilla rataelementeillä)
- Error (Virhemarginaali)
- Version (Datapisteen versionumero)

### 3.11 Trajectory

Trajectory on suunniteltu sisältämään tietyn asteroidin rataelementtitiedot tietyllä ajan hetkellä. Se sisältää attribuutit.

- AsteroidID (Rataelementtitietoihin liittyvän asteroidin järjestelmän sisäinen numero)  $> 0$
- TrajectoryID (Rataelementtitiedon järjestelmän sisäinen numero)  $> 0$
- EntryData (Rataelementtitietojen järjestelmäänkirjausaika)  $> 0$
- Axis
- Eccentricity  $\geq 0, \leq 1$
- Inclination  $\geq 0, < 360$
- Argument  $\geq 0, < 360$
- Longitude  $\geq 0, < 360$
- Anomaly  $\geq 0 < 360$
- Reference
- Epoch

### 3.12 Asteroid

Asteroid on suunniteltu sisältämään yhteenvetotietoa asteroidista tehdyistä havainnoista, sekä asteroidin nimitiedot. Se sisältää attribuutit.

- AsteroidID (Asteroidin järjestelmän sisäinen numero)  $> 0$
- Number (Asteroidin numero)  $\geq 0$
- Name (Asteroidin nimi)
- Designation (Asteroidin tunnus)

- NumberOfLightcurves (Asteroidiin liittyvien valokäyrien lukumäärä)  $\geq 0$
- NumberOfObservers (Asteroidia havainnoidein ryhmien ja henkilöiden lukumäärä)  $\geq 0$
- LastObservation (Viimeisimmän asteroidista havainnoidun valokäyrän havainnointiaika)

### 3.13 User

User on suunniteltu sisältämään tiedot, jotka järjestelmä palauttaa käyttäjiä haettaessa. Se sisältää seuraavat attribuutit:

- UserID (Järjestelmän sisäinen käyttäjätunnus)  $> 0$
- Name (Käyttäjän nimi)
- Level (Käyttäjän käyttäjätaso) Jokin TahitiLibraryn USER-vakioista
- EMail (Käyttäjän sähköpostiosoite)
- Status (Käyttäjän status)
- UserName (Käyttäjän kirjautumistunnus)
- Recommender (Suositteleva)
- RecommenderEMail (suosittelevan sähköpostiosoite)
- DefaultDetector (Käyttäjän oletushavainnointilaite)
- DefaultLocation (Käyttäjän oletussijainti)
- DefaultPhotometricSystem (Käyttäjän oletettu fotometrinen järjestelmä)
- DefaultTimeStandard (Käyttäjän oletusaikajärjestelmä)
- Information (Lisätietoa käyttäjästä)
- Comment (Ylläpitäjän käyttäjästä kirjoittama kommentti)
- LastLogin (Käyttäjän viimeisin sisäänkirjautumisaika)
- Version (käyttäjätietojen versionumero)

### 3.14 Settings

Settings on suunniteltu sisältämään järjestelmän asetustiedot. Se sisältää seuraavat attribuutit:

- AdminEMail (Ylläpitäjän sähköpostiosoite)
- MailServer (Järjestelmän postipalvelin)
- ServerPort (Järjestelmän postipalvelimen portti)  $\geq 0$

- ApproveMessage (Käyttäjälle lähetettävä hyväksymissähköposti)
- ApproveMessageHeader (hyväksymispostin otsake)
- PasswordMessage (Käyttäjälle lähetettävä salasananollousposti)
- PasswordMessageHeader (salasanapostin otsake)
- ReferenceMessage (Suositelijalle lähetettävä tiedustelusähköposti)
- ReferenceMessageHeader (tiedustelupostin otsake)

### 3.15 LogEntry

LogEntry on suunniteltu sisältämään tiedot yhdestä lokimerkinnästä. Se sisältää seuraavat attribuutit:

- EntryNumber (Järjestelmän sisäinen lokimerkinnän numero)
- EntryDate (Lokimerkinnän teko aika)
- Entry (Lokimerkintä)
- Maker (Merkinnän tehneen käyttäjän järjestelmän sisäinen numero)

### 3.16 TahitiLibrary

TahitiLibrary-luokka sisältää järjestelmässä käytössä olevat julkiset vakiot, sekä järjestelmän käyttämät julkiset apumetodit. Ensinnäkin on lueteltu järjestelmän vakiot, minkä jälkeen selitetään metodit. Lisäksi järjestelmän käytössä on matemaattisia standardivakioita kuten  $\pi$ . COL-alkuisia vakioita käytetään rivien järjestyksen määräämiseen valokäyrä- ja ratatietoa syötettäessä, USER-alkuiset vakiot ovat käyttäjätasojen ja loppujen vakioita käytetään laskettaessa rataelementeistä koordinaatteja.

- COL\_TIME = 0
- COL\_ERROR = 1
- COL\_ULTRAVIOLET = 2
- COL\_BLUE = 3
- COL\_VISUAL = 4
- COL\_RED = 5
- COL\_INFRARED = 6
- COL\_UNFILTERED = 7

TahitiLibrary
<pre> +COL_TIME: int = 0 {static,final} +COL_ERROR: int = 1 {static,final} +COL_ULTRAVIOLET: int = 2 {static,final} +COL_BLUE: int = 3 {static,final} +COL_VISUAL: int = 4 {static,final} +COL_RED: int = 5 {static,final} +COL_INFRARED: int = 6 {static,final} +COL_UNFILTERED: int = 7 {static,final} +COL_EPOCH: int = 8 {static,final} +COL_AXIS: int = 9 {static,final} +COL_ECCENTRICITY: int = 10 {static,final} +COL_INCLINATION: int = 11 {static,final} +COL_LONGITUDE: int = 12 {static,final} +COL_ARGUMENT: int = 13 {static,final} +COL_ANOMALY: int = 14 {static,final} +COL_REFERENCE: int = 15 {static,final} +USER_BROWSER: int = 0 {static,final} +USER_SUBMITTER: int = 1 {static,final} +USER_ADMIN: int = 2 {static,final} +MEAN_ANOMALY_ITERATION_MAX: int {static} +TRAJECTORY_ERROR_TOLERANCE: float {static} +DAYS_PER_YEAR: float = 365.2568983263 {static,final} +USER_DELETED: int = -1 {static,final} +UNIT_DAY: int = 0 {static,final} +UNIT_HOUR: int = 1 {static,final} +SECONDS_PER_DAY: int = 86400 {static,final} +KILOMETERS_PER_AU: float = 149597870.691 {static,final} +SPEED_OF_LIGHT: float = 299792.458 {static,final} +ABSOLUTE_MAGNITUDE_CONSTANT: int = 5 {static,final} </pre>
<pre> +countCoordinates(moment: float, traj: Trajectory): int[3] +distance(v: float[3]): float +countAlpha(v: float[3], u: float[3]): float +countEclipticalLongitude(asteroid: float[3], earth: float[3]): float +countEclipticalLatitude(asteroid: float[3], earth: float[3]): float +lighttimeCorrect(t: float, asteroid: float[3]): float +reduceMagnitude(m: float, asteroid: float[3]): float </pre>

Kuva 3: TahitiLibrary

- COL\_EPOCH = 8
- COL\_AXIS = 9
- COL\_ECCENTRICITY = 10
- COL\_INCLINATION = 11
- COL\_LONGITUDE = 12
- COL\_ARGUMENT = 13
- COL\_ANOMALY = 14
- COL\_REFERENCE = 15
  
- USER\_DELETED = -1
- USER\_BROWSER = 0
- USER\_SUBMITTER = 1
- USER\_ADMIN = 2
  
- UNIT\_DAY = 0
- UNIT\_HOUR = 1
  
- MEAN\_ANOMALY\_ITERATION\_MAX = 1000
- TRAJECTORY\_ERROR\_TOLERANCE =  $1^{-12}$
- DAYS\_PER\_YEAR = 365.2568983263
- SECONDS\_PER\_DAY = 86400
- KILOMETERS\_PER\_AU = 149597870,691 (km/AU)
- SPEED\_OF\_LIGHT = 299792,458 (km/s)
- ABSOLUTE\_MAGNITUDE\_CONSTANT = 5

**countCoordinates** Metodi laskee ajanhetken ja radan saatuaan asteroidin sijainnin aurinkoon nähden, ja palauttaa sijainnin euklidisessa avaruudessa. Laskennassa joudutaan iteroimaan keskianomaliaa (Mean Anomaly), ja jos asteroidin rata on äärimmäisen omituinen, saattaa iterointi kestää hyvinkin pitkään. Tästä johtuen, jos iteroinnissa tehdään iterointeja enemmän, kuin vakion MEAN\_ANOMALY\_ITERATION\_MAX ilmoittama määrä, metodi palauttaa nollavektorin. Lisäksi laskennassa käytetään vakiota TRAJECTORY\_ERROR\_TOLERANCE, jonka avulla ilmaistaan, kuinka tarkkasti ratayhtälö ratkaistaan ja vakiota DAYS\_PER\_YEAR, joka kertoo, montako päivää vuodessa on.

- Syötteet:

**Date** moment

**Trajectory** traj

- Palauttaa:

float[3] r

Koordinaatit saadaan seuraavasti:

$$C = M + \frac{2\pi(t - t_0)}{(\sqrt{a^3} * DPY)}$$

, missä  $M$  on rataelementti Anomaly,  $t$  ajanhetki,  $t_0$  rataelementti Epoch,  $a$  rataelementti Axis ja DPY vakio DAYS\_PER\_YEAR.

$$f(i) = \begin{cases} C + \frac{e \sin C}{1 - e \cos(C)} & , i = 0 \\ C + e \sin f(i - 1) & , i > 0 \end{cases}$$

, missä  $e$  on rataelementti Eccentricity.

Funktiota  $f$  iteroidaan kunnes muutos on pienempi kuin vakio TRAJECTORY\_ERROR\_TOLERANCE tai  $i$  ylittää vakion MEAN\_ANOMALY\_ITERATION\_MAX. Merkitään lopetusehdot täyttävää funktion arvoa  $f(c) = F$ . Seuraavassa vaiheessa saadaan lopputulos  $\vec{r}$  matriisikertolaskulla:

$$\vec{r} = \begin{pmatrix} \cos \omega \cos \Omega - \sin \omega \sin \Omega \cos i & -\cos \omega \sin \Omega \cos i - \sin \omega \cos \Omega & \sin \Omega \sin i \\ \sin \omega \sin \Omega \cos i + \cos \omega \sin \Omega & \cos \omega \cos \Omega \cos i - \sin \omega \sin \Omega & -\cos \Omega \sin i \\ \sin \omega \sin i & \cos \omega \sin i & \cos i \end{pmatrix} \begin{pmatrix} a * (\cos F - e) \\ a * \sqrt{1 - e^2} \sin F \\ 0 \end{pmatrix}$$

, missä  $a, e, i, \omega$  ja  $\Omega$  ovat rataelementtejä.

**distance** Funktio distance laskee kolmiulotteisen vektorin pituuden.

- Syötteet:

float[3] v

- Palauttaa:

float

Pituus  $d$  saadaan kaavalla  $d = \sqrt{v_1^2 + v_2^2 + v_3^2}$ , missä  $v_1, v_2$  ja  $v_3$  ovat syötteenä saatavan vektorin  $\vec{v}$  komponentit.

**countAlpha** Funktio countAlpha laskee kahden vektorin välisen vaihekulman.

- Syötteet:

float[3] v

float[3] u

- Palauttaa:

**float**

Vaihekulma  $\alpha$  saadaan kaavalla

$$\alpha = \arccos \frac{\vec{v} \cdot \vec{u}}{|\vec{v}| |\vec{u}|}$$

**countEclipticLongitude** Funktio countEclipticLongitude laskee ekliptikaalisen pituusasteen saadessaan syötteen asteroidin ja maan paikkavektorit.

- Syötteen:

float[3] asteroid

float[3] earth

- Palauttaa:

**float**

Ekliptikaalinen pituus  $\lambda$  saadaan kaavalla

$$\lambda = \arctan(a_2 - e_2, a_1 - e_1)$$

, missä  $a_i$  ja  $e_i$  ovat vektoreiden  $\vec{asteroid}$  ja  $\vec{earth}$   $i$ :nnet komponentit.

**countEclipticLatitude** Funktio countEclipticLatitude laskee ekliptikaalisen leveysasteen saadessaan syötteen asteroidin ja maan paikkavektorit.

- Syötteen:

float[3] asteroid

float[3] earth

- Palauttaa:

**float**

Ekliptikaalinen leveys  $\beta$  saadaan kaavalla

$$\beta = \arcsin \frac{a_3 - e_3}{|\vec{asteroid} - \vec{earth}|}$$

, missä  $a_i$  ja  $e_i$  ovat vektoreiden  $\vec{asteroid}$  ja  $\vec{earth}$   $i$ :nnet komponentit.

**lighttimeCorrect** Funktio lighttimeCorrect laskee valoaikakorjauksen juliaaniselle päiväykselle tietyssä paikassa olevalle kappaleelle. Ja palauttaa korjatun ajan.

- Syötteet:

float t

float[3] asteroid

- Palauttaa:

**float**

Valoaikakorjattu aika  $t_C$  saadaan kaavasta  $t_C = t - \frac{\delta E}{cF}$ , missä  $E$  on vakio KILOMETERS\_PER\_AU,  $c$  on vakio SPEED\_OF\_LIGHT,  $F$  on vakio SECONDS\_PER\_DAY ja  $\delta$  on asteroidin etäisyys maasta.

**reduceMagnitude** Funktio reduceMagnitude muuttaa kirkkauden vastaamaan kirkkautta, joka asteroidilla olisi jos se olisi yhden valovuoden päässä maasta saadessaan parametrinaan kirkkauden ja asteroidin ja maan sijainnit. Kirkkauden muutos tehdään vain absoluuttisille mittauksille.

- Syötteet:

float m

float[3] asteroid

float[3] earth

- Palauttaa:

**float**

Muunnettu kirkkaus  $m_C$  saadaan kaavasta  $m_C = m - E \log(|\vec{asteroid}| |\vec{asteroid} - \vec{earth}|)$ , missä  $E$  on vakio ABSOLUTE\_MAGNITUDE\_CONSTANT.

## 4 Tahiti–Api

Tässä luvussa perehdytään Tahiti–Apin rakenteeseen. Luvussa Tahiti–Apia kutsutaan lyhesti Apiksi.

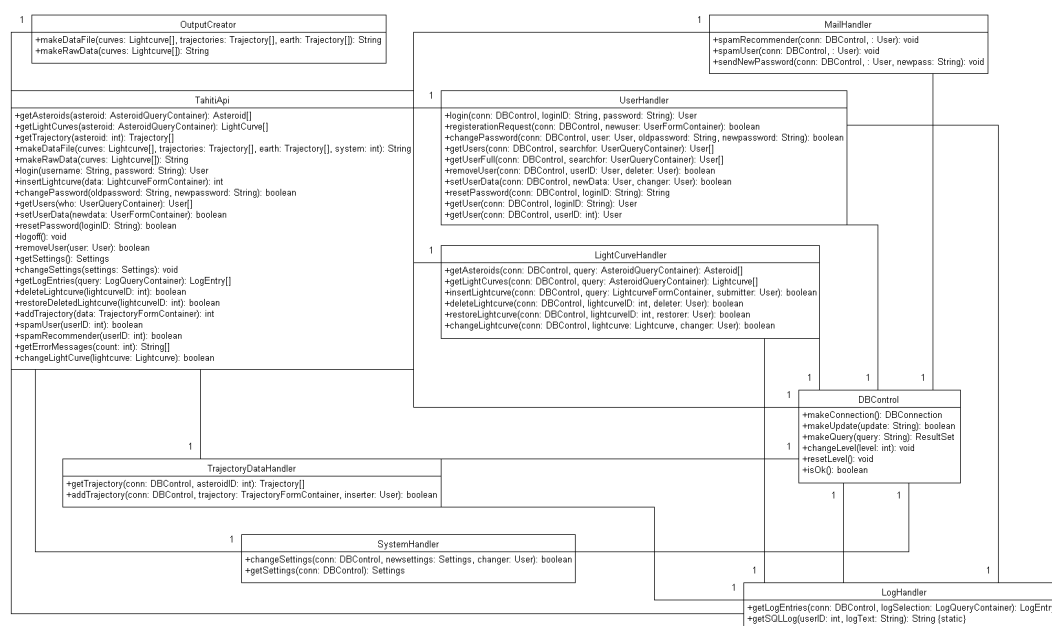
Apia käytetään rajapintana, jonka kautta käyttöliittymä pystyy käyttämään erilaisia järjestelmän palveluita. Itse Apiin ei rakenneta varsinaista toiminnallisuutta, vaan Apin tehtävänä on lähinnä kontrolloida palveluiden käyttöä, ohjata pyynnöt oikeille käsittelijöille sekä tarjota yhtenäinen, selkeä rajapinta järjestelmän tarjoamille palveluille. Apista muodostetaan ilmentymä jokaista järjestelmän käyttäjää kohden. Tämä mahdollistaa sen, että Api tietää aina, kuka sen palveluita on käyttämässä. Tällöin Apin ei tarvitse luottaa ylempältä tasolta tulevaan oikeusinformaatioon, eikä jokaiseen pyyntöön tarvitse erikseen liittää käyttäjätunnusta.

Apin alapuolelta löytyvät luokat voidaan jakaa käsittelijäluokkiin sekä muihin luokkiin. Käsittelijäluokkia ovat seuraavat luokat:



- TrajectoryHandler
- UserHandler
- LightcurveHandler
- MailerHandler
- SystemHandler
- OutputCreator

Jokainen käsittelijä luokka vastaa pienestä järjestelmän osa-alueesta, sekä osa-alueeseen liittyvien toimintojen toteuttamisesta. Jako on tehty siten, ettei yksikään käsittelijä ei ole tekemisissä toisensa kanssa, jolloin kutsurakenteet on saatu pysymään yksinkertaisina sekä testaaminen saadaan yksinkertaistettua. Niissä Api-kutsuissa, joissa tarvitaan useamman kuin yhden käsittelijän työtä, on töiden ohjaaminen pyritty tekemään sarjallisesti Api-kutsun sisään. Api ja Apin alaisuudesta löytyvät luokat ovat kuvattuina luokkakaaviossa 4.



Kuva 4: Järjestelmän luokkakaavio

Käsittelijäluokkien varsinaiset tehtävät käydään läpi luokkakohtaisissa kuvauksissa, mutta luokkien vastualueet ovat pääosin seuraavat: TrajectoryHandler tarjoaa rataelementtietojen käsittelyssä tarvittavia palveluita. LightcurveHandlerin vastuulla ovat vastaavat palvelut valokäyrien osalta. OutputCreatorin tehtävänä on muodostaa annetuista tiedoista määrämuotoisia tulostiedostoja, eikä se koske laisinkaan tietokantaan. UserHandler puolestaan tarjoaa käyttäjätietojen hallintaan tarvittavia palveluita, ja SystemHandlerin avulla ylläpitäjät voivat muuttaa järjestelmän asetuksia. Viimeinen käsittelijä, MailHandler suorittaa määrämuotoisten sähköpostiviestien muodostamisen sekä varsinaisten viestien lähettämisen. Viestin lähetys tapahtuu omassa säikeessään, eikä käyttäjä joudu siis odottamaan viestien toimittamista.

Käsittelijäluokkien lisäksi Apin alapuolelta löytyy muutamia erikoisasemassa olevia luokkia. DBControl-oliosta luodaan session alussa ilmentymä, jonka kautta kaikki istunnon tietokantaoperaatiota suoritetaan. Luotu ilmentymä annetaan parametrinä kaikille niille metodikutsuille, jotka tarvitsevat tietokantayhteyttä. LogHandler-luokkaa ei lasketa varsinaiseksi käsittelijäksi, vaikka sen kautta voidaankin myös noutaa tietokannasta lokitietoja. LogHandlerin pääasiallinen tehtävä on kuitenkin lokikirjoituksessa tarvittavien SQL-lauseiden muodostaminen, ja tämän vuoksi luokkaan tulee staattinen metodi näiden lauseiden muodostamista varten. Muutoin luokka toteutetaan kuten muutkin olio-luokat.

Lokikirjoitus toteutetaan järjestelmässä siten, että varsinaiset muutokset sekä lokitietojen kirjoitus tehdään yhdessä atomisessa transaktiossa. Tämä on mahdollista siksi, ettei lokikirjoittaja tee muutoksia kantaan, vaan palauttaa kutsuvalle käsittelijälle lokikirjoitukseen tarvittavat SQL-lauseet. Kaikki tarvittavat SQL-lauseet kootaan tämän jälkeen yhteen taulukkoon, joka annetaan tietokantahallinnalle suoritettavaksi.

## 4.1 Yleistä Apista

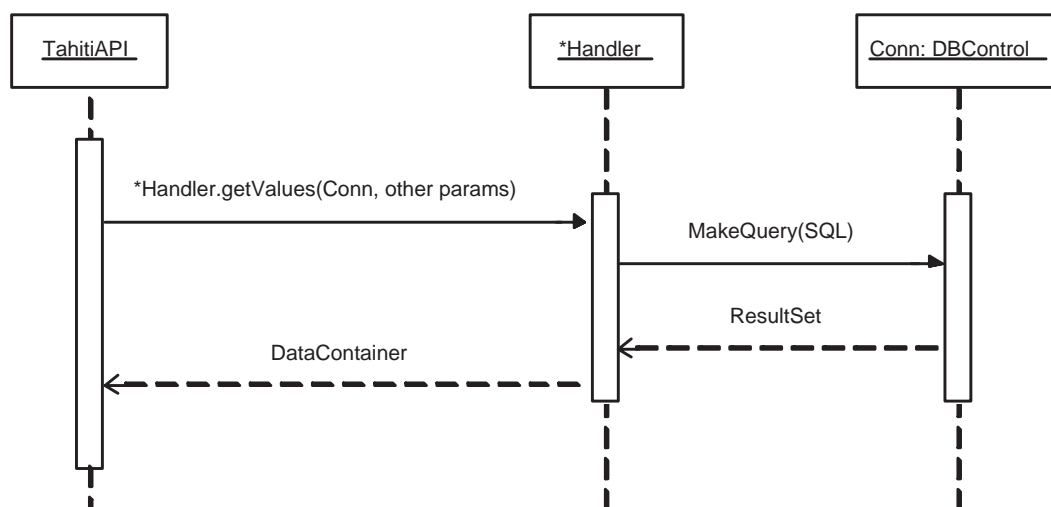
Kun ilmentymä Apista luodaan, luodaan ilmentymälle automaattisesti DBControl-ilmentymä, jonka avulla tietokantaoperaatiot suoritetaan. Muista oliopohjaisista käsittelijöistä luodaan ilmentymiä vain tarvittaessa. Käytännössä tämä tarkoittaa sitä, että jokaisen Api-metodikutsun yhteydessä Api tarkistaa, onko tarvittavista käsittelijöistä jo olemassa ilmentymä, ja mikäli tällaista ei ole, ilmentymä luodaan ja viite luotuun ilmentymään tallennetaan Apin tietorakenteeseen.

Luokkaviitteiden lisäksi Apin tietorakenteisiin tallennetaan tietoja käsittelyssä tapahtuvista virheistä, mihin palataan käsittelijäluokkien yhteisissä piirteissä, jossa käsitellään virhekäsittelyyn liittyvät asiat. Api pitää myös tallessa sisällään tiedon ilmentymäkohtaisesta käyttäjästä, minkä avulla käyttöoikeustarkistukset voidaan tehdä helposti. Käyttöoikeudet tarkistetaan ja metodien käyttö kontrolloidaan vain Apissa, vaikkakin muutamat palvelumetodit saattavat myöskin käyttää istuntokohtaisen käyttäjän tietoja hyväkseen varsinaisessa käsittelyssä. Mikäli käsittelijä tarvitsee näitä käyttäjätietoja, antaa Api parametrinä viitteen sisäiseen käyttäjäolioonsa.

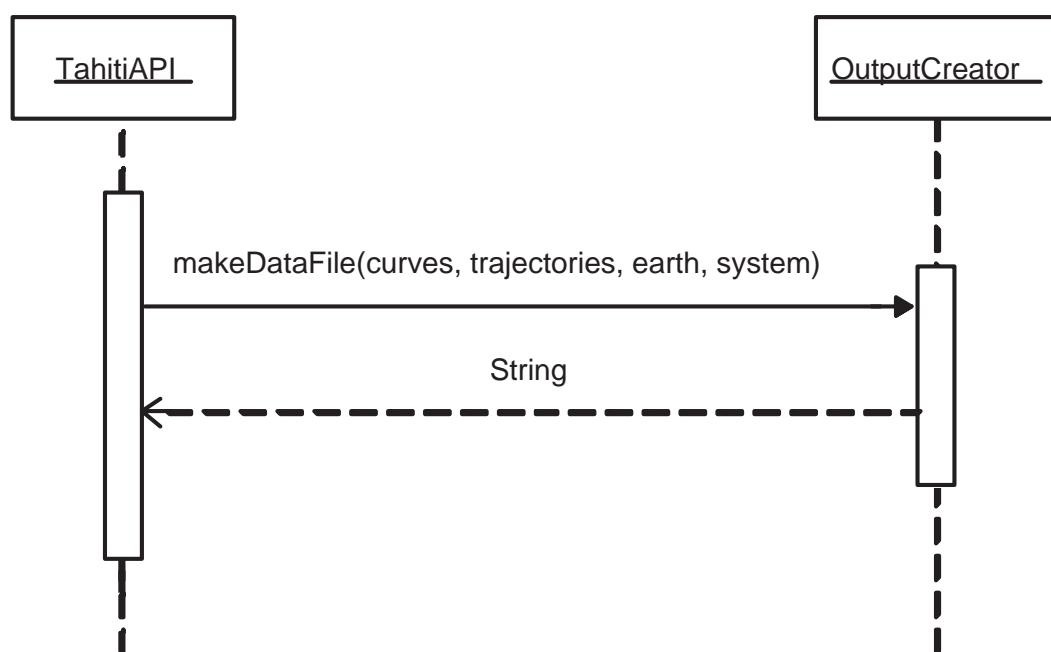
## 4.2 Kutsurakenteesta

Järjestelmän Apin alainen kutsurakenne on melko suoraviivainen. Käsittelijät eivät ole yleisesti tekemissä toistensa kanssa, eikä järjestelmässä ole callback-tyyppisiä kutsurakenteita. Kuvasta 5 selviää, kuinka kontrolli kulkee järjestelmän sisällä normaalissa tiedonhaussa. Kaikki järjestelmän tietohaut tapahtuvat pääpiirteittäin tämän saman mallin mukaisesti, paitsi selaaajien tekemät käyttäjähaut, joiden rakenne poikkeaa tästä rakenteesta hiukan. Tähän poikkeamaan palataan myöhemmin. Normaalissa haussa Api kutsuu Handler-luokan metodia, joka tekee varsinaisen työn. Käsittelijä-luokkien metodit saavat myös parametrinä käytettävän tietokantayhteyden, jonka kautta tietokantaan kohdistuvat operaatiot suoritetaan.

Kuvasta 6 selviää kutsurakenne tulostietojen muodostuksessa. OutputCreator ei käytä tietokantayhteyttä, vaan kaikki tarvittava tieto tulee suoraan metodikutsujen mukana. Api kutsuu OutputCreatorin metodia niillä datoilla, joita tulostietojen muodostamiseen tarvitaan.

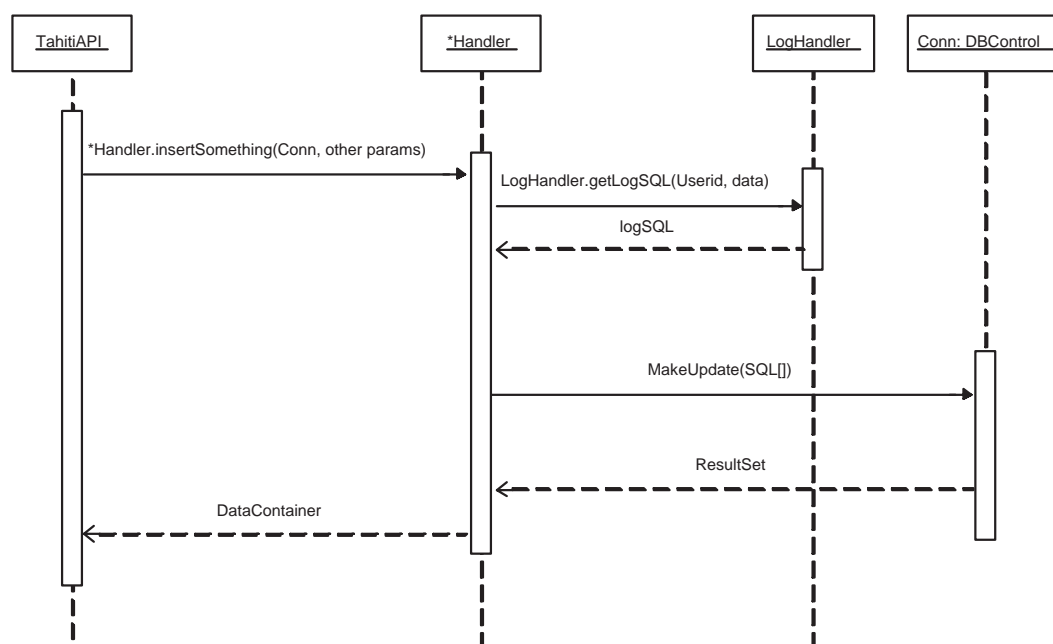


Kuva 5: Normaalin haun sekvenssikaavio



Kuva 6: Tulostietojen muodostuksen sekvenssikaavio

Tietojen lisääminen järjestelmään tapahtuu myös tietynlaisen kaavan mukaisesti. Tämä rakenne selviää kuvasta 7. Aluksi Api kutsuu käsittelijää, joka käy annetut syötteet läpi sekä muodostaa lisäämiseen tarvittavat SQL-lauseet. Tämän jälkeen käsittelijä kutsuu LogHandleria, joka palauttaa käsittelijälle lokikirjoitukseen tarvittavat SQL-lauseet. Lopuksi lisäykset ja lokimuutokset annetaan DBControl-ilmentymälle, joka tekee muutokset tietokantaan. Samaa kutsurakennetta käytetään myös valokäyrätietoihin kohdistuvien muutosten kanssa.

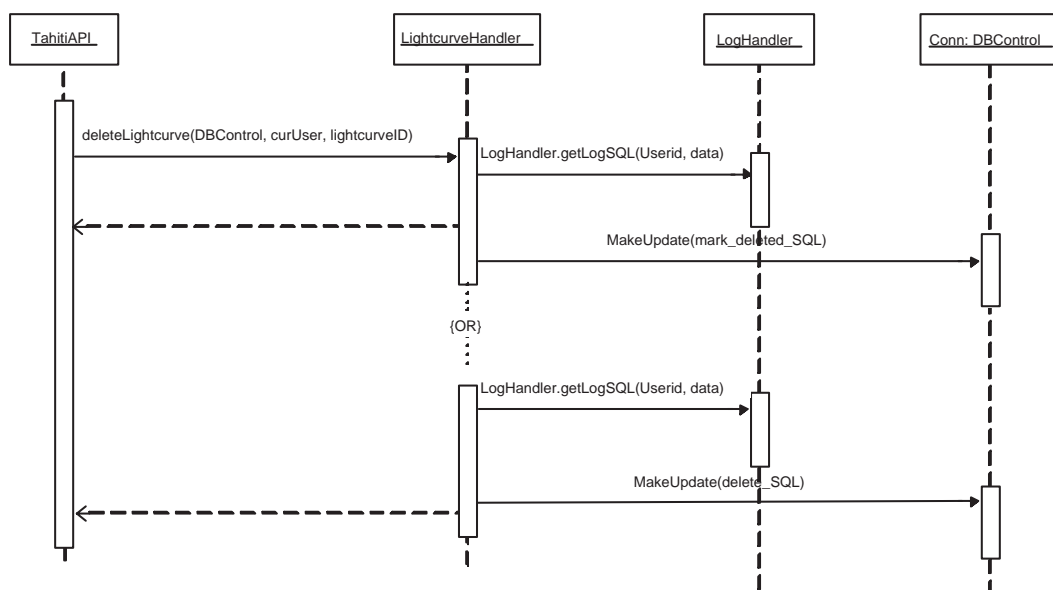


Kuva 7: Normaalin lisäyksen sekvenssikaavio

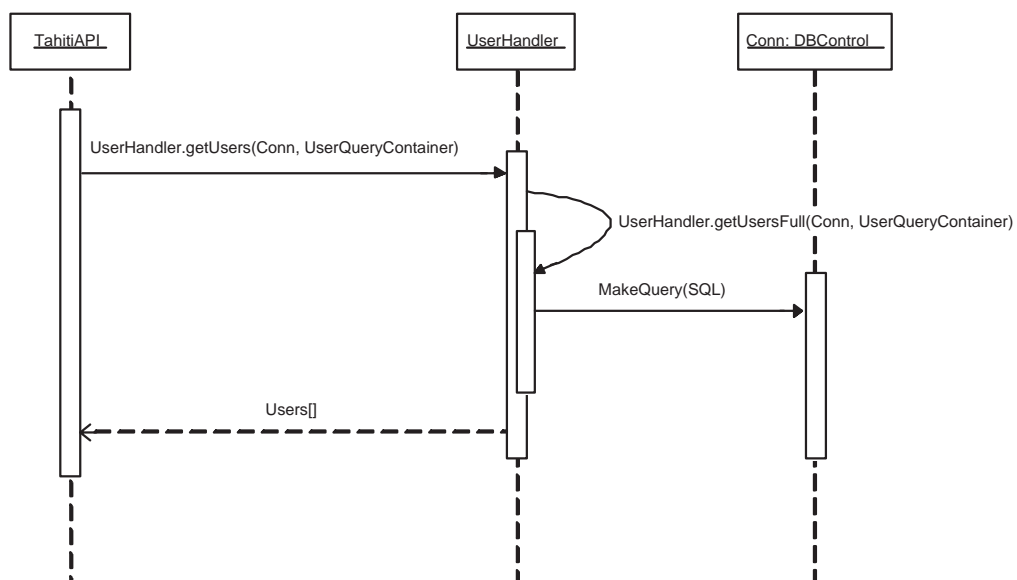
Valokäyrien poistossa esiintyvä kutsurakenne ei poikkea suuresti lisäyksessä esiintyvistä rakenteesta. Poiston toimintaa kuvaa kuva 8. Eron valokäyrien lisäämiseen muodostaa poiston kaksivaiheisuus: aluksi valokäyrä merkitään poistetuksi, ja vasta tämän jälkeen se voidaan poistaa järjestelmästä kokonaan. Kokonaan poistaminen on mahdollista vain, mikäli käyttäjä on tasoltaan ylläpitäjä.

Syöttäjä-tason käyttäjille tarjottava käyttäjähaku poikkeaa normaaleista hauista siinä, että UserHandlerin getUsers-metodi kutsuu saman luokan getUsersFull-metodia, joka suorittaa varsinaisen haun. Tulosten saamisen jälkeen getUsers-metodi poistaa saaduista tuloksista ne tiedot, joita syöttäjille ei näytetä. Tämä rakenne selviää kuvasta 9.

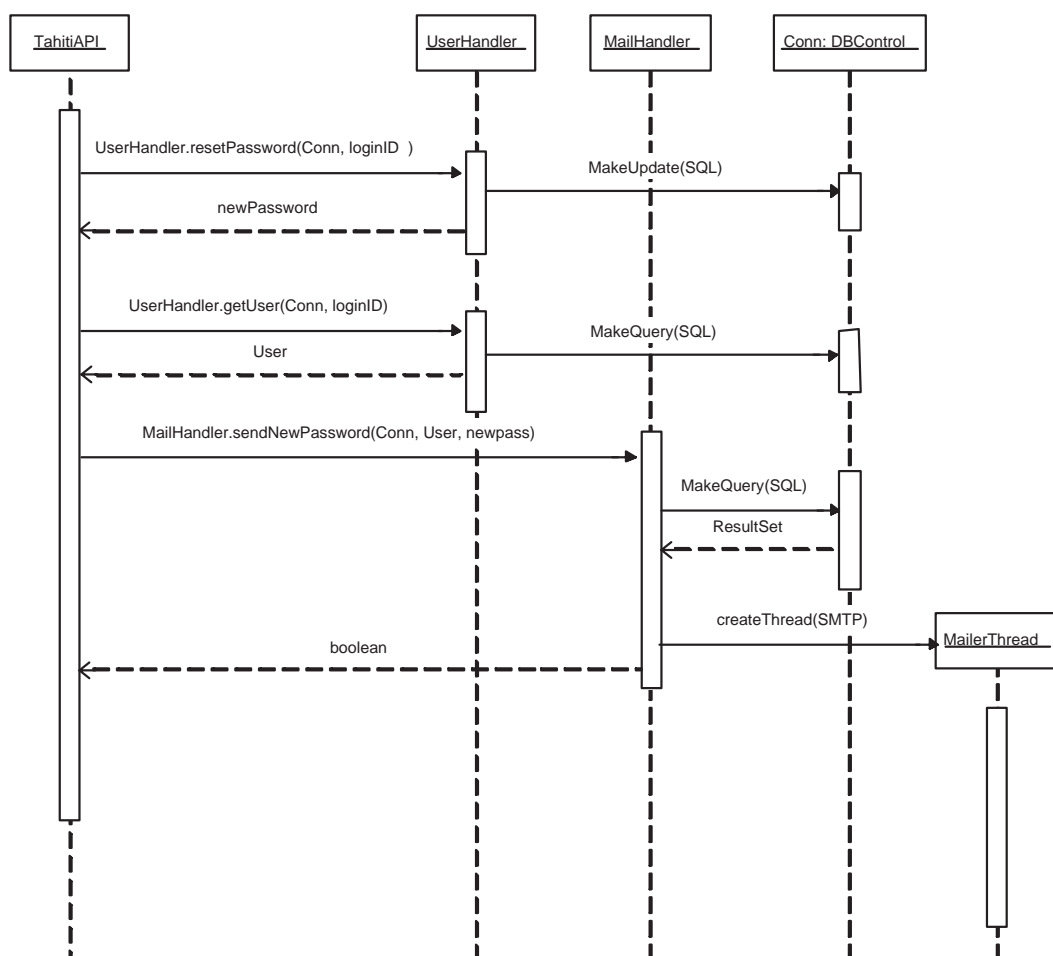
Käyttäjän salasanan resetointi on yksi järjestelmän monimutkaisimpia toimintoja. Aluksi kutsutaan UserHandlerin metodia, jolloin käyttäjälle generoidaan uusi, satunnainen salasana. Tämän jälkeen tämä arvottu salasana palautetaan Apille, joka kutsuu seuraavaksi UserHandlerin getUser-metodia, joka palauttaa sen käyttäjän tiedot, jonka salasana juuri resetoitiin. Lopuksi annetaan nämä tiedot MailHandlerille, joka muodostaa määrämukoisen sähköpostin, jossa uusi salasana kerrotaan käyttäjälle. Varsinaista sähköpostilähetystä varten käynnistetään oma säie. Tämä rakenne selviää kuvasta 10. Myös käyttöoikeuksien myöntämisessä käytettävä Api-kutsu noudattaa samankaltaista rakennetta.



Kuva 8: Valokäyrän poiston sekvenssikaavio

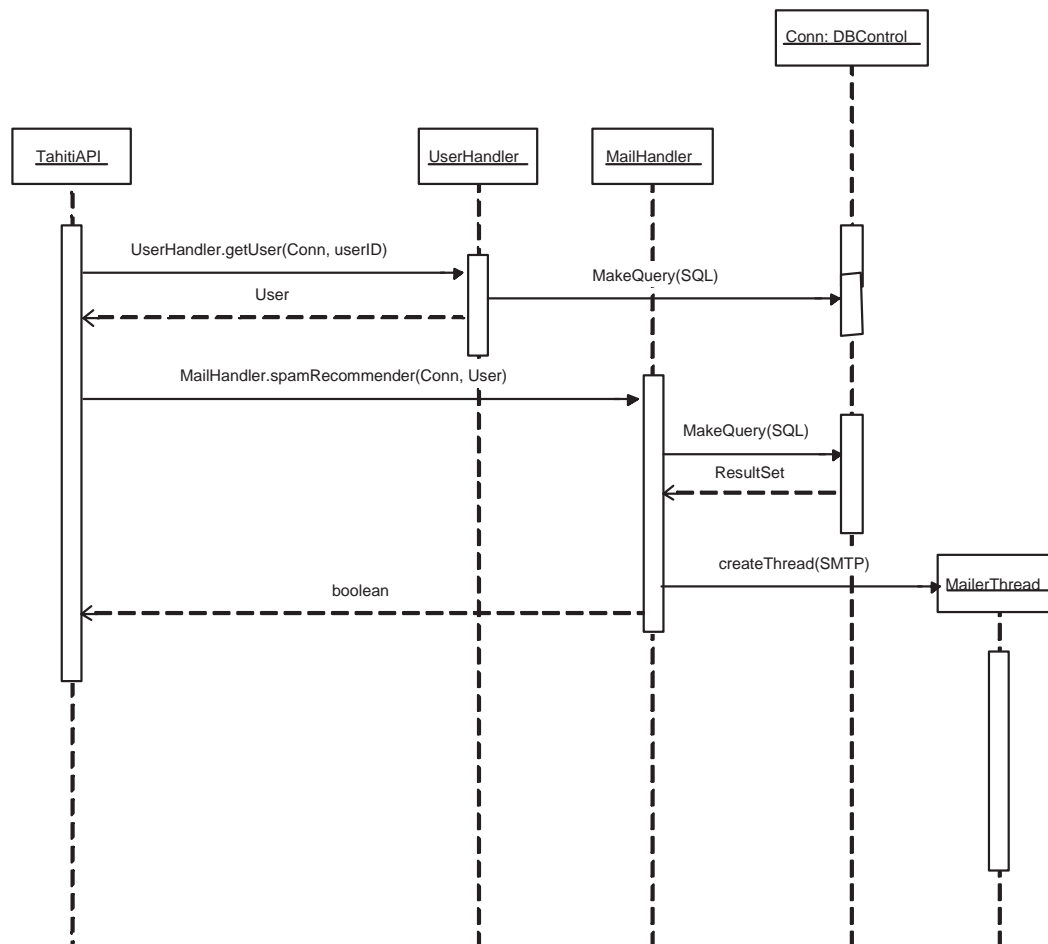


Kuva 9: Syöttäjätason käyttäjähaun sekvenssikaavio



Kuva 10: Salasanan resetoinnin sekvenssikaavio

Hyväksyjälle lähetettävän sähköpostin lähetyksessä käytettävä kontrollirakenne selviää kuvasta 11. Rakenne on muuten hyvin samankaltainen salasanan resetoinnin kanssa, mutta nyt ei tarvita kahta UserHandlerin metodikutsua.



Kuva 11: Sähköpostin lähettämisen sekvenssikaavio

### 4.3 Käsittelijäluokkien yhteisiä piirteitä

Jokainen Apin alapuolelta löytyvä Käsittelijä-olio sisältää tiettyjä yhteisiä piirteitä. Jokaiseen luokkaan kuuluu oma sisäinen errorMessages-muuttuja, johon tieto kaikista käsittelyssä mahdollisesti tapahtuvista virheistä laitetaan. Lisäksi jokaisesta luokasta löytyy getErrorMessages-metodi, jolla tapahtuneet virheet saadaan palautettua ylemmälle tasolle. Käsittelijäluokkien konstruktorit eivät yleisesti ottaen suorita minkäänlaisia erityisiä operaatioita (poislukien DBControl), vaan niitä tarvitaan vain olioiden muodostamiseen. Kaikki syötteiden tarkistamiset ovat myöskin käsittelijöiden vastuulla.

Edellämainittu getErrorMessages-metodi ei tarvitse toimiakseen minkäänlaisia parametrejä, vaan se palauttaa kaikki käsittelijän sisäisessä virhemuuttujassa olevat virheet int-taulukossa kutsujalleen. Samalla kun virhetiedot siirretään ylemmälle tasolle, poistetaan tieto näistä aikaisemmin

tapahtuneista virheistä sisäisestä muuttujasta. Käsittelijäoliot eivät siis kerää minkäänlaista historiaa istunnon aikana tapahtuneista virheistä.

Myös TahitiApi-oliossa käytetään samaa logiikkaa virheiden käsittelyyn. Api tarkistaa jokaisen saamansa palautusarvon yhteydessä, onko virheitä tapahtunut. Mikäli paluuarvo kertoo virheestä, kutsuu Api kyseisen käsittelijän `getErrorMessage`-metodia, ja kirjoittaa saadut tiedot omaan virhemuuttujaansa. Vasta tämän jälkeen palautetaan tulos varsinaiselle kysyjälle eli käyttöliittymälle.

Järjestelmän virheistä kertovia paluuarvoja ovat boolean-muuttujien `false`-arvot sekä oiloiden `null`-viitteet. Niissä Apin metodeissa, joissa palvelun toteuttaminen vaatii useamman kuin yhden käsittelijän käyttöä, lopetetaan suoritus heti ensimmäiseen virheeseen. Samaa kerrasta poikkiperiaatetta noudatetaan myös käsittelijöiden sisällä. Mikäli kutsuttavan metodin palauttama virhetieto on eri tyyppiä kuin Api-kutsun normaalisti palauttava tieto, joutuu Api vielä muodostamaan tilanteeseen sopivan virheestä kertovan palautusarvon, joka sitten palautetaan ylemmän tason kysyjälle.

Käsittelijät tarkistavat syötteet virheellisten tietojen varalta ennen käsittelyn jatkamista. Tähän tarkistamiseen kuuluu sekä joillekin tiedoille tapahtuvat oikeellisuustarkistukset että SQL-komentojen poistaminen annetuista syötteistä. Syötteiden oikeellisuudesta kerrotaan tarkemmin luvussa 8. Mikäli syötteet sisältävät pakollisia tietokenttiä, tarkistetaan myös, että nämä tiedot ovat täytettyinä.

Luokkien `LightcurveHandler` ja `UserHandler` yhteydessä saattaa esiintyä niin kutsuttuja likaisia kirjoituksia, joissa myöhemmin tehtävä kirjoitusoperaatio ylikirjoittaa aikaisemman kirjoituksen tulokset. Näihin varaudutaan käyttämällä erityisiä versionumeroita käyttäjätietojen ja valokäyrien muutoksissa.

Versionumeroiden toimintalogiikka on sellainen, että päivitystä tehdessään käyttäjän antamiin tietoihin liitetään versionumero, johon muutokset perustuvat. Käsittelijäluokka sitten tekee päivityksen tietokantaan vain, mikäli käyttäjän antama, päivitettävä versionumero täsmää kannassa olevan objektin versioonumeroon. Samalla myös versionumeroa kasvatetaan.

## 4.4 Lokikirjoituksesta

Järjestelmän lokikirjoitus toteutetaan normaalien tietokantaoperaatioiden yhteyteen atomisesti. Tässä luvussa on kuvattu, missä muodossa lokitekstit missäkin tapauksessa kirjoitetaan tietokantaan. Tietokantaan tallettavien monikkojen muoto selviää tietokantakuvauksesta, kun taas tässä luvussa keskitytään vain Entry-kentän sisältämään tietoon. Entry-kenttään siis talletetaan varsinaiseen operaatioon liittyvää tietoa.

Entry-kentässä oleva tieto jaetaan viiteen osaan. Näiden osien välissä käytetään erotinmerkkinä kahta peräkkäistä putkimerkkiä (`||`). Entry-kentän sisäiset tietokentät ovat seuraavat:

- Luokka:

Luokalla tarkoitetaan sitä käsittelijäluokkaa, joka muutoksen on tehnyt. Tämän kentän avulla tapahtumalokia selaava ylläpitäjä näkee nopeasti, mihin tapahtunut muutos on saattanut vaikuttaa.



- Teksti:

Tämä kenttä sisältää informaation varsinaisesta tapahtumasta. Esimerkiksi valokäyrää lisätessä tekstikentässä lukee: 'New observation submitted for asteroid X'

- Kentät

Jos kyseessä on muutos, kerrotaan tässä niiden kenttien nimet, joiden arvot muuttuivat muutoksen yhteydessä. Kuten muissakin moniarvoisissa kentissä, erotinmerkkinä käytetään pilkkua.

- Uudet arvot

Tästä osiosta löytyvät kaikki operaatioissa kirjatut uudet arvot. Arvot on erotettu toisistaan pilkuilla.

- Vanhat arvot

Mikäli kyseessä on muutos, löytyy tästä kentästä muutettujen kenttien vanhat arvot. Sisäisenä erotinmerkkinä käytetään pilkkua.

Kaikissa lokikirjoitusta vaativissa operaatioissa ei kuitenkaan kaikkia edellämainittuja kenttiä täytetä. Esimerkiksi tiedon syöttämisessä ei alkuperäisiä arvoja ole olemassa. Nämä tyhjät kentät kuitenkin ovat mukana talletetuissa riveissä.

## 5 Tahiti–Apin luokat

Apin luokat vastaavat järjestelmän palveluiden toteuttamisesta. Esimerkiksi lokin kirjoitus, tietojen käsittely ja tallentaminen sekä tietokantahaut ovat apin apuluokkien tehtävinä.

Kuten edellisessä luvussa kerrottiin, Api toimii rajapintana, jonka kautta käyttöliittymä voi hyödyntää erilaisia järjestelmän palveluita. Samalla mainittiin, että Apin alaisuudessa toimivat erilaiset käsittelijät, jotka hoitavat varsinaisen työn. Tässä luvussa esitellään ensiksi Api ja sitten sen alaisuudessa toimivat käsittelijäluokat. Koska Apin metodien nimet ovat hyvin kuvaavia, ja koska varsinaisen toiminnallisuus on todellakin sijoitettu käsittelijöihin, Apin metodien toimintaa on kuvattu vain niissä tapauksissa, joissa se on katsottu tarpeelliseksi. Jokaisen Apin metodin esittelyn yhteydessä on kuitenkin listattu kyseisen Api-metodin kutsumat käsittelijöiden metodit. Käsittelijöiden esittelyssä käsittelijöiden metodit on selitetty kattavasti.

**Konstruktori** Apin konstruktori luo DBControl-ilmentymän, joka luo yhteyden tietokantaan.

Tietokantayhteyttä tarvitaan lähes kaikissa järjestelmän tarjoamissa palveluissa.

- Kutsuttavat metodit:

```
new DBControl()
```

**getAsteroids**

- Syötteen:

**AsteroidQueryContainer** asteroid

- Palauttaa:

**Asteroid**[ ]

- Kutsuttavat metodit:

**LightcurveHandler.getAsteroids(DBControl, asteroid)**

**getLightcurves**

- Syötteet:

**AsteroidQueryContainer** asteroid

- Palauttaa:

**Lightcurve**[ ]

- Kutsuttavat metodit:

**LightcurveHandler.getLightcurves(DBControl, asteroid)**

**getTrajectory**

- Syötteet:

**int** asteroid

- Palauttaa:

**Trajectory**[ ]

- Kutsuttavat metodit:

**TrajectoryHandler.getTrajectory(DBControl, asteroid)**

**makeDataFile**

- Syötteet:

**Lightcurve**[ ] curves

**Trajectory**[ ] trajectories

**Trajectory**[ ] earth

**int** system

- Palauttaa:

**String**

- Kutsuttavat metodit:

`OutputCreator.makeDataFile(curves, trajectories, earth, system)`

#### `makeRawData`

- Syötteet:  
`Lightcurve[ ] curves`
- Palauttaa:  
`String`
- Kutsuttavat metodit:  
`OutputCreator.makeRawData(curves)`

#### `registrationRequest`

- Syötteet:  
`UserFormContainer newuser`
- Palauttaa:  
`boolean`
- Kutsuttavat metodit:  
`UserHandler.registrationRequest(DBControl, newuser)`

**login** Kirjautumisen yhteydessä muutetaan tietokantayhteyden oikeustasoa. Tämä tapahtuu `DBControl`-luokan `changeLevel`-metodin avulla.

- Syötteet:  
`String username`  
`String password`
- Palauttaa:  
`User`
- Kutsuttavat metodit:  
`UserHandler.Login(DBControl,user, pass)`  
`Conn.changeLevel((int)curUser.userLevel)`

#### `insertLightcurve`

- Syötteet:  
`LightcurveFormContainer data`

- Palauttaa:

**boolean**

- Käyttöedellytykset:

**Vähintään syöttäjätason oikeudet**

- Kutsuttavat metodit:

**LightcurveHandler.insertLightcurve(DBControl, data, curUser)**

**insertLightcurveAtlas** Tätä asteroidin lisäysmetodia käytetään vain Atlas-syöttimen avulla. Se eroaa normaalista insertLightcurve-metodista siinä, että tälle metodille annetaan parametreinä myös havaintoon liittyvät, kiinteät ratatiedot.

- Syötteet:

**LightcurveFormContainer** data

**double[ ]** earthAspect

**double[ ]** asteroidAspect

- Palauttaa:

**boolean**

- Käyttöedellytykset:

**Vähintään syöttäjätason oikeudet**

- Kutsuttavat metodit:

**LightcurveHandler.insertLightcurveAtlas(DBControl, data, earthAspect, asteroidAspect, curU**

### **changePassword**

- Syötteet:

**String** oldpassword

**String** newpassword

- Palauttaa:

**Boolean**

- Käyttöedellytykset:

**Vähintään syöttäjätason oikeudet**

- Kutsuttavat metodit:

**UserHandler.changePassword(DBConn, curUser, oldpassword, newpassword)**

**getUsers** Käsittelijässä haku on hajautettu kahteen eri metodiin, joista ensimmäinen, `UserHandler.getUsers`, toimii syöttäjätason hakukoneena. Se palauttaa kaikki normaalit syöttäjille näkyvät tiedot annettuihin hakuehtoihin sopivista käyttäjistä. Toinen metodi, `UserHandler.getUsersFull`, on puolestaan ylläpitotason hauissa käytettävä metodi. Se palauttaa käyttäjistä myös sellaisia tietoa, joita käyttäjät eivät itse näe, kuten kommentitiedot. Kumpikaan metodi ei palauta käyttäjien salasanoja missään muodossa.

- Syötteen:

**UserQueryContainer who**

- Palauttaa:

**User[ ]**

- Käyttöedellytykset:

**Vähintään syöttäjätason oikeudet**

- Kutsuttavat metodit:

**UserHandler.getUsers(DBControl, who)**

**UserHandler.getUserFull(DBControl, who)**

**setUserData** Käyttäjätietojen muuntaminen on sallittua, mikäli muutettavan käyttäjän numero täsmää istunnon omistajan numeroon tai muuntaja on käyttäjätasoltaan ylläpitäjä. Mikäli muutoksen tekijänä on ylläpitäjä, tekee käsittelijä tapahtuneesta merkinnän järjestelmän lokiin.

- Syötteen:

**UserFormContainer newdata**

- Palauttaa:

**boolean**

- Käyttöedellytykset:

**Vähintään syöttäjätason oikeudet**

**Ylläpito-oikeudet yleiseen käyttöön**

- Kutsuttavat metodit:

**UserHandler.setUserData(DBControl, newdata, curUser)**

**resetPassword** Aluksi Api kutsuu `UserHandler.resetPassword`:ia, joka palauttaa uuden salasanan selkokielisenä Apille. Tämän jälkeen kutsutaan `UserHandler.getUser`:ia, jonka avulla saadaan kaikki käyttäjän tiedot Apille. Lopuksi Api kutsuu saaduilla tiedoilla `MailHandler.sendNewPassword`:ia, joka lähettää käyttäjälle tiedon uudesta salasanasta sähköpostissa.

- Syötteet:

**String** loginID

- Palauttaa:

**boolean**

- Kutsuttavat metodit:

**UserHandler.resetPassword(DBControl, loginID)**

**UserHandler.getUser(DBControl, loginID)**

**MailHandler.sendNewPassword(DBControl, User, newpass)**

### **changeLightcurve**

- Syötteet:

**LightcurveFormContainer** lightcurve

- Palauttaa:

**boolean**

- Käyttöedellytykset:

**Vain ylläpitäjien käytössä**

- Kutsuttavat metodit:

**LightcurveHandler.changeLightcurve(DBControl, lightcurve, curUser)**

**logoff** Tämä metodi ei tarvitse syötteitä, eikä palauta mitään. Se pelkästään nolaa API:n tilatiedoista löytyvän User-muuttujan sekä muuntaa tietokantayhteydessä käytettävät käyttöoikeudet vastaavalle tasolle.

- Käyttettävät metodit:

**Conn.changeLevel(1)**

### **removeUser**

- Syötteet:

**User** user

- Palauttaa:

**boolean**

- Käyttöedellytykset:

**Ylläpitäjä**

- Kutsuttavat metodit:

**UserHandler.removeUser(DBControl, user, curUser)**

**getSettings**

- Palauttaa:

**Settings**

- Käyttöödellytykset:

**Ylläpitäjä**

- Kutsuttavat metodit:

**SystemHandler.getSettings(DBControl)**

**changeSettings**

- Syötteet:

**Settings newSettings**

- Käyttöödellytykset:

**Ylläpitäjä**

- Kutsuttavat metodit:

**SystemHandler.changeSettings(DBControl, settings, curUser)**

**getLogEntries**

- Syötteet:

**LoqQueryContainer query**

- Palauttaa:

**LogEntry[ ]**

- Käyttöödellytykset:

**Ylläpitäjä**

- Kutsuttavat metodit:

**LogHandler.getLogEntries(DBControl, query)**

**deleteLightcurve**

- Syötteet:

**int** lightcurveID

- Palauttaa:

**boolean**

- Käyttöedellytykset:

**Ylläpitäjä**

- Kutsuttavat metodit:

**LightcurveHandler.deleteLightcurve(DBControl, lightcurveID, curUser)**

#### **restoreDeletedLightcurve**

- Syötteet:

**int** lightcurveID

- Palauttaa:

**boolean**

- Käyttöedellytykset:

**Ylläpitäjä**

- Kutsuttavat metodit:

**LightcurveHandler.restoreLightcurve(DBControl, lightcurveID, curUser)**

#### **addTrajectory**

- Syötteet:

**TrajectoryFormContainer** data

- Palauttaa:

**boolean**

- Käyttöedellytykset:

**Ylläpitäjä**

- Kutsuttavat metodit:

**TrajectoryHandler.addTrajectory(DBControl, data, curUser)**



**spamRecommender** Tämän metodin avulla voidaan lähettää suosittelijalle määrämuotoinen sähköposti, jolla pyydetään suositteluiden vahvistamista. Aluksi API-kutsuu `UserHandler.getUser:ia`, jolla saadaan käyttäjän tiedot. Tämän jälkeen nämä tiedot annetaan `MailHandler`lle, joka lähettää suosittelijalle sähköpostin.

- Syötteet:

**int** UserID

- Palauttaa:

**boolean**

- Käyttöedellytykset:

**Ylläpitäjä**

- Kutsuttavat metodit:

**UserHandler.getUser(DBControl, userID)**

**MailHandler.spamRecommender(DBControl, User)**

**spamUser** Tämän metodin avulla voidaan automaattisesti lähettää tieto käyttäjälle järjestelmän käyttöoikeuksien saamisesta. Samassa yhteydessä käyttäjälle myös muodostetaan satunnainen salasana, joka liitetään lähetettävään sähköpostiin. Uusi salasana muodostetaan kutsumalla `UserHandler`in `resetPassword`ia, joka palauttaa muodostetun salasana selkokielenä kutsujalle.

- Syötteet:

**int** userID

- Palauttaa:

**boolean**

- Käyttöedellytykset:

**Ylläpitäjä**

- Kutsuttavat metodit:

**UserHandler.getUser(DBControl, userID)**

**UserHandler.resetPassword(DBControl, User.loginID)**

**MailHandler.spamUser(DBControl, User, password)**

## 5.1 Luokan DBControl kuvaus

Luokka hoitaa kaikki Apin tarvitsemat tietokantaoperaatiot toimien näin eräänlaisena porttina Apin ja tietokannan välillä. Mikäli jonkin Apin metodin tarvitsee käsitellä tietokantaa, on kyseisen metodin kutsuttava DBControl-luokan metodeita. Tietokantayhteyden muodostamiseen ja tietokantakyselyihin tarvittavien metodien lisäksi luokasta löytyvät metodit myös tietokannan käyttöoikeustasojen hallintaan. Käyttöoikeustaso määrää, millaisia komentoja käyttäjä voi tietokantaan suorittaa.

**DBControl** luo uuden tietokantayhteysolion, jonka avulla luokan tietokantaoperaatiot voidaan välittää tietokannalle. Olio talletetaan luokan sisäiseen muuttujaan.

**makeUpdate** suorittaa tietokannassa parametrina annetut sql-ylläpito-operaatiot ja palauttaa int-muodossa tiedon onnistumisesta. Palautettava int-arvo sisältää tiedon muuttuneiden rivien määrästä. Toiminnan onnistuminen edellyttää yhteyden olemassaoloa.

- Syötteet:

**String**[ ] update

- Palauttaa:

**int**

**makeUpdate** suorittaa sql-parametrissa annetut sql-ylläpito-operaatiot ja tarkastaa samalla, päivittikö ne oikean, eli expectedRowCountissa annetun määrän rivejä. Jos rivejä päivittyi oikea määrä, metodi tallentaa tapahtuman kuvauksen (log) tietokantaan ja kertoo onnistumisesta palauttamalla true-arvon. Mikäli rivejä päivittyi väärä määrä tai metodin suorituksen aikana tapahtui jokin muu virhe, metodi peruu sql-ylläpito-operaatiot, jättää tapahtumakuvausten tallentamatta tietokantaan ja palauttaa false-arvon.

- Syötteet:

**String**[ ] sql

**int**[ ] expectedRowCount

**String**[ ] log

- Palauttaa:

**boolean**

**makeQuery** suorittaa tietokannassa parametrina annetun kyselyn. Paluuarvona annetaan ResultSet-olio, johon kyselyn tulokset talletetaan.

- Syötteet:

**String** query

- Palauttaa:

**ResultSet**

**changeLevel** muuttaa tietokantakäyttöoikeustason. Tasoja on kolme: selaaaja, syöttäjä ja ylläpitäjä. Uusi käyttöoikeustaso ilmoitetaan parametrina. Käyttöoikeustaso määrää suoraan, millaisia operaatiota käyttäjä saa kantaan suorittaa.

- Syötteen:

**int** level

**isOk** palauttaa tiedon tietokantayhteyden tilasta boolean-muuttujana. Jos yhteys on kunnossa, palautetaan TRUE, muutoin FALSE.

- Palauttaa:

**boolean**

**closeConnection** sulkee tietokantayhteyden. Mikäli tietokantayhteyttä ei ole olemassa tai se on jo aikaisemmin suljettu, metodi ei tee mitään. Metodilla ei ole syötteitä eikä paluuarvoja.

## 5.2 Luokan LightcurveHandler kuvaus

Luokka tarjoaa Tahiti-Apille valokäyrien käsittelyyn tarvittavat palvelut. Näitä palveluita ovat muun muassa uusien ratatietojen lisääminen järjestelmään, sekä ratatietojen hakeminen tietokannasta.

**getAsteroids** hakee tietokannasta asteroidien otsikkotietoja. Metodin toiminta on erittäin suoraviivaista, haku ehdot puretaan containerista, ja haku ehtoihin sopivien asteroidin tiedot palautetaan kutsujalle. Virhetilanteessa palautetaan null-viite, ja kirjoitetaan tieto tapahtuneesta virheestä sisäiseen virhemuuttujaan.

- Syötteen:

**DBControl** conn

**AsteroidQueryContainer** query

- Palauttaa:

**Asteroid**[ ]

**getLightcurves** tarjoaa valokäyrien tietojen hakemiseen tarvittavan palvelun. Metodi käy parametrina saadun AsteroidQueryContainerin tiedot läpi, tarkistaen mitkä kaikki kentät sisältävät dataa. Tämän jälkeen metodi muodostaa SQL-lauseen, jonka avulla varsinainen haku tehdään. Mikäli haussa ei löydy yhtään haku ehtoihin sopivaa valokäyrää, palautetaan LightCurve-tilukko, joka ei sisällä yhtään objektia. Virhetilanteessa metodi palauttaa null-viite, ja tieto virheestä kirjataan sisäiseen virhemuuttujaan.

- Syötteet:

**DBControl** conn

**AsteroidQueryContainer** query

- Palauttaa:

**Lightcurve**[ ]

**insertLightCurve** mahdollistaa valokäyrien lisäämisen järjestelmään. Aluksi metodi tarkistaa, että kaikki pakolliset tietokentät ovat täytettyinä. Tarvittavien komentojen muodostaminen tapahtuu siten, että aluksi tutkitaan kuinka monta erillistä havaintoa tietokantaan laitetaan, sillä jokainen eri filterillä tehty havainto talletetaan järjestelmään omaksi valokäyräkseen. Näiden havaintojen kaikki otsikkotiedot ovat filter-kenttää lukuunottamatta identtisiä.

Tietojen lisäämisessä käytetyt SQL-lauseet laitetaan yhteen String-taulukkoon, johon myös liitetään lokikirjoittamisessa tarvittavat SQL-lauseet. Jokaisesta järjestelmään tallennetusta erillisestä havainnosta tehdään oma lokimerkintänsä. Lopuksi kaikki komennot annetaan tietokantahallinnalle, joka tekee muutokset tietokantaan yhdessä transaktiossa. Ennen tietokantakirjoitusta metodi joutuu myös laskemaan oletusvalokäyrän perusteella asteroidin sijaintitietoja, joita jokaisesta datapointista tietokantaan talletetaan.

Mikäli lisäämisessä tapahtuu yksikin virhe, kaikki mahdollisesti jo tehdyt lisäykset perutaan. Virheistä metodi ilmoittaa normaalisti palauttamalla false, jolloin ylemmän tason käsitteily joutuu pyytämään sattuneita virheitä `getErrorMessage`-metodin avulla. Mikäli lisäys onnistuu, palauttaa metodi true.

- Syötteet:

**DBControl** conn

**LightcurveFormContainer** query

**User** submitter

- Palauttaa:

**boolean**

**insertLightcurveAtlas** Tätä metodia käytetään Atlas-syötteminen yhteydessä, ja se eroo normaalista valokäyrälisäyksestä siinä, että asteroidien ratatiedot annetaan myös parametreinä, eikä niitä siis erikseen lasketa.

- Syötteet:

**DBControl** conn

**LightcurveFormContainer** query

**double**[ ] earthAspect

**double**[ ] asteroidAspect

**User** submitter

- Palauttaa:

**boolean**

**deleteLightcurve** mahdollistaa valokäyrän poistamisen järjestelmästä. Poistaminen tapahtuu kaksiosaisesti siten, että ensimmäisellä kerralla valokäyrä vain merkataan poistetuksi. Mikäli ylläpitäjä haluaa poistaa sellaisen valokäyrän, joka on jo merkattu poistetuksi, poistetaan valokäyrä kokonaan järjestelmän tietokannasta. Valokäyrien kokonaan poistaminen on mahdollista vain, mikäli poistaja on tasoltaan ylläpitäjä. Tämä kontrolli on toteutettava metodikutsuun, koska Api ei tiedä valokäyrien tiloista mitään.

Molemmissa tapauksissa tehdystä poistosta tehdään merkintä järjestelmän tapahtumalokiin. Mikäli poistetuksimerkitseminen onnistuu, palauttaa metodi true, virhetilanteessa palautetaan false.

- Syötteen:

**DBControl** conn

**int** lightcurveID

**User** deleter

- Palauttaa:

**boolean**

**restoreLightCurve** tarjoaa mahdollisuuden poistetuiksi merkittyjen valokäyrien palauttamiseen. Metodin toiminta on suoraviivaista, ja sen käyttämisestä tehdään merkintä lokiin. Mikäli palautuksessa tapahtuu virhe, eli esimerkiksi palautettavaa valokäyrää ei ole olemassa, palautetaan false ja kirjoitetaan tieto virheestä sisäiseen virhemuuttujaan. Onnistuneesta palautuksesta metodi tiedoittaa palauttamalla true.

- Syötteen:

**DBControl** conn

**int** lightcurveID

**User** restorer

- Palauttaa:

**boolean**

**changeLightcurve** mahdollistaa jo järjestelmässä olevien valokäyrien tietojen muuttamisen. Valokäyrien muuttamisesta tehdään aina merkintä järjestelmä lokiin. Mikäli muuttamisessa tapahtuu virhe, palauttaa metodi false. Onnistuneesta muutoksesta palautetaan true.

- Syötteen:

**DBControl** conn

**LightcurveFormContainer** lightcurve

**User** changer

- Palauttaa:

**boolean**

### 5.3 Luokan LogHandler kuvaus

Luokkaan on sijoitettu lokitoimintoihin liittyvät metodit. Näitä ovat `getLogEntries`, joka hoitaa lokin lukemisen, ja `getSQLLog`, joka muuttaa annetut tekstimuotoiset lokiviestit SQL-lauseiksi, joilla lokiviestit voidaan kirjoittaa tietokantaan. Luokka ei sisällä varsinaisia lokin kirjoittamismetodeita, sillä lokin kirjoittaminen hoidetaan aina varsinaisten tietokantaoperaatioiden yhteydessä. `GetSQLLog` on kuitenkin metodi, jota hyödynnetään lokin kirjoittamistoiminnossa.

`getLogEntries` palauttaa lokitietoja `LogEntry`-taulukossa `LogQueryContainer`issa annettujen rajojen mukaisesti. Haku tapahtuu tietokannasta. Mikäli kysely jostain syystä epäonnistuu, metodi palauttaa nullin.

- Syötteen:

**DBControl** conn

**LogQueryContainer** logSelection

- Palauttaa:

**LogEntry**[ ]

`getSQLLog` muokkaa annetun lokitiedon lokitiedon tietokantaan kirjoittavaksi SQL-komennoksi ja palauttaa sen. Jos suorituksen aikana tapahtuu virhe, metodi palauttaa null-viitteen. Metodi on staattinen.

- Syötteen:

**int** userID

**String** cls

**String** entry

**String**[ ] fields

**String**[ ] oldValues

**String**[ ] newValues

- Palauttaa:

**String**

## 5.4 Luokan MailHandler kuvaus

Luokka hoitaa järjestelmän sähköpostien lähettämisen. Luokan metodit hakevat tietokannasta kulloinkin tarvittavan sähköpostin pohjan, lisäävät siihen argumentteina saatuja tietoja ja suorittavat varsinaisen sähköpostin lähettämisen erillisessä säikeessä. Metodien palauttavat boolean-arvot eivät kerro mitään varsinaisen viestin lähettämisestä, vaan saatujen palautetietojen perusteella ylempi taso saa tietoonsa, onnistuiko viestien muodostaminen laisinkaan.

Tietokannassa olevat sähköpostipohjat sisältävät erityisiä muuttujia, jotka korvataan annetuista parametreista saatavilla tiedoilla. Käytössä olevat muuttujat ovat seuraavat:

- \$username, korvataan automaattisesti parametrinä annetusta User-objektista löytyvällä käyttäjänimellä
- \$login, korvataan parametrinä annetun User-objektin kirjautumistunnuksella
- \$password, korvataan käyttäjän salasanalla, mikäli sellainen kuuluu metodin parametreihin
- \$recommender, korvataan käyttäjän suosittelijan nimellä

**spamRecommender** lähettää suosittelijalle sähköposti-ilmoituksen siitä, että joku on käyttänyt häntä suosittelijana.

- Syötteet:

**DBControl** conn

**User** user

- Palauttaa:

**boolean**

**spamUser** ilmoittaa käyttäjälle sähköpostitse, että hänet on hyväksytty syöttäjäksi. Käyttäjälle generoitu salasana kerrotaan samassa yhteydessä.

- Syötteet:

**DBControl** conn

**User** newuser

**String** password

- Palauttaa:

**boolean**

**sendNewPassword** lähettää käyttäjälle uuden salasanan. Yleisimmin käytetään, kun käyttäjä on itse pyytänyt generoitua salasanaa.

- Syötteet:

**DBControl** conn

**User** user

**String** newpass

- Palauttaa:

**boolean**

## 5.5 Luokan `OutputCreator` kuvaus

Luokka hoitaa tulosteiden muodostamisen. Tulostustiedosto muodostetaan kahdella vaihtoehtoisella tavalla, joko laskemalla valokäyriin ratatiedot tai tulostamalla pelkästään raakamuotoinen data. Kun rataelementtitiedot lasketaan mukaan, voidaan lisäksi valita tulostus magnitudin tai intensiteetin mukaan. Luokka käyttää hyväkseen `TahitiLibrary`n tarjoamia kirjastometodeita laskemaan tulostustiedostoja. Tulostustiedoston rakenne on määritelty projektin määrittelydokumentissa [Tah03].

`makeDataFile` laatii käyttäjän haluamista valokäyristä ns. tulostustiedoston. Tiedoston muoto on määritelty määrittelydokumentissa.

- Syötteet:

**Lightcurve**[ ] curves

**Trajectory**[ ] orbitals

**Trajectory** earth

**int** system

- Palauttaa:

**String** outputFile

`makeRawData` tekee tulostustiedoston muuttamatta alkuperäistä dataa mitenkään. Tieto näytetään siis samassa muodossa kuin missä se on järjestelmään talletettukin.

- Syötteet:

**Lightcurve**[ ] curves

- Palauttaa:

**String** file



## 5.6 Luokan `SystemHandler` kuvaus

Luokan vastuulla on järjestelmäasetusten hallinta. Järjestelmäasetusten lukemista ja kirjoittamista varten luokasta löytyvät omat metodinsa.

`changeSettings` tarjoaa järjestelmän asetusten muuttamiseen tarvittavan palvelun. Ennen muuttamista tarkistetaan, ettei yksikään järjestelmän asetuksista jää tyhjäksi. Varsinaisten tietojen muuttaminen tietokannassa tapahtuu yksinkertaisella SQL:n UPDATE-lauseella. Mikäli päivittämisessä tapahtuu jokin virhe, palautetaan false, ja kirjoitetaan virheen tiedot sisäiseen virhemuuttujaan.

- Syötteet:

`DBControl` conn

`Settings` newSettings

`User` changer

- Palauttaa:

`boolean`

`getSettings` mahdollistaa järjestelmän asetusten noutamiseen. Metodi palauttaa kaikki asetustaulusta löytyvät datat. Virhetilanteessa palautetaan null-viite.

- Syötteet:

`DBControl` conn

- Palauttaa:

`Settings`

## 5.7 Luokan `TrajectoryHandler` kuvaus

`TrajectoryHandler`-luokka vastaa asteroidien ratatiedoista. Luokassa ovat metodit ratatietojen lisäämistä ja lukemista varten. Ratatietoja ei milloinkaan poisteta järjestelmästä, minkä takia tällaista toimintoa ei myöskään ole olemassa.

`getTrajectory` mahdollistaa asteroidien ratatietojen hakemisen järjestelmästä. Sen käyttämiseen tarvitaan asteroidin järjestelmän sisäinen tunnus, jonka perusteella tiedot haetaan kannasta. Tietojen hakemisen jälkeen saadut tiedot laitetaan `Trajectory`-containereista muodostuvaan taulukkoon, joka palautetaan kutsuvalle API:n ilmentymälle. Mahdollisessa virhetilanteessa tiedot virheestä kirjoitetaan sisäiseen virhemuuttujaan, ja palautetaan null-viite

- Syötteet:

`DBControl` conn

**int** asteroidID

- Palauttaa:

**Trajectory**[ ]

**addTrajectory** tarjoaa palvelun ratatietojen lisäämiseen järjestelmään. Lisäämisen yhteydessä tarkistetaan myös, sisältävätkö lisättävät ratatiedot ennalta tuntemattomia asteroideja. Mikäli tällaisia tuntemattomia asteroideja löytyy, lisätään myös ne automaattisesti järjestelmään. Koska annetut ratatiedot saattavat sisältää myös sellaisia ratatietoja, joita järjestelmästä jo löytyy, tarkistetaan ennen lisäystä myös ratatietojen olemassaolo. Tarkistus tapahtuu siten, että kaikkia annettujen ratatietojen kenttiä verrataan yksi kerrallaan tietokannassa oleviin tietoihin. Tässä vertailussa otetaan huomioon ratatiedoista kaikki varsinaiset tietokennät, lukuunottamatta kenttiä `OrbitalRefence` ja `EntryDate`. Myöskään järjestelmän sisäisiä tunnuslukuja ei oteta vertailussa huomioon.

Jokaisesta lisättävästä ratatiedosta ja asteroidista muodostetaan myös merkintä järjestelmän tapahtumalokiin. Tämä tapahtuu normaalin lokinkirjoituskäytännön mukaisesti. Jos lisäyksessä tapahtuu virheitä, palauttaa metodi `false`, eikä kantaan lisätä mitään. Onnistuneesta ratatietojen lisäämisestä metodi palauttaa `true`.

- Syötteet:

**DBControl** conn

**TrajectoryFormContainer** Trajectory

**User** inserter

- Palauttaa:

**boolean**

## 5.8 Luokan `UserHandler` kuvaus

`UserHandler`-luokka sisältää kaikki käyttäjähallinnassa tarvittavat metodit. Käyttäjähallinnan laajuuden takia metodeita löytyy moniin erilaisiin tilanteisiin. Metodeita on esimerkiksi sisäänkirjautumista, salasanan muuttamista, käyttäjätietojen hakua ja muuttamista sekä käyttäjän poistamista varten.

**login** tarkistaa annetun käyttäjätunnus-salasana -parin oikeellisuuden, sekä tarkistuksen jälkeen palauttaa annetun käyttäjän tiedot `User`-objektissa. Koska salasanat ovat tietokannassa salatussa muodossa, joudutaan annettu salasana muuntamaan tähän muotoon ennen tarkistusta. Mikäli annettu käyttäjätunnus-salasana -pari löytyy tietokannasta, palautetaan tiedot kutsujalle. Mikäli tietoja ei löydy, palautetaan `null`-arvo, sekä kirjoitetaan sisäiseen virhemuuttajaan tieto tapahtuneesta (`Invalid username / password`). Lopuksi, ennen tietojen palauttamista kutsujalle, metodi vielä päivittää tietokannassa olevan käyttäjäkohtaisen `LastLogin`-kentän arvon.

- Syötteet:

**DBControl** conn

**String** loginID

**String** password

- Palauttaa:

**User**

**registrationRequest** tarjoaa uusien käyttäjien rekisteröintipyyntöpalvelun. Sen tehtävä on kirjata annetun **User**-objektin sisältämät tiedot tietokantaan. Uusista rekisteripyyntöistä tehdään myös merkintä lokiin. Uuden käyttäjän salasana jätetään tässä vaiheessa tyhjäksi, mutta hyväksymisen yhteydessä käyttäjälle muodostetaan satunnainen salasana.

Ensimmäisenä annetusta **User**-objektista puretaan käyttäjän tiedot ja tarkistetaan, että kaikki tarvittava data on annettu. Ennen varsinaisen pyynnön kirjaamista tarkistetaan vielä, ettei annettua käyttäjätunnusta ole jo olemassa. Mikäli pyynnön lisäämiselle ei ole esteitä, kutsutaan vielä **LogHandler**-luokan **getSQLLog**-metodia, joka palauttaa lokikirjoitukseen tarvittavan **SQL**-lauseen. Tämän jälkeen molemmat **SQL**-lauseet laitetaan **String**-taulukkoon, joka annetaan tietokannanhallinnalle. Jos suorittamisessa tapahtuu virheitä, kirjoitetaan virheilmoitus sisäiseen virhemuuttujaan, ja palautetaan **false**. Muussa tapauksessa palautetaan **true**.

- Syötteet:

**DBControl** conn

**UserFormContainer** newuser

- Palauttaa:

**boolean**

**changePassword** mahdollistaa käyttäjän salasanan vaihtamisen. Aluksi sekä annettu vanha salasana että uusi salasana muutetaan salattuun muotoon. Tämän jälkeen verrataan annetun käyttäjän kannassa olevaa salasanaa vanhaan, ja mikäli ne täsmäävät, päivitetään salasana uudeksi. Mikäli käsittelyssä tapahtuu virheitä tai annettu salasana ei täsmää kannassa olevaan salasanaan, palautetaan **false** ja kirjoitetaan tieto virheestä sisäiseen virhemuuttujaan. Jos vaihtaminen onnistui, palautetaan **true**.

- Syötteet:

**DBControl** conn

**User** user

**String** oldpassword

**String** newpassword

- Palauttaa:

**boolean**

**getUsers** tarjoaa syöttäjätason käyttäjille mahdollisuuden käyttäjien etsimiseen järjestelmästä.

Varsinaisen haun tekee metodi **getUserFull**, joka hakee käyttäjätiedot tietokannasta. Tämän jälkeen nollataan tuloksena saaduista **User**-objekteista ne kentät, joita syöttäjien ei kuulu nähdä.

- Syötteet:

**DBControl** conn

**UserQueryContainer** searchfor

- Palauttaa:

**Users**[ ]

**getUserFull** suorittaa varsinaiset käyttäjien haut. Lisäksi ylläpitäjä-tason käyttäjät pystyvät kutsumaan metodia suoraan, jolloin palautettavat tiedot sisältävät enemmän informaatiota. Ensimmäiseksi tarkistetaan, mitkä kaikki kentät annetussa **User**-objektissa sisältävät tietoa, ja näiden tietojen perusteella muodostetaan annetuista ehdoista SQL-kysely. Hakuehtoihin sopivat tulokset laitetaan **User**-taulukkaan, ja se palautetaan kutsujalle.

Metodi itse ei kontrolloi käyttöään millään tavalla, vaan se on jätetty ylempään tason asiaksi. Mikäli käsittelyssä tapahtuu jokin virhe, palauttaa metodi pelkästään null, ja kirjoittaa tiedot virheestä sisäiseen virhemuuttujaan. Myös metodia käyttävän, normaalin **getUsers**-metodin tulee varautua tähän.

- Syötteet:

**DBControl** conn

**UserQueryContainer** searchfor

- Palauttaa:

**Users**[ ]

**removeUser** mahdollistaa käyttäjien poistamisen järjestelmästä. Aluksi metodi tarkistaa, onko parameterinä annettu käyttäjännumero tehnyt järjestelmään yhtään havaintoa. Mikäli yhtään havaintoa ei löydy, poistetaan käyttäjän tiedot järjestelmästä kokonaan. Jos tehtyjä havaintoja löytyy, merkitään käyttäjä poistetuksi muuttamalla käyttäjätasoksi **TahitiLibrary**stä löytyvän **USER\_DELETED** vakion osoittama arvo. Käyttäjää poistettaessa sekä lokitaulu että havaintotaulu täytyy käydä läpi käyttäjää koskevien merkintöjen varalta. Tieto käyttäjän poistamisesta kirjoitetaan lokiin normaalin lokikirjoitusrutiinin mukaisesti.

Mikäli poistaminen onnistuu (eli käyttäjä löytyi, ja poistettiin), palautetaan true. Virhetilanteessa toimitaan kuten muissakin metodeissa, eli tieto virheestä kirjoitetaan sisäiseen virhemuuttujaan ja palautetaan false. Metodin antama palaute ei kerro poistettiinkö käyttäjä kokonaan, vai merkittiinkö vain poistetuksi.

- Syötteet:

**DBControl** conn

**User** userID

**User** deleter

- Palauttaa:

**boolean**

**setUserData** tarjoaa käyttäjätietojen muuttamiseen tarvittavan palvelun. Aluksi tarkistetaan, onko kyseessä käyttäjän itsensä tekemä muutos, vai ylläpidon. Jos muutos on käyttäjän itsensä tekemä ja muuntaja on tasoltaan syöttäjä, ei kaikkien kenttien muuttaminen ole mahdollista. Salasanan vaihtoa ei tämän metodin kautta sallita laisinkaan.

Mikäli kyseessä on ylläpitäjän tekemä muutos käyttäjän tietoihin, tarkistetaan koskeeko muutos käyttäjän tasoa, ja jos koskee, tapahtumasta tarvitaan myös merkintä järjestelmän lokiin. Lisäksi metodin pitää ennen muutosten tekemistä tarkistaa, että kaikki pakolliset käyttäjätiedot ovat mukana uudessa käyttäjäobjektissa. Annettu **UserData** tulkitaan siten, että jokainen tietokantaan päätyvä arvo otetaan suoraan tästä annetusta objektista, jolloin mahdolliset tyhjät kentät saattavat poistaa tietokannasta siellä jo olevaa informaatiota.

Virhetilanteissa toimiminen tapahtuu samoin, kuten kaikissa muissakin metodeissa, eli sisäisen virhetietomuuttujan avulla. Mikäli päivittäminen onnistuu ongelmitta, palautetaan **true**.

- Syötteet:

**DBControl** conn

**UserFormContainer** newData

**User** changer

- Palauttaa:

**boolean**

**resetPassword** mahdollistaa unohtettujen salasanojen nollaamisen. Aluksi metodi tarkistaa, että annetulla **loginID**:llä oleva käyttäjä löytyy järjestelmästä. Tämän jälkeen muodostetaan pseudo-satunnainen salasana, joka salataan, ja jolla ylikirjoitetaan käyttäjän senhetkinen salasana. Tämän jälkeen uusi salasana palautetaan ylemmälle tasolle salaamattomana, jotta se voidaan lähettää sähköpostilla käyttäjälle. Mikäli käsittelyssä tapahtuu jonkinlainen virhe, palautetaan **null**-viite, ja tiedot tapahtuneesta virheestä kirjoitetaan sisäiseen virhemuuttajaan.

- Syötteet:

**DBControl** conn

**String** loginID

- Palauttaa:

**String**

**getUser** tarjoaa mahdollisuuden käyttäjätietojen hakemiseen sekä kirjautumistunnuksen että sisäisen käyttäjänumeron perusteella. Mikäli metodia on kutsuttu kirjautumistunnuksella, haetaan aluksi annettuun tunnukseen liittyvä sisäinen tunnus, ja kutsutaan tämän jälkeen metodia saadulla käyttäjänumerolla.

Metodi palauttaa annettuihin tietojen liittyvän **User**-objektin, mikäli sellainen löytyy. Virhetilanteessa palautetaan null, ja kirjoitetaan tieto tapahtuneesta virheestä sisäiseen virhemuuttujaan.

- Syötteet:

1. **DBControl** conn

**String** loginID

2. **DBControl** conn

**int** userID

- Palauttaa:

**User**

## 6 Tietokanta

Tahiti-projektin tietokannaksi on valittu PostgreSQL-tietokantaohjelmisto. Tietokantaan talletetaan kaikki järjestelmään talletettava tieto. Tässä luvussa kuvataan tietokanta yksityiskohtaisesti esittelemällä tietokannan numerosarjat, käyttäjät, taulut, näkymät ja funktiot

### 6.1 Numerosarjat

Tahiti-tietokanta käyttää PostgreSQL-tietokantaohjelmiston tarjoamaa numerosarjaa (serial) muuttujatyypinä yksikäsitteisten kannan sisäisten numerointien tekemiseen. Kannassa yksikäsitteinen numero annetaan käyttäjille, havainnoille, asteroideille ja lokimerkinnöille.

```
CREATE SEQUENCE TahitiUser_UserNumber
CREATE SEQUENCE Asteroid_AsteroidNumber
CREATE SEQUENCE Observation_ObservationNumber
CREATE SEQUENCE LogEntry_EntryNumber
```

Kun kantaan halutaan tallettaa uusia käyttäjätietoja, havaintoja, asteroideja tai lokimerkintöjä, haetaan kannasta tallennuksessa käytettävä sarjanumero. Tätä numeroa käytetään koko tallennuksen ajan. Jos siis esimerkiksi halutaan tallettaa uusi valokäyrä, samaa kannan antamaa sarjanumeroa käytetään sekä taulun **Observation** että taulun **Lightcurve** uuden alkion luomisessa. Sarjanumero saadaan kannasta suorittamalla kantaan kysely

```
SELECT nextval('<sequence>');
```

Kyselyssä <sequence> on mikä tahansa kannan neljästä numerosarjasta. Tuloksena kanta antaa sarjanumeron yksisarakkeisessa taulussa sarakkeessa nextval. Numerosarjassa on 2147483647 yksikäsitteistä numeroa.

## 6.2 Käyttäjät

Tahiti-tietokantaan luodaan kolme käyttäjää vastaamaan järjestelmän eri käyttäjäryhmiä. Näitä käyttäjiä käytetään eritasoisten yhteyksien luomisessa tietokantaan. Tietokantaan luodut käyttäjät ovat TahitiBrowser, TahitiSubmitter sekä TahitiAdmin. TahitiBrowser-käyttäjälle annetaan ainoastaan oikeus selata (select) tauluja Asteroid, Observation, Trajectory, Lightcurve, Datapoint ja numerosarjat LogEntry\_EntryNumber ja TahitiUser\_UserNumber sekä oikeus lisätä (insert) monikoita tauluihin User sekä LogEntry. TahitiSubmitter-käyttäjä saa lisäksi selausoikeuden tauluun User sekä järjestelmän numerosarjoihin, päivitys- (update) ja lisäysoikeudet tauluihin User, Observation, Lightcurve ja DataPoint. TahitiAdmin-käyttäjä saa selaus, lisäys, päivitys ja poisto (delete) -oikeudet kaikkiin Tahiti-tietokannan tauluihin ja numerosarjoihin.

```
CREATE USER TahitiBrowser WITH PASSWORD 'tahiti';
GRANT select ON Asteroid, Observation, Trajectory, Lightcurve, DataPoint,
      LogEntry_EntryNumber, TahitiUser_UserNumber, LightcurveOverview,
      AsteroidOverview
      TO TahitiBrowser;
GRANT insert ON TahitiUser, LogEntry
      TO TahitiBrowser;
```

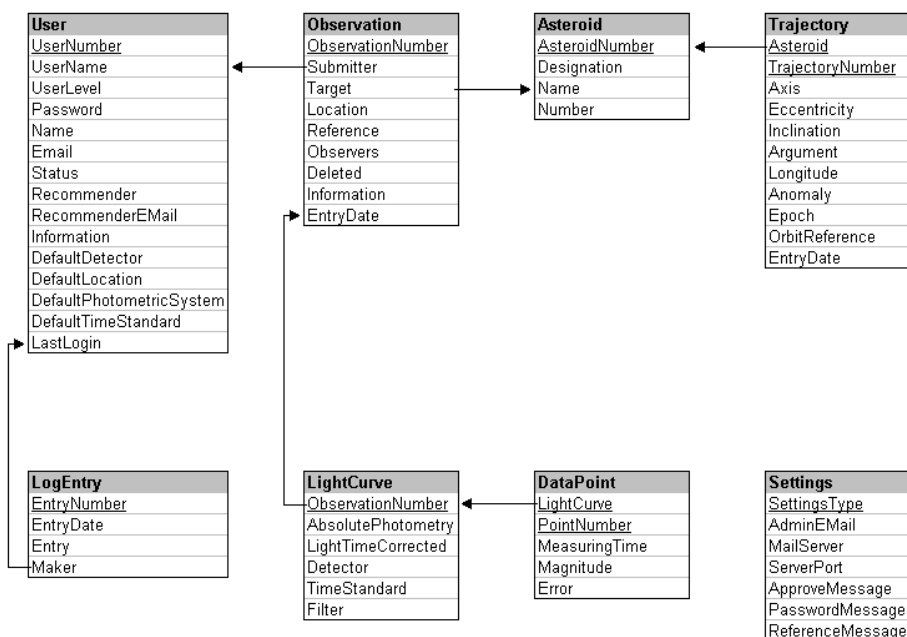
```
CREATE USER TahitiSubmitter WITH PASSWORD 'tahiti';
GRANT select ON Asteroid, Observation, Trajectory, Lightcurve, DataPoint,
      TahitiUser, LogEntry_EntryNumber, TahitiUser_UserNumber,
      Observation_ObservationNumber, LightcurveOverview,
      AsteroidOverview
      TO TahitiSubmitter;
GRANT insert ON TahitiUser, LogEntry, Observation, Lightcurve, DataPoint
      TO TahitiSubmitter;
GRANT update ON TahitiUser, Observation, Lightcurve, DataPoint
      TO TahitiSubmitter;
```

```
CREATE USER TahitiAdmin WITH PASSWORD 'tahiti';
GRANT select ON Asteroid, Observation, Trajectory, Lightcurve, DataPoint,
      TahitiUser, LogEntry, Settings, LogEntry_EntryNumber,
      TahitiUser_UserNumber, Observation_ObservationNumber,
      Asteroid_AsteroidNumber, LightcurveOverview, AsteroidOverview
      TO TahitiAdmin;
GRANT insert, update, delete ON TahitiUser, LogEntry, Observation, Lightcurve,
      DataPoint, Asteroid, Trajectory, Settings
```

TO TahitiAdmin;

## 6.3 Taulut

Tässä luvussa kuvataan tietokannassa olevat taulut ensin kuvana (Kuva 12), josta selviävät taulujen nimet sekä niiden pää- ja viiteavaimet. Tämän jälkeen jokaisesta taulusta on oma alilukunsa, jossa esitellään taulumäärittelylause ja sanallinen kuvaus taulusta. Kuvassa nuoli tarkoittaa viiteavainta sen alkupäässä olevasta attribuutista nuolen osoittaman taulun pääavaimeen.



Kuva 12: Tietokanta

### 6.3.1 TahitiUser

Taulu TahitiUser sisältää kaikki järjestelmän syöttäjät, ylläpitäjät ja rekisteröintipyyntöä. Taulun monikkojen identifioimiseen käytetään numerosarjan TahitiUser\_UserNumber tuottamia sarjanumeroita. Jos tauluun lisättäessä ei annetta sarakkeelle UserNumber lainkaan arvoa, järjestelmä asettaa sarakkeen arvoksi oletusasetuksena numerosarjan TahitiUser\_UserNumber seuraavan arvon. Oikeuksissa on kolme tasoa. Taso 0 merkitsee rekisteröintipyyntöä, taso 1 syöttäjää ja taso 2 ylläpitäjää. Taso -1 merkitsee, että ylläpitäjä on poistanut käyttäjän kannasta. Tason oletusarvo on 0. Mikäli luokan TahitiLibrary vakioarvoihin tehdään muutoksia, oikeustasotkin täytyy päivittää kantaan. Tauluun luodaan sen luontivaiheessa automaattisesti ylläpitäjätasoinen käyttäjä Tahiti.

```
CREATE TABLE TahitiUser (
```



```

UserNumber          INT4          DEFAULT nextval('User_UserNumber'),
UserName            TEXT          NOT NULL,
UserLevel           INT2          DEFAULT 0 CHECK (Userlevel in (-1,0,1,2)),
Password TEXT,
Name                TEXT          NOT NULL,
EMail               TEXT          NOT NULL,
Status              TEXT          NOT NULL,
Recommender         TEXT          NOT NULL,
RecommenderEMail   TEXT          NOT NULL,
Information          TEXT,
DefaultDetector     TEXT,
DefaultLocation     TEXT,
DefaultPhotometricSystem TEXT,
DefaultTimeStandard TEXT,
LastLogin            TIMESTAMP,
Comment             TEXT,
Version INT4,
CONSTRAINT pk_TahitiUser_UserNumber PRIMARY KEY (UserNumber));

```

### 6.3.2 Asteroid

Asteroid-tauluun talletetaan asteroidin nimi, tunnus ja numero. Taulun monikkojen identifioimiseen käytetään numerosarjan Asteroid\_AsteroidNumber tuottamia sarjanumeroita. Jos tauluun lisättäessä ei annetta sarakkeelle AsteroidNumber lainkaan arvoa, järjestelmä asettaa sarakkeen arvoksi oletusasetuksena numerosarjan Asteroid\_AsteroidNumber seuraavan arvon. Maa on talletettu järjestelmään asteroidina, jonka AsteroidNumber on 0, nimi on Earth, Designation on Earth.

```

CREATE TABLE Asteroid (
AsteroidNumber      INT4          DEFAULT nextval('Asteroid_AsteroidNumber'),
Designation         TEXT,
Name                TEXT,
Number              INT4,
CONSTRAINT pk_Asteroid_AsteroidNumber PRIMARY KEY (AsteroidNumber));

```

### 6.3.3 Observation

Taulu Observation sisältää kaikki asteroideista tehdyt havainnot. Taulun monikkojen identifioimiseen käytetään numerosarjan Observation\_ObservationNumber tuottamia sarjanumeroita. Jos tauluun lisättäessä ei annetta sarakkeelle ObservationNumber lainkaan arvoa, järjestelmä asettaa sarakkeen arvoksi oletusasetuksena numerosarjan Observation\_ObservationNumber seuraavan arvon. Deleted-sarakkeen oletusarvo on false ja EntryDate sarakkeen lisäyksen ajankohta. Taulussa on kaksi viiteavainta. Sarake Submitter viittaa tauluun TahitiUser ja kertoo havainnon tekijän. Sarake Target puolestaan viittaa tauluun Asteroid ja kertoo havainnon kohteen. Havainnon kohteen poiston yhteydessä myös havainto poistetaan.

```

CREATE TABLE Observation (
  ObservationNumber      INT4              DEFAULT nextval('Observation_ObservationNumber'),
  Deleted                BOOLEAN           DEFAULT FALSE,
  Information            TEXT,
  Location               TEXT              NOT NULL,
  Observers              TEXT              NOT NULL,
  Reference              TEXT,
  Submitter              INT4,
  Target                 INT4              NOT NULL,
  Version                INT4,
  EntryDate              TIMESTAMP         DEFAULT now(),
  CONSTRAINT pk_Observation_ObservationNumber PRIMARY KEY (ObservationNumber),
  CONSTRAINT fk_Observation_Submitter FOREIGN KEY (Submitter)
      REFERENCES TahitiUser
      ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT fk_Observation_Target FOREIGN KEY (Target)
      REFERENCES Asteroid ON DELETE CASCADE ON UPDATE CASCADE);

```

### 6.3.4 Lightcurve

Taulun Lightcurve monikot laajentavat Observation taulun havaintoja, ja antavat havainnolle valokäyrälle ominaiset attribuutit. Jos valokäyrän yleistykseenä oleva havainto poistetaan, poistuu myös valokäyrä. Suodattimien (Filter) arvot tarkistetaan kuuluvaksi TahitiLibrary-luokassa määriteltyjen vakioiden joukkoon. Jos TahitiLibrary-luokan vakioita muutetaan, tähänkin tarkistukseen tulee tehdä muutos. Lisäksi valokäyrään talletetaan redundanttia tietoa asteroidin sijainnista havaintohetkellä. Sijaintitietoja ovat sarakkeet PhaseAngle (vaihekulma), DistanceFromEarth (etäisyys maasta), DistanceFromSun (etäisyys auringosta), Longitude (ekliptikaalinen pituus) ja Latitude (ekliptikaalinen leveys). Jos Valokäyrään liittyvälle asteroidille talletetaan järjestelmään uudet rataelementit, täytyy järjestelmän tarkistaa vaihtuivatko oletusrataelementtien arvot. Jos oletusrataelementit muuttuvat täytyy redundantit sarakkeet laskea uudelleen.

```

CREATE TABLE LightCurve (
  ObservationNumber      INT4              NOT NULL,
  AbsolutePhotometry     BOOLEAN           NOT NULL,
  LightTimeCorrected     BOOLEAN           NOT NULL,
  PhotometricSystem      TEXT,
  Detector                TEXT,
  TimeStandard            TEXT,
  DefaultTrajectory      INT4,
  PhaseAngle              FLOAT4,
  DistanceFromEarth       FLOAT4,
  DistanceFromSun         FLOAT4,
  Longitude                FLOAT4,
  Latitude                 FLOAT4,

```

```

Filter          INT4          NOT NULL CHECK (Filter in (2, 3, 4, 5, 6, 7)),
Version,
CONSTRAINT pk_LightCurve_ObservationNumber PRIMARY KEY (ObservationNumber),
CONSTRAINT fk_LightCurve_ObservationNumber FOREIGN KEY (ObservationNumber)
REFERENCES Observation ON DELETE CASCADE ON UPDATE CASCADE);

```

### 6.3.5 DataPoint

Valokäyrän (Lightcurve) muodostavat havaintopisteet talletetaan tauluun DataPoint. Piste identifoidaan siihen liittyvän valokäyrän sekä havaintopistenumeron (PointNumber) avulla. Jos pisteeseen liittyvä valokäyrä tuhoetaan, myös piste tuhoutuu. Lisäksi havaintopisteen yhteyteen talletetaan redundanttia tietoa asteroidin ja maan sijainnista havaintohetkellä. Redundanttia tietoa ovat sarakkeet AsteroidX, AsteroidY, AsteroidZ, EarthX, EarthY ja EarthZ. Sarakkeet ilmaisevat asteroidin ja maan koordinaatit auringosta katsottuna havaintohetkellä. Jos valokäyrään, johon havaintopiste kuuluu, liittyvälle asteroidille talletetaan järjestelmään uudet rataelementit, täytyy järjestelmän tarkistaa vaihtuivatko oletusrataelementtien arvot. Jos oletusrataelementit muuttuvat täytyy redundantit sarakkeet laskea uudelleen.

```

CREATE TABLE DataPoint (
LightCurve          INT4          NOT NULL,
PointNumber         INT4          NOT NULL,
MeasuringTime       TIMESTAMP     NOT NULL,
Magnitude           FLOAT4        NOT NULL,
Error               FLOAT4,
EarthX FLOAT4,
EarthY FLOAT4,
EarthZ FLOAT4,
AsteroidX FLOAT4,
AsteroidY FLOAT4,
AsteroidZ FLOAT4,
Version INT4,
CONSTRAINT pk_DataPoint_LightCurve_PointNumber PRIMARY KEY (LightCurve, PointNumber),
CONSTRAINT fk_DataPoint_LightCurve FOREIGN KEY (LightCurve)
REFERENCES LightCurve ON DELETE CASCADE ON UPDATE CASCADE);

```

### 6.3.6 Trajectory

Tauluun Trajectory talletetaan asteroidien (Asteroid) rataelementit. Rataelementti identifoidaan siihen liittyvän asteroidin ja rataelementtinumeron (TrajectoryNumber) avulla. Taulun monikon lisäyspäiväksi (EntryDate) asetetaan oletusarvoisesti lisäyksen ajanhetki.

```

CREATE TABLE Trajectory (
Asteroid          INT4          NOT NULL,
TrajectoryNumber  INT4          NOT NULL,

```

```

Axis          FLOAT4          NOT NULL,
Eccentricity  FLOAT4          NOT NULL,
Inclination   FLOAT4          NOT NULL,
Argument      FLOAT4          NOT NULL,
Longitude     FLOAT4          NOT NULL,
Anomaly       FLOAT4          NOT NULL,
Epoch        TIMESTAMP    NOT NULL,
OrbitReference TEXT,
EntryDate     TIMESTAMP    DEFAULT now(),
CONSTRAINT pk_Trajectory_Asteroid_TrajectoryNumber PRIMARY KEY (Asteroid, TrajectoryNumber),
CONSTRAINT fk_Trajectory_Asteroid FOREIGN KEY (Asteroid)
REFERENCES Asteroid ON DELETE CASCADE ON UPDATE CASCADE);

```

### 6.3.7 LogEntry

Tauluun LogEntry talletetaan järjestelmän tapahtumaloki. Taulun monikkojen identifiointiin käytetään numerosarjan LogEntry\_EntryNumber tuottamia sarjanumeroita. Jos tauluun lisättäessä ei anneta sarakkeelle EntryNumber lainkaan arvoa, järjestelmä asettaa sarakkeen arvoksi oletusasetuksena numerosarjan LogEntry\_EntryNumber seuraavan arvon. Taulun monikon lisäyspäiväksi (EntryDate) asetetaan oletusarvoisesti lisäyksen ajanhetki.

```

CREATE TABLE LogEntry (
EntryNumber      INT4          DEFAULT nextval('LogEntry_EntryNumber'),
EntryDate        TIMESTAMP    DEFAULT now(),
Entry            TEXT          NOT NULL,
Maker            INT4,
CONSTRAINT pk_LogEntry_EntryNumber PRIMARY KEY (EntryNumber),
CONSTRAINT fk_LogEntry_Maker FOREIGN KEY (Maker)
REFERENCES TahitiUser ON DELETE CASCADE ON UPDATE CASCADE);

```

### 6.3.8 Settings

Settings-tauluun talletetaan järjestelmän asetukset. Tauluun voidaan periaatteessa tallettaa useita eri asetuksia. Asetusten identifiointiin käytetään saraketta SettingsType. Sarakkeen SettingsType oletusarvo on Default.

```

CREATE TABLE Settings (
SettingsType     TEXT          DEFAULT 'Default',
AdminEMail       TEXT          NOT NULL,
MailServer       TEXT          NOT NULL,
ServerPort       INT2          NOT NULL,
ApproveMessage   TEXT,
ApproveMessageHeader TEXT,
PasswordMessage  TEXT,

```

```

PasswordMessageHeader TEXT,
ReferenceMessage      TEXT,
ReferenceMessageHeader TEXT,
CONSTRAINT pk_Settings_SettingsType PRIMARY KEY (SettingsType));

```

## 6.4 Näkymät

Tietokannan päälle rakennettavan rajapinnan avuksi tietokantaan luodaan näkymä asteroidien tarkasteluun sekä valokäyrien tarkasteluun. Nämä kaksi näkymää ovat tarpeen, koska sekä asteroideja että valokäyriä tarkastellessa joudutaan tekemään yhteenvetoja niihin liittyvistä tiedoista. Näkymien muodostamiseen on käytetty luvussa 6.5 esiteltyjä funktioita.

### 6.4.1 AsteroidOverview

Näkymä AsteroidOverview on tarkoitettu hauille, joiden hakuehdoissa ei ole valokäyriin liittyviä hakuehtoja. Toisin sanottuna hauille, joissa hakuehdot ovat yhdistelmiä seuraavista: asteroidin nimi, numero tai tunnus, valokäyrien määrä ja havainnoitsijoiden määrä. Näkymä on taulu, josta käyvät ilmi asteroidin järjestelmän sisäinen tunnus (AsteroidID), asteroidin numero (Number), asteroidin nimi (Name), Asteroidin tunnus (Designation), asteroidista tehtyjen valokäyrähavaintojen määrä (NumberOfLightcurves), eri havainnoitsijoiden lukumäärä (NumberOfObservers) ja viimeisimmän asteroidista tehdyn havainnon päivämäärä (LastObservation). Tätä näkymää voidaan käyttää yllä mainituissa hauissa näkymän LightcurveOverview sijaan, jos järjestelmän toimintaa halutaan nopeuttaa.

```

CREATE VIEW AsteroidOverview (AsteroidID, Number, Name, Designation,
                              NumberOfLightcurves, NumberOfObservers,
                              LastObservation)
AS SELECT AsteroidNumber, Number, Name, Designation,
          count(distinct ObservationNumber), count(distinct Observers),
          max(MeasuringTime)
FROM Asteroid, Observation, Datapoint
WHERE AsteroidNumber = Target and
      ObservationNumber = LightCurve
GROUP BY AsteroidNumber, Number, Name, Designation;

```

### 6.4.2 LightcurveOverview

Tätä näkymää käytetään haettaessa valokäyrä- tai asteroiditietoja kannasta. Siihen on koottu kaikki valokäyrään liittyvät tiedot.

```

CREATE VIEW LightcurveOverview (AsteroidID, AsteroidName, AsteroidNumber,
AsteroidDesignation, LightCurveID,
ObservationTime, DefaultTrajectoryID,
EarthTrajectoryID, EntryDate, Location,

```

```

Reference, Observers, Information,
Deleted, AbsolutePhotometry,
LightTimeCorrected, PhotometricSystem,
Detector, TimeStandard, Filter,
Submitter, NumberOfPoints,
                                PhaseAngle, DistanceFromSun,
DistanceFromEarth, Longitude, Latitude,
Version)
AS
SELECT AsteroidNumber, Asteroid.Name, Asteroid.Number,
       Asteroid.Designation, Observation.ObservationNumber,
       observation_time(Observation.ObservationNumber),
       DefaultTrajectory, DefaultEarthTrajectory,
       EntryDate, Location, Reference,
       Observers, Information, Deleted, AbsolutePhotometry,
       LightTimeCorrected, PhotometricSystem, Detector, TimeStandard,
       Filter, Submitter, number_of_datapoints(Observation.ObservationNumber),
       PhaseAngle, DistanceFromSun, DistanceFromEarth, Longitude, Latitude,
       Observation.Version
FROM Asteroid, Observation, Lightcurve
WHERE Target = AsteroidNumber and
       Observation.ObservationNumber = Lightcurve.ObservationNumber;

```

### 6.4.3 LightcurveUpdate

Näkymä listaa niiden valokäyrien järjestelmän sisäiset numerot, joiden redundantit sijaintitiedot eivät ole ajan tasalla.

```

CREATE VIEW LightcurveUpdate (LightcurveNumber)
AS
SELECT ObservationNumber
FROM Lightcurve, Observation
WHERE Lightcurve.ObservationNumber = Observation.ObservationNumber and
      (DefaultTrajectory !=
       default_trajectory(Target, observation_time(ObservationNumber)) or
       DefaultEarthTrajectory !=
       default_trajectory(0, observation_time(ObservationNumber)))

```

## 6.5 Funktiot

Tietokanta tarjoaa myös funktioita tiedon hakemiseksi kannasta. Funktioita voi käyttää minkä tahansa kyselyn apuna tai hakemalla niiden avulla yksittäistä tietoa. Määriteltyjä funktioita käytetään näkymien muodostukseen.

### 6.5.1 default\_trajectory

Funktio ottaa syötteekseen asteroidin järjestelmän sisäisen numeron ja ajanhetken ja palauttaa asteroidin ensimmäisen ennen annettua ajanhetkeä voimassa olevan rataelementtimonikon järjestelmän sisäisen tunnusnumeron.

```
CREATE FUNCTION default_trajectory(INT4, TIMESTAMP) RETURNS INT4
AS '
SELECT TrajectoryNumber
FROM Trajectory
WHERE Asteroid = $1 and
      Epoch <= $2 and
      Epoch >= ALL (SELECT Epoch FROM Trajectory WHERE Asteroid = $1 and Epoch <= $2);
' LANGUAGE 'SQL';
```

### 6.5.2 observation\_time

Funktio ottaa syötteekseen valokäyrän järjestelmän sisäisen numeron ja palauttaa valokäyrän ensimmäisen havainnon havaintohetken.

```
CREATE FUNCTION observation_time(INT4) RETURNS TIMESTAMP
AS '
SELECT min(MeasuringTime)
FROM DataPoint
WHERE Lightcurve = $1;
' LANGUAGE 'SQL';
```

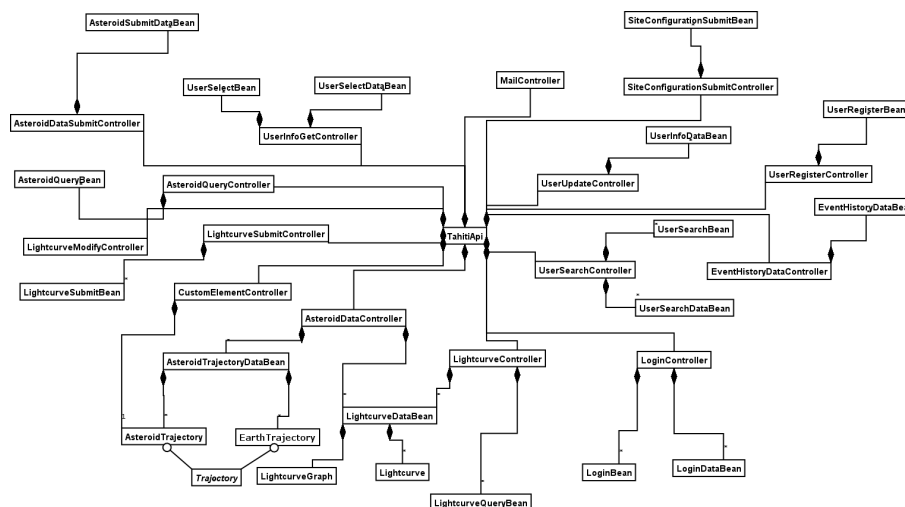
### 6.5.3 number\_of\_datapoints

Funktio ottaa syötteekseen valokäyrän järjestelmän sisäisen numeron ja palauttaa valokäyrässä olevien havaintopisteiden lukumäärän.

```
CREATE FUNCTION number_of_datapoints(INT4) RETURNS INT4
AS '
SELECT count(*)
FROM DataPoint
WHERE Lightcurve = $1;
' LANGUAGE 'SQL';
```

## 7 Käyttöliittymäluokat

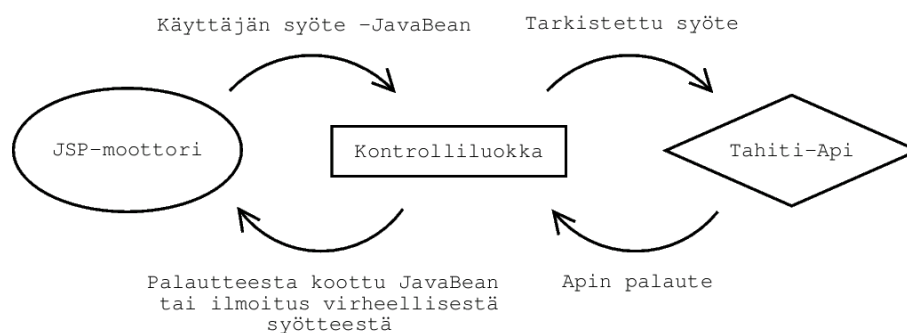
Alla on kuvattuna kaikki käyttöliittymässä käytettävät luokat, joista koottu luokkakaavio esitetään kuvassa 13. Käyttöliittymäluokat ovat jaettu kolmeen eri osaan. Tieto eri komponenttien välillä siirretään JavaBeaneissa, joita myös käytetään www-lomakkeiden tietojen tarkistamiseen.



Kuva 13: Käyttöliittymäluokkien luokkakaavio

Käyttöliittymän kontrollointiin ja tiedon esikäsittelyyn käytetään erillisiä kontrolliluokkia, jotka ovat Javan Servlet-luokan ilmentymiä. Jokaista käyttöliittymässä tapahtuvaa tiedonkäsittelytoimintoa varten on oma kontrolliluokkansa. Käyttöliittymän kolmas osa on erillinen JSP-sivutaso, joka esittää ruudulle tietokannassa olevan tiedon, ja näin toimii käyttäjälle näkyvänä konkreettisenä käyttöliittymänä.

Käyttöliittymässä on pyritty käyttämään Model-Control-View –suunnittelumallia, joka havainnollistetaan kuvassa 14. Modelina toimii tietokannassa oleva tieto ja Control-osana toimivat sekä esikäsittelijä-servletit toimintoille että varsinaiset apikutsut. Viewiä kuvaavat erilliset järjestelmän JSP-sivut.



Kuva 14: Käyttöliittymän toiminta MCV-mallin mukaan

## 7.1 Kontrolliluokat

Järjestelmän jokaiselle toiminnolle tai toimintoryhmälle on oma kontrolli- eli esikäsittelijäluokkansa, jota kutsutaan myös servlet-luokaksi. Kontrolliluokkien tarkoituksena on tarkistaa käyttäjän tekemän palvelupyynnön parametrien oikeellisuus, jotta mahdollisesta virheparametrasta voidaan tehdä hyvä virheilmoitus. Jos palvelupyynnö läpäisee tarkastuksen, se annetaan edelleen eteenpäin



Tahiti-Apille, joka toteuttaa varsinaisen toiminnon, ja palauttaa vastauksen servletin tekemälle kutsulle. Kutsun paluuarvona tullut data muunnetaan JavaBeaniksi, jota hyödynnetään JSP-sivulla käyttäjälle annettaessa palautetta hänen haluamastaan toiminnosta. Kontrolliluokat myös ohjaavat käyttäjän eri kohtaan käyttöliittymässä saadusta palautteesta riippuen.

### 7.1.1 AsteroidQueryController

AsteroidQueryController-luokan tarkoituksena on tarkastaa *AsteroidQueryBean*-luokan avulla käyttäjän antaman asteroidihakuehdon tiedot ja lähettää kysely eteenpäin järjestelmälle Tahiti-Apin *getAsteroids*-metodin välityksellä. Api-kutsun palautettua tiedot luokalle luokka ohjaa käyttäjän takaisin asteroidihakusivulle, jolla hakutulokset näytetään. Jos käyttäjän antamat hakuehdot olivat virheelliset, luokka ohjaa käyttäjän takaisin asteroidihakusivulle ja palauttaa virheilmoituksen JSP-sivun käsiteltäväksi.

### 7.1.2 AsteroidDataController

AsteroidDataController-luokan tarkoituksena on hakea käyttäjän valitseman asteroidin tiedot Tahiti-Apin *getLightcurves*- ja *getTrajectory*-metodien välityksellä. Asteroidi valitaan painamalla asteroidin nimeä asteroidihakutaulussa, ja toiminto siirtää käyttäjän asteroidin-tiedot sivulle, jolla hän voi tarkastella tarkemmin kyseisen asteroidin havainto- sekä ratatietoja. Tiedot paketoidaan kahteen erilliseen luokkaan: *LightcurveDataBean*, *EarthTrajectoryDataBean* ja *AsteroidTrajectoryDataBean*.

### 7.1.3 LightcurveController

LightcurveController saa 'Asteroidin tiedot' -sivulla sijaitsevan valokäyrähakutulostaulun yhteydessä olevan lomakkeen tiedot, jossa määritellään minkä toiminnon käyttäjä valitsi (näytä data sivulla, luo tiedoista raakadatatieosto tai luo tiedoista muunnettu datatieosto), mitkä valokäyrät käyttäjä haluaa toimintoon mukaan, mitkä rataelementit käyttäjä haluaa valituille valokäyrille sekä missä muodossa haluttu havaintotieto esitetään. Apuna käytetään Tahiti-Apin *makeDataFile*- ja *makeRawData*-metodeja. Kaikki tämä tieto talletetaan *LightcurveQueryBean*-luokan ilmentymään. Toiminnosta riippuen luokka ohjaa käyttäjän asteroidin tiedot -sivulle, raakadatatieoston näyttösivulle tai muunnetun datan näyttösivulle. Jos käyttäjän valitsema toiminto on tiedon näyttäminen sivulla, luo LightcurveController jokaista sivulla näytettävää valokäyrää varten LightcurveGraph-luokan ilmentymän, jossa muodostetaan kuva valokäyrän datapisteistä ajan funktiona. Tämä luokka tallennetaan LightcurveDataBeanin yhteyteen.

### 7.1.4 CustomElementController

CustomElementController-luokan tehtävä on ohjata käyttäjä sivulle, jolla hän voi antaa omia ratatietoja haluamalleen valokäyrälle. Kun käyttäjän on antanut valokäyrätiedot, CustomElementController liittää uudet käyttäjän antamat valokäyrätiedot oikean valokäyrän yhteyteen, ja ohjaa käyttäjän takaisin asteroidin tiedot -sivulle.

### 7.1.5 LightcurveSubmitController

LightcurveSubmitController-luokka ottaa vastaan käyttäjän antaman uuden valokäyrädatan ja tekee tarkistuksen *LightcurveSubmitDataBean*-luokan avulla. Tiedon ollessa oikeellista luokka kutsuu Tahiti-Apin insertLightCurve-metodia. Jos tiedoista löytyy virheitä, käyttäjä palautetaan valokäyrän syöttösivulle jolla virheestä ilmoitetaan käyttäjälle.

### 7.1.6 UserInfoUpdateController

UserInfoUpdateController-luokka päivittää käyttäjän antamia tietoja itsestään. Luokka tarkastaa käyttäjän antamat arvot *UserInfoDataBean*-luokan avulla. Jos käyttäjän antamat tiedot olivat oikeellisia, luokka kutsuu Tahiti-Apin käyttäjätietojen setUserData-metodia ja siirtää käyttäjän takaisin tietojen päivityssivulle metodikutsusta palattuaan. Tietojen ollessa virheellisiä käyttäjä palautetaan tietojen päivityssivulle jolla virheistä ilmoitetaan. Luokan avulla myös päivitetään kenen tahansa käyttäjätietoja ylläpitäjän niitä päivittäessä.

### 7.1.7 UserSearchController

UserSearchController-luokka saa käyttäjältä osittaisen nimen, jonka perusteella järjestelmästä haetaan muiden käyttäjien tietoja. Jos nimi on oikeellinen, järjestelmä kutsuu Tahiti-Apin käyttäjienhaku-metodia, joka palauttaa listan käyttäjistä. Tämän jälkeen käyttäjä palautetaan käyttäjien etsintä-sivulle, jossa löydettyjen käyttäjien tiedot näytetään. Käyttäjän antaessa virheellisen nimitiedon hänet ohjataan takaisin käyttäjien etsintä-sivulle, jolla virhe ilmoitetaan. UserSearchBean välitetään Tahiti-Apille, jonka palauttamasta datasta kootaan UserSearchDataBean, jonka UserSearchController välittää JSP-sivuille.

### 7.1.8 UserInfoGetController

UserInfoGetController-luokka hakee käyttäjän valitseman toisen käyttäjän tiedot järjestelmästä, sekä siirtää käyttäjän sivulle, jolla nämä tiedot näytetään. Tiedot välitetään UserSelectBeanissa Tahiti-Apille, ja Tahiti-Apin getUsers-metodin palauttamista käyttäjätiedoista kootaan UserDataBean, joka välitetään JSP-sivuille.

### 7.1.9 LightcurveModifyController

LightcurveModifyController-luokalla on kaksi erillistä toimintoa. Luokka sekä merkkää valokäyriä poistettavaksi, että ohjaa ylläpitäjän oikealle valokäyräsyöttösivulle hänen halutessaan muuttaa yksittäisen valokäyrän tietoja. Tahiti-Apin deleteLightcurve- ja restoreLightcurve -metodeja käytetään tässä apuna.

### 7.1.10 AsteroidDataSubmitController

AsteroidDataSubmitController-luokka tarkastaa ylläpitäjän antaman asteroidiratatietodatan *AsteroidSubmitDataBean*-luokan avulla, ja antaa tiedon eteenpäin Tahiti-Apin changeLightcurve-

metodille sen ollessa oikeellista. Tahiti–Api–kutsusta palattuaan ylläpitäjä ohjataan takaisin asteroidihakusivulle.

#### 7.1.11 MailController

Ylläpitäjän halutessa lähettää postia rekisteröityneelle käyttäjälle tai suositteijalle tämä luokka käynnistää Tahiti–Apin kulloinkin parhaiten tehtävään soveltuvan metodin (spamUser tai spamRecommender), joka sitten lähettää kyseiselle henkilölle postia. Tahiti–Apille annetaan parametrimina sähköpostin vastaanottajan sähköpostiosoite.

#### 7.1.12 SiteConfigurationSubmitController

Luokan tarkoituksena on päivittää ylläpitäjän muuttamia järjestelmäasetuksia. Luokka tarkistaa arvojen oikeellisuuden *SiteConfigurationSubmitBean*-luokan avulla. Jos ylläpitäjä antaa väärää tietoa johonkin kenttään, hänet ohjataan takaisin asetussivulle, jolla virhe ilmoitetaan. Tietojen ollessa oikeellisia luokka kutsuu Tahiti–Apin *changeSettings*-metodia, joka päivittää asetukset.

#### 7.1.13 LoginController

LoginController-luokka käsittelee käyttäjän sisään- ja uloskirjautumispyynnöt ja lähettää ne eteenpäin Tahiti–Apin login- tai logoff-metodeille. Jos sisäänkirjautuminen onnistuu, käyttäjä ohjataan takaisin sivulle, jolta hän kirjautui sisään, ja sisäänkirjautumislomakkeen kohdalle ilmestyy nappi, jonka avulla käyttäjä voi kirjautua ulos järjestelmästä. Jos käyttäjätunnus tai salasana ovat virheelliset, käyttäjälle ilmoitetaan siitä sisäänkirjautumislomakkeen yhteydessä. Kirjautumispyyntöön käytetään LoginBeania, jonka luokka välittää Tahiti–Apille. Tahiti–Apin palauttamasta vastauksesta kootaan LoginDataBean, joka välitetään JSP-sivuille.

#### 7.1.14 EventHistoryDataController

Ylläpitäjän halutessa tapahtumahistoriasivulle tämä luokka hakee Tahiti–Apin *getLogEntries*-metodin avustuksella järjestelmästä tapahtumahistoriaa. Haun suoritettuaan tapahtumahistoria laitetaan *EventHistoryDataBean*-luokan ilmentymään, ja ylläpitäjä siirretään tapahtumahistoriasivulle, jolla itse tapahtumahistoriatieto näytetään.

#### 7.1.15 UserRegisterController

UserRegisterController-luokka käsittelee käyttäjän tekemän rekisteröintipyynnön. Käyttäjän antamien parametrien oikeellisuus tarkistetaan *UserRegisterBean*-luokan avulla. Jos rekisteröintipyyntö on oikeanmuotoinen, se lähetetään eteenpäin Tahiti–Apin *registrationRequest*-metodille. Virheellisestä syötteestä ilmoitetaan käyttäjälle rekisteröintipyyntösivulla.

## 7.2 Bean-tietoluokat

Tietoa järjestelmän käyttöliittymän eri komponenttien välillä siirretään JavaBean-luokissa. Järjestelmän tietoja ovat Tahiti-Apin lähettämät tiedot käyttöliittymälle sekä käyttäjän antamat tiedot www-lomakkeiden yhteydessä. Kaikille käyttöliittymän tietoluokille on yhteinen rajapinta, jota käytetään Tahiti-Apin metodien parametrien välitykseen. Jokaisessa Bean-tietoluokassa on metodi, jonka avulla luokan sisältämien tietojen oikeellisuus tarkistetaan.

### 7.2.1 AsteroidQueryBean

AsteroidQueryBean-luokka sisältää käyttäjän antamat asteroidihakuehdot. Tätä luokkaa käytetään sekä hakusivun hakujen suorittamiseen että tarkennettujen hakujen tekemiseen valokäyräsivulla. Luokan ilmentymä annetaan JSP-moottorilta parametrina AsteroidQueryController-kontrolliluokalle.

### 7.2.2 LightcurveSubmitBean

LightcurveSubmitBean sisältää käyttäjän syöttämät valokäyrätiedot. Luokan ilmentymä annetaan JSP-moottorilta parametrina LightcurveSubmitController-kontrolliluokalle.

### 7.2.3 LightcurveDataBean

LightcurveDataBean sisältää yhden asteroidin kaikki hakuehdot täyttävät valokäyrät. Kontrolliluokka AsteroidDataController luo tämän luokan Api:lta saamansa valokäyrädatan pohjalta.

### 7.2.4 AsteroidTrajectoryDataBean

AsteroidTrajectoryDataBean sisältää yhden asteroidin kaikki rataelementit. Kontrolliluokka AsteroidDataController luo tämän luokan Api:lta saamansa rataelementtidatan pohjalta.

### 7.2.5 EarthTrajectoryDataBean

EarthTrajectoryDataBean sisältää maapallon kaikki rataelementit. Kontrolliluokka AsteroidDataController luo tämän luokan Api:lta saamansa rataelementtidatan pohjalta.

### 7.2.6 UserInfoDataBean

UserInfoDataBean sisältää yhden käyttäjän kaikki tiedot. Luokan ilmentymä voidaan antaa JSP-moottorilta kontrolliluokalle uuden käyttäjän rekisteröinnin tai käyttäjätietojen päivityksen yhteydessä, mutta se voidaan antaa myös kontrolliluokalta JSP-sivuille käyttäjätietoja haettaessa.

### 7.2.7 UserSearchBean

UserSearchBean sisältää käyttäjän tekemän käyttäjähaun syötteen. JSP-moottori toimittaa tämän luokan ilmentymän kontrolliluokalle. Käyttäjätasosta riippuu, mitä tietoja käyttäjähaussa voi

antaa.

### 7.2.8 UserSearchDataBean

UserSearchBean sisältää käyttäjän tekemän käyttäjähaun tulokset. Kontrolliluokka toimittaa tämän luokan ilmentymän JSP-sivuille. Käyttäjätasosta riippuu, mitä tietoja järjestelmä palauttaa käyttäjähaun tuloksena.

### 7.2.9 LoginBean

LoginDataBean sisältää sisään- tai uloskirjautuvan käyttäjän toiminnon syötteet, eli käytännössä tunnuksen ja salasanan tai uloskirjautumispyynnön.

### 7.2.10 LoginDataBean

LoginDataBean sisältää järjestelmän vastauksen kirjautumispyyntöön.

### 7.2.11 EventHistoryDataBean

EventHistoryDataBean sisältää järjestelmän antamat tapahtumahistoriatiedot. Kontrolliluokka antaa tämän luokan ilmentymän JSP-sivulle.

### 7.2.12 SiteConfigurationDataBean

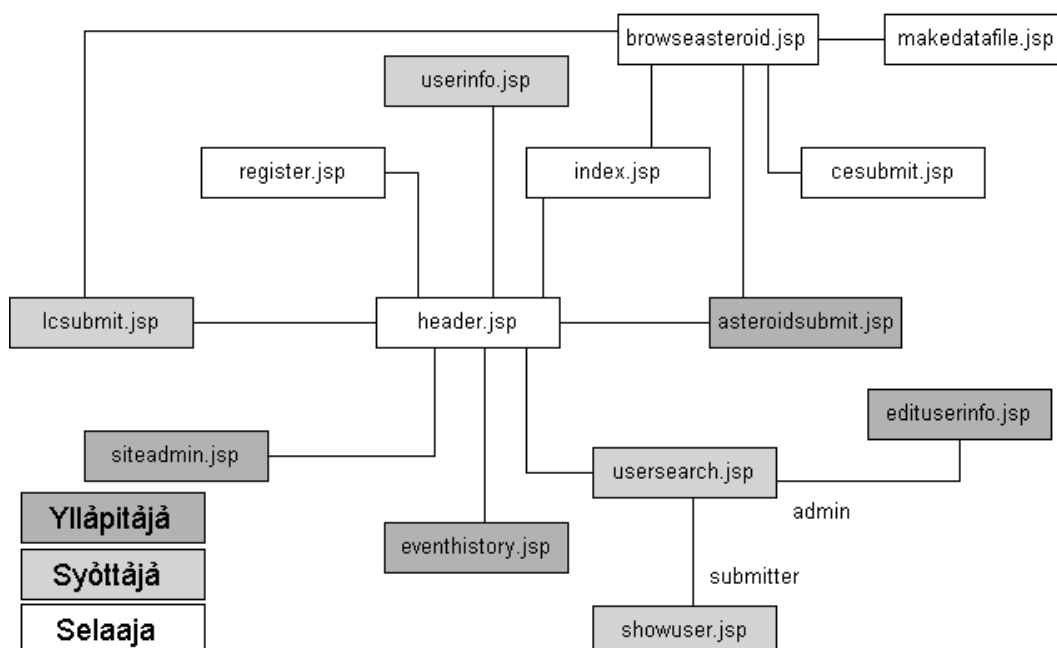
SiteConfigurationDataBean sisältää järjestelmän asetustiedot. Tämän luokan ilmentymä annetaan kontrolliluokalta JSP-sivulle tai JSP-moottorilta kontrolliluokalle.

## 8 Jsp-käyttöliittymäsivut

Alla on kuvaukset jokaisesta järjestelmän tärkeimmästä käyttöliittymäsivusta. Tärkeimmistä käyttöliittymän lomakkeista on esitetty listamuodossa kaikki kentät ja niiden tarkoitukset. Monista pienemmistä lomakkeista on sanallinen selitys sekä kuva. Käyttöliittymäkuvat eivät ulkoasullisesti tai teksteiltään vastaa lopullista tuotetta. Ne ovat tarjottu dokumentin lukijalle havainnollistamaan niihin liittyvää tekstiä. Joistakin yksinkertaisemmista sivuista on jätetty havainnollistava kuva pois, koska sivulla esiintyvät komponentit voidaan selittää sanallisesti riittävän tarkasti ja havainnollisesti.

Www-sivuston sivukartta on esitetty kuvassa 15. Header.jsp-sivu sisältää navigointipalkin, jolta pääsee kaikille järjestelmän pääsivuille käyttäjäryhmäkohtaisesti. Sivukartassa on väritetty eri väreillä sivut, jotka kuuluvat tietylle käyttäjäryhmälle.

Jokaisella järjestelmän sivulla näkyy navigaatiopalkki sivun yläreunassa. Navigaatiopalkin avulla käyttäjä voi navigoida järjestelmän eri pääsivujen välillä. Navigaatiopalkissa käyttäjän senhetkinen sivu on visualisoitu lihavoimalla kyseisen sivun linkki. Navigointipalkin linkit muuttuvat



Kuva 15: Järjestelmän sivukartta.

sen mukaan, mihin käyttäjäryhmään sivujen käyttäjä kuuluu. Käyttäjäryhmäkohtaiset navigaatiopalkkinäkymät ovat listattu alla, kuva ylläpitäjän navigaatiopalkista on esitelty header.jsp-sivun kuvauksen yhteydessä.

Selaaajalla on näkyvissä seuraavat linkit:

- Home – Järjestelmän aloitussivu, jolla sijaitsee asteroidihaku.
- Register – Rekisteröintisivu.

Syöttäjälle navigointipalkissa näytetään seuraavat linkit:

- Home – Järjestelmän aloitussivu, jolla sijaitsee asteroidihaku.
- Submit lightcurve data – Lisää järjestelmään uutta valokäyrädataa.
- User information – Käyttäjätietojen muuttaminen.
- User search – Muiden käyttäjien tietojen haku sekä katsominen.

Ylläpitäjän navigointipalkissa on seuraavat linkit:

- Home – Järjestelmän aloitussivu, jolla sijaitsee asteroidihaku.
- Register – Rekisteröintisivu.
- Submit lightcurve data – Lisää järjestelmään uutta valokäyrädataa.
- Submit asteroid data – Lisää järjestelmään uutta asteroidien ratatietodataa.

- User information – Käyttäjätietojen muuttaminen.
- User admin – Käyttäjähallinta.

## 8.1 header.jsp – Järjestelmän otsikkosivu

Järjestelmän otsikkosivulla määritellään html-sivun otsikkotagit. Samoin sivu luo käyttäjälle hänen navigaatiopalkkinsa, sisään- tai uloskirjautumislomakkeen sekä määrittää sivuilla käytettävän tyylytiedoston ja sivun otsikon.



Kuva 16: Otsikko sekä ylläpitäjän navigaatiopalkki

## 8.2 index.jsp – aloitussivu

Järjestelmän aloitussivu toimii samalla asteroidihakusivuna. Hakusivu on jaettu kahteen osaan, haku-ehdotoihin sekä hakutuloksiin. Perusasetuksena on tyhjä haku, joka tuottaa hakutuloksiin kaikkien asteroidien nimet, joilla on vähintään yksi valokäyrä järjestelmässä.

Hakuehto on toteutettu normaalina html-lomakkeena kuvassa 17. Lomakkeessa ovat seuraavat kentät:

- Asteroidin nimi tai tunnus (asteroid name or designation)
- Asteroidin numero (asteroid number)
- Aikaväli (time interval) – käyttäjä voi jättää toisen ajoista tyhjiksi. Alkuajankohdan jäädessä tyhjäksi järjestelmä etsii valokäyriä, jotka ovat havaittu viimeistään loppuajankohtaan mennessä. Loppuajankohdan jäädessä tyhjäksi järjestelmä hakee valokäyriä, jotka ovat tapahtuneet alkuajankohdan jälkeen.
- Havainnoitsijan nimi (observer name) – Havainnoitsijan osittainen nimi kelpuutetaan.
- Valokäyrien minimimäärä (minimum lightcurves) – Vähintään näin monta valokäyriä, tyhjäksi jätettynä järjestelmä hakee asteroideja, joilla on vähintään yksi valokäyrä.
- Havaintopisteiden vähimmäismäärä (minimum data points) – Vähintään näin monta havaintopistettä, tyhjäksi jätettynä järjestelmä hakee asteroideja, joilla on valokäyrä, jossa on vähintään yksi havaintopiste.
- Minimi ja maksimi vaihekulma (Min/Max phase angle) – Kaksi kenttää vaihekulman määrittelyyn. Vasemmanpuoleinen kenttä on minimiarvolle, oikeanpuoleinen maksimiarvolle. Kenttät voidaan jättää täysin tyhjiksi, täyttää vain toinen niistä tai täyttää kummatkin niistä.

Asteroid name or designation

Asteroid number

Time interval (DD.MM.YYYY)  .  .  -  .  .

Observer name

Observing site

Minimum lightcurves

Minimum data points

Min/Max phase angle

Min/Max ecliptical latitude

Min/Max ecliptical longitude

Min/Max heliocentric distance

Min/Max geocentric distance

Information

Detector 

- Photoelectric
- CCD
- Other

Filters

Visual  Ultraviolet

Unfiltered  Blue

Infrared  Red

---

Abs. Photom.  LT Corrected

Kuva 17: Asteroidihaku



- Minimi ja maksimi epliktikaalinen leveys (Min/Max ecliptical latitude) – Kaksi kenttää epliktikaalisen leveyden määrittelyyn, vasemmanpuoleiseen kenttään laitetaan minimiarvo, oikeanpuolimmaiseen maksimiarvo. Kentät voidaan jättää täysin tyhjiksi, täyttää vain toinen niistä tai täyttää kummatkin niistä.
- Minimi ja maksimi epliktikaalinen pituus (Min/Max ecliptical longitude) – Kaksi kenttää ekliptikaalisen pituuden määrittelyyn, vasemmanpuoleiseen laitetaan minimiarvo, oikeanpuolimmaiseen maksimiarvo. Kentät voidaan jättää täysin tyhjiksi, täyttää vain toinen niistä tai täyttää kummatkin niistä.
- Minimi- ja maksimimatka auringosta (Min/Max heliocentric distance) – Kaksi kenttää, joilla voidaan määrittää minimi (vasen) sekä maksimi (oikea) matka auringosta asteroidiin. Kentät voidaan jättää täysin tyhjiksi, täyttää vain toinen niistä tai täyttää kummatkin niistä.
- Minimi- ja maksimimatka maasta (Min/Max geocentric distance) – Kaksi kenttää, joilla voidaan määrittää minimi (vasen) sekä maksimi (oikea) matka maa-planeetasta asteroidiin. Kentät voidaan jättää täysin tyhjiksi, täyttää vain toinen niistä tai täyttää kummatkin niistä.
- Havaintoväline (detector) – Järjestelmä hakee kaikki asteroidit, joiden havainnot ovat tehty käyttäjän määrittelemillä laitteilla. Käyttäjällä on valittavanaan kolme eri vaihtoehtoa, tai mikä tahansa niiden yhdistelmistä: Photoelectric, CCD tai other. Käyttäjän valitessa valinnan other, hänen tulee kirjoittaa muun laitteen osittainen tai täydellinen nimi valintalaatikon vieressä olevaan kenttään.
- Suotimet (filters) – Kuusi erillistä valintalaatikkaa järjestelmässä käytettäville suotimille näkyvä (visual), suodattamaton (unfiltered), infrapuna (infrared), ultravioletti (ultraviolet), sininen (blue) sekä punainen (red).
- Absoluuttinen fotometria (abs. photom.)
- Valoaikakorjaus (LT Corrected)

Lomakkeen lopussa on search-nappi, jolla käyttäjä voi antaa käskyn järjestelmälle haun suorittamisesta. Search-napin vieressä on reset-nappi, jolla käyttäjä voi palauttaa hakulomakkeen oletusarvoihin.

Käyttäjän hakiessa tietoja tulokset esitellään hakutulostaulussa. Sivun ollessa perustilassa hakutulostaulussa esitetään kaikki asteroidit, joilla on vähintään yksi valokäyrä, jossa on vähintään yksi havaintopiste. Hakutulostaulussa, joka on esitetty kuvassa 18, ovat seuraavat sarakkeet:

- Asteroidin numero (number)
- Asteroidin nimi sekä mahdollinen tunnus (asteroid name)
- Valokäyrien määrä (lightcurves)
- Havainnoitsijoiden määrä (observers)
- Viimeinen havaintopäivämäärä (observation date)

Number	Asteroid name	Lightcurves	Observer(s)	Observation date
	(1999 CM138)	2	1	03.02.2001
	AMOS	12	2	23.01.1988
	Augustirostris	23	2	01.10.1984
	Bukovanska	16	3	09.05.1994
	California	53	12	10.01.2003
	Chaplin	5	1	11.04.1997
	Dawson	22	5	16.05.2002
243	Ida	7	2	16.09.1980
#, 1-200, 201-300				
101-130, 131-160, 161-200				
<b>8 asteroid(s) and 126 lightcurve(s) found.</b>				

Kuva 18: Asteroiditulostaulu

Jos asteroidille on järjestelmässä sekä nimi että tunnus, tunnus esitetään nimen perässä suluissa. Asteroidit, joille on pelkkä tunnus annetaan tunnus nimikentässä suluissa.

Asteroidit ovat oletusarvoisesti järjestetty taulukossa numeron mukaan nousevaan järjestykseen. Käyttäjä voi vaihtaa asteroidin nimen mukaiseen järjestykseen painamalla ”Asteroid name”-otsikkoa hakutulostaulussa. Senhetkinen järjestys näytetään lihavoituna otsikkotekstinä. Hakutulostaulun asteroidien nimet tai tunnukset ovat linkkejä kyseisen asteroidin tarkemmat tiedot sisältävälle sivulle. Asteroidin viimeinen havaintoaika on myös linkki, jolla käyttäjä pääsee automaattisesti edellämainitulle sivulle, ja siellä hänelle esitellään kyseisen viimeisen havainnon tiedot. Jos hakutulostauluun tulee enemmän kuin 40 asteroidia, annetaan käyttäjälle mahdollisuus selata erillisten tulossivujen kautta kaikkia tulosasteroideja. Erilliset hakutulossivut jaetaan maksimissaan kymmeneen eri alisivuun. Algoritmi toimii hierarkisesti, ja jakaa koko tulosalueen ensin kymmeneen yhtä suureen osaan. Näiden osien alku- ja loppunimistä tai numeroista määritellään alueiden alku- ja loppukirjaimet tai numerot. Käyttäjän valitessa ylimmän tason alueen, järjestelmä avaa uuden maksimissaan kymmeneen osaan jakautuvan alisivuston. Näin jatketaan, kunnes ollaan päädytty hakutulostaulussa sivuihin, jotka ovat jakautuneet enintään neljääkymmeneen eri asteroidiin.

### 8.3 register.jsp – rekisteröitymissivu

Rekisteröitymissivulla selaajalla on mahdollisuus täyttää lomake, jolla lähetetään järjestelmään rekisteröintipyyntö. Rekisteröintipyyntölomake on jaettu pakollisiin sekä vapaaehtoiisiin kenttiin. Lomake on kuvattu kuvassa 19.

Lomakkeen pakolliset kentät ovat:

- Käyttäjätunnus (username) – Henkilön haluama käyttäjätunnus. Käyttäjätunnuksen on oltava vähintään neljämerkkinen. Käyttäjätunnuksen on oltava yksikäsitteinen ja se saa sisältää vain numeroita sekä kirjaimia.

Username	<input type="text"/>
Name	<input type="text"/>
E-mail	<input type="text"/>
Status	<input type="text" value="Amateur"/>
Additional information	<input type="text"/>
Reference name	<input type="text"/>
Reference email	<input type="text"/>
<b>Default values</b> (optional)	
Observing site	<input type="text"/>
Photometric system	<input type="text"/>
Detector	<input type="text" value="Photoelectric"/>
Time standard	<input type="text" value="Standard signal"/>
<input type="button" value="register"/>	

Kuva 19: Rekisteröintipyyntölomake

- Henkilön nimi (name) – Rekisteröityvän henkilön tai yhteisön nimi.
- Sähköpostiosoite (e-mail) – Henkilön tai yhteisön sähköpostiosoite. Sähköpostiosoitteen muoto tarkistetaan yksinkertaisella tarkistuksella oikeelliseksi.
- Henkilön ammattimaisuus (status) – Kenttä, jolla henkilö voi ilmaista, onko hän ammattimainen havainnoitsija (tähtitieteilijä) vai harrastelija.
- Lisätietoja hakijasta (additional information) – Käyttäjä voi halutessaan antaa lisätietoja itsestään, laitteistostaan tai mistä tahansa muusta asiasta tähän vapaamuotoiseen tekstikenttään.
- Suosittelijan nimi (reference name) – Suosittelijan nimi tähän kenttään. Suosittelijan nimen tulee olla vähintään viisikirjaiminen.
- Suosittelijan sähköpostiosoite (reference email) – Suosittelijan sähköpostiosoite tulee tähän kenttään. Sähköpostiosoite tarkastetaan, kuten rekisteröityvän henkilön antama sähköpostiosoitekin.

Lomakkeen vapaaehtoiset kentät, jotka ovat käyttäjän oletusarvoja havaintojen syöttämisen yhteydessä, ovat:

- Havainnointipaikka (observing site) – Rekisteröijä voi antaa oletusarvoisen havainnointipaikan.
- Fotometrinen järjestelmä (photometric system) – Rekisteröijän käyttämä fotometrinen järjestelmä.
- Havainnointiväline (detector) – Valintakenttä, josta rekisteröijä voi valita kolme erilaista havainnointivälinettä: Photoelectric, CCD, other. Jos rekisteröijä valitsee vaihtoehdon other, hänen tulee antaa vapaavalintainen havainnointiväline pudotusvalikon oikealla puolella olevaan tekstikenttään.
- Aikalähde (time standard) – Rekisteröijä voi valita pudotusvalikosta erilaisia aikalähteitä. Pudotusvalikossa ovat vaihtoehdot Standard signal (standardisignaali), computer clock (tietokoneen kello) tai other. Jos rekisteröijä valitsee vaihtoehdon other, hänen on annettava vapaavalintainen aikalähde pudotusvalikon oikealla puolella olevaan tekstikenttään.

Lomakkeen alapuolella on “register”-nappi, jolla rekisteröintihakemus lähetetään järjestelmään. Rekisteröintihakemuksen lähdettyä järjestelmään käyttäjä ohjataan takaisin etusivulle index.jsp.

#### 8.4 browseasteroid.jsp – Asteroidin tiedot-sivu

Asteroidin tiedot-sivulla esitetään ne valokäyrätiedot, jotka vastaavat käyttäjän antamaa asteroidihakuehtoa. Index.jsp-sivun hakuehdot kopioituvat tämän sivun valokäyrähakulomakkeeseen, joka on samankaltainen kuin index.jsp-sivun lomake. Hakulomake on esitetty kuvassa 20.

Käyttäjän hakuehtoa vastaavat asteroidin valokäyrät näytetään taulussa hakulomakkeen alla. Taulussa on esitetty kuvassa 21, ja siinä on seuraavat kentät:

**search filters**

Time interval  .  .  -  .  .

Minimum phase angle

Minimum height

Min. datapoints

Observer

Observing site

Information

Time standard

Detector

Visual  Ultraviolet

Unfiltered  Blue

Infrared  Red

Abs. photometry  LT Corrected

Kuva 20: Valokäyrähakulomake

Entry date	Observer	Observation date	Asteroid element	Earth element	Points	Filter
<input checked="" type="checkbox"/>	20.02.2003	McDonald	23.01.1988	06.05.2002 <input type="text" value="custom"/>	01.17.2001 <input type="text" value="custom"/>	40 Visual
<input checked="" type="checkbox"/>	20.02.2003	McDonald	25.1.1988	06.05.2002 <input type="text" value="custom"/>	01.01.1992 <input type="text" value="custom"/>	40 Visual

Get data as intensity

**2 lightcurves found**

Kuva 21: Valokäyrätaulostaulu

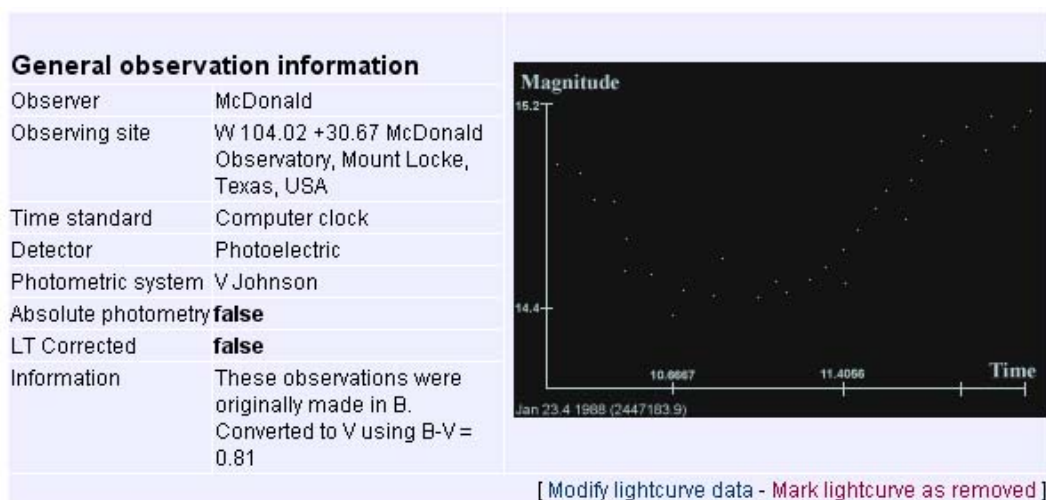
- Kirjauspäivämäärä (entry date) – Valokäyrän kirjauspäivämäärä järjestelmään.
- Havainnoija (observer)
- Havainnointipäivämäärä (observation date)
- Asteroidin rataelementit (asteroid element) – Vapaavalintainen pudotusvalikko, jolla valitaan kyseiselle valokäyrälle asteroidin rataelementti. Listaan ilmestyvät kaikki asteroidille löytyvät rataelementit järjestelmästä, ja oletusvalintana on valokäyrän antama oletusrataelementti. Pudotusvalikon vieressä on ”custom”-painike, jolla käyttäjä voi antaa erillisen sivun kautta omat rataelementtinsä.
- Maan rataelementit (earth element) – Identtinen pudotusvalikko, kuin yllä asteroidin rataelementille, jossa on kaikki maan rataelementit järjestelmässä. Käyttäjä voi myös antaa omat rataelementit samanlaisella ”custom”-napin painalluksella.
- Havaintopisteiden määrä (points) – Valokäyrässä olevien havaintopisteiden määrä.
- Suodin (filter) – Millä suotimella valokäyrä on otettu.
- Valintalaatikko jokaisen asteroidin vasemmalla puolella. Tällä valintalaatikolla käyttäjä valitsee kyseisen valokäyrän mukaan siihen toimintoon, jonka käyttäjä haluaa. Toiminnot ovat kuvattu alla.

Valokäyrätulostaulussa on valokäyrätietojen lisäksi valintalaatikko, jolla käyttäjä voi valita halutessaan valokäyrän tiedot joko magnituditietona tai intensiteettitietona. Tämä tieto näkyy vain erillisissä tulostiedoissa, sivulla näytettävä valokäyrän raakadata on aina magnitudimuodossa.

Taulussa on kolme nappia, jotka kaikki tekevät eri toiminnon. ”Show raw data”-napilla käyttäjä voi hakea asteroidin tiedot-sivulle kaikkien valitsemiensa valokäyrien tiedot. Tietojen yhteydessä havaintopisteistä luodaan kuva. ”Make data file”-napilla käyttäjä voi luoda valitsemistaan valokäyristä ja niiden ratatiedoista erillisen lasketun tulostiedoston. Tulostiedosto näytetään tekstinä erillisellä sivulla, jolla on myös mahdollisuus kopioida se tiedostomuodossa itselleen. Viimeisenä ”Make raw file”-nappi luo valokäyrän tiedoista raakadatamaisen tulostiedoston, joka näytetään eri sivulla. Raakadatatieosto muistuttaa järjestelmään syötettäviä atlas-tiedostoja.

Jos käyttäjä haluaa nähdä valokäyrän tietoja painamalla ”show raw data”-nappia, tiedot näytetään havainnointidata-otsikon alla kaikista valitsemista valokäyristä. Valokäyrätietotaulussa esitetään kaikki järjestelmään talletetut arvot valokäyrästä sekä taulun oikeassa reunassa kuva havaintopisteistä ajan sekä magnitudin funktiona. Taulun kuva on kuvassa 22. Jos käyttäjä on ylläpitäjä tai syöttäjä ja valokäyrä on hänen syöttämänsä, hänelle näytetään valokäyrän poistolinkki. Ylläpitäjä näkee poistolinkin lisäksi vielä valokäyrän muuntolinkin, jolla ylläpitäjä pääsee valokäyrän syöttösivulle, jossa lomake on esitetytty valokäyrän tiedoilla.

Lopuksi sivun alalaidassa näytetään kaikki järjestelmään talletetut asteroidin ratatiedot sekä kaikki maan ratatiedot. Ratatietotaulu on esitetty kuvassa 23.



Observation geometry						
System	J2000.0 ecliptic					
Epoch	2447183.9 (1988 Jan 23.4)					
Earth	-0.532668	0.827645	0.000002	-0.014747	-0.009376	0.000000
Object	-2.636411	1.063640	-0.012989	-0.004220	-0.009323	-0.000200
Aspect data	2.8429	2.1169	15.60	173.60	-0.40	2447183.90000 J2000.0

Lightcurve data		
Time	Visual	Error
9.9528	14.8605	.0046
10.0306	14.7550	.0047
10.0583	14.7239	.0050
10.0861	14.6927	.0032

Kuva 22: Valokäyrän tiedot

Entry date	Epoch	Axis (a) (AU)	Eccentricity (e)	Inclination (i) (deg)	Longitude (W) (deg)	Perihelion (w) (deg)	Anomaly (M) (deg)
19.02.2003	20020506	2.8619752	0.12241260	11.31752	207.39735	276.41296	16.2648000

Kuva 23: Asteroidin sekä maan ratatietotaulu

## 8.5 cesubmit.jsp – Omien ratatietojen syöttö

Sivulla käyttäjälle esitetään lomake, johon käyttäjä syöttää omat rataelementtinsä valitsemaalleen valokäyrälle. Samalla sivulla voidaan syöttää joko maan tai asteroidin ratatiedot. Sivulla on lomake tätä varten, jossa ovat kenttinä samat attribuutit kuin asteroidin syöttösivulla olevat sarakkeiden nimet.

Käyttäjän painaessa submit nappia, ratatiedot liitetään oikean valokäyrän yhteyteen ja käyttäjä ohjataan takaisin asteroidien tiedot-sivulle. Reset-näppäällä käyttäjä voi tyhjentää lomakkeen kentät.

## 8.6 datafile.jsp – Datatiedoston näyttö

Sivulla näytetään käyttäjälle hänen valitsemistaan valokäyrädataista koostettu tieto, joko raaka-muodossa (jos käyttäjä painoi ”make raw file”-nappia tai lasketussa muodossa (jos käyttäjä painoi ”make data file”-nappia). Tietojen näyttämisen lisäksi sivulla on linkki, jolta käyttäjä voi siirtää tiedon tiedostona omalle koneellensa.

## 8.7 lsubmit.jsp – Valokäyrän syöttö

Valokäyrän syöttösivulla käyttäjä voi syöttää järjestelmässä jo oleville asteroideille valokäyrätietoja. Tiedot syötetään lomakkeeseen (kuva 24), jonka pakolliset kentät on selitetty seuraavassa:

- Asteroidin numero (object number) – Pakollinen kenttä, jos asteroidille ei ole annettu nimeä.
- Asteroidin nimi (object name) – Pakollinen kenttä, jos asteroidille ei ole annettu numeroa.
- Asteroidin tunnus (object designation) – Pakollinen kenttä, jos asteroidille ei ole annettu nimeä eikä numeroa.
- Havainnointipaikka (observing site) – Kenttään tulee käyttäjän mahdollinen oletushavainnointipaikka. Jos käyttäjällä ei ole oletushavainnointipaikkaa, tai hän haluaa laittaa jonkin muun paikan, kenttää on mahdollisuus editoida.
- Nolla-aika (zero time)
- Nollamagnitudi (zero magnitude)
- Havainnointiaikayksikkö (unit of time)
- Absoluuttinen fotometria (absolute photometry)
- Valoaikakorjaus (lighttime corrected)
- Valokäyrätiedon sarakkeet (columns) – Matriisi, jossa ovat seuraavat suodinrivit: Aika, ultravioletti, sininen, näkyvä, punainen, infrapuna, suodattamaton sekä virhe. Jokaiselle havainnon suotimelle käyttäjän antamassa havaintotiedostossa määritellään paikka pystyrivissä olevien numeroitujen sarakkeiden kautta. Jos jotakin suodinta ei ole havaintotiedoissa, jätetään sarakkeeksi ”none”. Erotinmerkkinä kentille käytetään sarkainmerkkejä tai välilyöntejä.



Object number	<input type="text"/>
Object name	<input type="text"/>
Object designation	<input type="text"/>
Observing site	<input type="text" value="User's default"/>
Observing time	<input type="text"/>
Zero time	<input type="text"/>
Zero magnitude	<input type="text" value="0"/>
Unit of time	<input type="text" value="Day"/>
Absolute photometry	<input type="checkbox"/>
Lighttime corrected	<input type="checkbox"/>
<b>Columns</b>	
	1 2 3 4 5 6 7 8 None
Time	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input checked="" type="radio"/>
U	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input checked="" type="radio"/>
B	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input checked="" type="radio"/>
V	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input checked="" type="radio"/>
R	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input checked="" type="radio"/>
I	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input checked="" type="radio"/>
Unfiltered	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input checked="" type="radio"/>
Error	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input checked="" type="radio"/>
Data	<input type="text"/>
<b>Optional information</b>	
Photometric system	<input type="text" value="User's default here too!"/>
Detector	<input type="text" value="And here"/>
Time standard	<input type="text" value="Even here"/>
Reference (if published)	<input type="text"/>
Additional information	<input type="text"/>
<input type="button" value="submit data"/>	

Kuva 24: Valokäyrien syöttölomake

- Valokäyrän havaintotiedot (data) – Lomake ottaa vastaan tekstimuotoisena valokäyrän havainnointitiedot. Tekstin on vastattava valokäyrätiedon sarakkeissa määriteltyä sarakemuotoa.

Pakollisten kenttien lisäksi valokäyrälle voidaan antaa seuraavat vapaaehtoiset kentät:

- Fotometrinen järjestelmä (photometric system) – Käyttäjän oletettu fotometrinen järjestelmä laitetaan tähän kenttään. Jos käyttäjällä ei ole kyseistä oletusarvoa, hän voi kirjoittaa kenttään vapaamuotoisen arvon.
- Havainnointiväline (detector) – Käyttäjän oletusarvoinen havainnointiväline.
- Aikalähde (time standard) – Käyttäjän oletusaikalähde. Jos sellaista ei ole, kenttään voi kirjoittaa haluamansa aikalähteen.
- Lähde (reference) – Jos valokäyrä on julkaistu jossain lähteessä, havainnoijan tulisi ilmoittaa siitä tässä.
- Lisäinformaatiota (additional information) – Käyttäjä voi kirjoittaa tähän kenttään havainnon oton aikana vallinneista olosuhteista tai muista satunnaisista havaintoon liittyvistä asioista.

Painamalla submit-nappia käyttäjä antaa tiedot järjestelmälle. Tietojen olessa virheellisiä käyttäjälle ilmoitetaan virheistä tällä sivulla.

## 8.8 asteroidsubmit.jsp – Asteroidin ratatietojen syöttö

Ylläpitäjä voi syöttää tämän sivun kautta järjestelmään uusia asteroideja ja niiden ratatietoja. Asteroidiksi luetaan myös maa. Sivulla on yksi lomake, jonka kautta käyttäjä voi antaa tietoja järjestelmälle. Lomake on kuvattu kuvassa 25.

Asteroidien tietojen syöttölomakkeessa on samankaltainen matriisi kuin valokäyrien syöttölomakkeessa. Matriisissa on kymmenen eri saraketta, joista kaikki paitsi referenssi ovat pakollisia. Käyttäjä valitsee sarakejärjestyksen joka vastaa hänen tiedostossaan tai tekstissään olevaa sarakejärjestystä.

Tietotekstikenttään (data text) käyttäjä liimaa tekstimuotoisena asteroidien rataelementtitiedot. Jos käyttäjällä on tiedot erillisessä tiedostossa, hän voi liittää kyseisen tiedoston tiedostokenttään (data file).

Painamalla ”submit”-nappia, käyttäjä lähettää tiedot järjestelmälle. Jos tiedoissa on virheitä, käyttäjälle ilmoitetaan näistä tällä sivulla.

## 8.9 userinfo.jsp – Käyttäjän omien tietojen päivitys

Käyttäjän omien tietojen päivitys tapahtuu lomakkeella, jossa ovat samankaltaiset kentät kuin rekisteröintilomakkeessa. Uusina kenttinä lomakkeesta löytyvät salasanan vaihtamiseen tarkoitettut

Columns	1	2	3	4	5	6	7	8	9	None
Asteroid designation	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
EPOCH	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
REF	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
a (AU)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
Eccentricity	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
Inclination	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
Longitude (W)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
Perihelion (w)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
Anomaly (M)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
Data text										

Data file

Kuva 25: Asteroidien ratatiedon syöttölomake

Username	<input type="text"/>
Name	<input type="text"/>
E-mail	<input type="text"/>
Status	<input type="text" value="Amateur"/>
Additional information	<input type="text"/>
Reference name	<input type="text"/>
Reference email	<input type="text"/>
<b>Default values</b> (optional)	
Observing site	<input type="text"/>
Photometric system	<input type="text"/>
Detector	<input type="text" value="Photoelectric"/>
Time standard	<input type="text" value="Standard signal"/>
<input type="button" value="register"/>	

Kuva 26: Käyttäjän tietolomake

uusi salasana- (new password) ja uuden salasanan varmistuskenttä (confirm new password). Käyttäjän ei myöskään anneta muuttaa rekisteröinnin yhteydessä antamaansa suosittelijan nimeä tai sähköpostiosoitetta. Käyttäjien omien tietojen päivitysloMAKE on kuvassa 26.

Käyttäjän painaessa ”edit”-nappia, uudet tiedot lähetetään järjestelmälle. Tiedot tarkastatetaan ja virheistä ilmoitetaan käyttäjälle tällä sivulla.

## 8.10 usersearch.jsp – Käyttäjien etsintä



Kuva 27: Ylläpitäjien käyttäjäetsintätulostaulu

Käyttäjien etsintä –sivulla näytetään eri tietoja riippuen siitä, mihin käyttäjäryhmään sivujen käyttäjä kuuluu. Sekä syöttäjille että ylläpitäjille annetaan sama etsintämahdollisuus lomakkeella. He voivat hakea käyttäjiä järjestelmästä osittaisen nimen perusteella. Ylläpitäjille kuitenkin näytetään enemmän informaatiota käyttäjistä kuin syöttäjälle hakutuloksissa. Kuva 27 on ylläpitäjän näkymä. Syöttäjälle ei näytetä käyttäjätunnusta.

Ylläpitäjälle annetaan kaksi erillistä linkkiä hakutulostaulun oikeassa reunassa. ”Change”-linkillä ylläpitäjä voi siirtyä sivulle, jolla päivitetään kyseisen käyttäjän kaikkia tietoja. ”Remove”-linkillä ylläpitäjä voi poistaa kyseisen käyttäjän järjestelmästä kokonaan. Syöttäjälle annetaan mahdollisuus tarkastella yksittäisen käyttäjän kaikkia rekisteröintitietoja painamalla kyseisen henkilön nimeä, joka on linkki.

## 8.11 edituserinfo.jsp – Käyttäjän tietojen päivitys

Ylläpitäjä päivittää käyttäjien tietoja tämän sivun kautta. Sivun lomake on kuvassa 28. Lomake vastaa käyttäjän omaa tietojenpäivitysloMAKEtta. Ylläpitäjälle annetaan mahdollisuus myös muuttaa käyttäjän antamaa suosittelijan nimeä sekä salasanaa. Ylläpitäjä voi myös muuttaa käyttäjän käyttäjätasoa lomakkeen pudotusvalikon kautta (userlevel). Käyttäjätason lisäksi ylläpitäjä voi lisätä kommentin muille ylläpitäjille kyseisestä käyttäjästä (admin comment). Käyttäjän antaman suosittelijan sähköpostiosoitteen alla ilmoitetaan, jos kyseinen sähköpostiosoite tai nimi löytyy jo järjestelmästä.

Muutokset lähetetään järjestelmälle ”change”-napin painalluksella. Mahdollisista virheistä ilmoitetaan tällä sivulla. Sivulla on myös kolme muuta nappia: ”Send reference email”-nappi lähettää rekisteröijän määrittämälle suosittelijalle määrämuotoisen sähköpostin kysellen kyseisestä rekisteröijästä. ”Reset & send password”-nappi lähettää viestin rekisteröijälle uudesta salasanasta. Viimei-

<input type="button" value="send reference email"/>	<input type="button" value="reset &amp; send password"/>	<input type="button" value="send accept email"/>
Username	<input type="text"/>	
Name	<input type="text"/>	
E-mail	<input type="text"/>	
Status	<input type="text" value="Amateur"/>	
Additional information	<input type="text"/>	
Reference name	<input type="text"/>	
Reference email	<input type="text"/>	
<b>Mr. referrer</b> in our system as an administrator.		
<b>Default values</b> (optional)		
Observing site	<input type="text"/>	
Photometric system	<input type="text"/>	
Detector	<input type="text" value="CCD"/>	
Time standard	<input type="text" value="Computer clock"/>	
Userlevel	<input type="text" value="Browser"/>	
Admin comment	<input type="text"/>	
<input type="button" value="change"/>		

Kuva 28: Ylläpitäjien käyttäjän tietojen muuttolomake

senä ”Send accept email”-nappi lähettää hyväksymisviestin rekisteröityneelle käyttäjälle. Viestissä ilmoitetaan käyttäjän hyväksymisestä syöttäjäksi järjestelmään. Viestin yhteydessä rekisteröijälle generoidaan satunnainen salasana, joka liitetään viestiin.

## 8.12 siteadmin.jsp – Järjestelmän asetusten päivitys

Site admin email address	<input type="text"/>
Site smtp server and port	<input type="text"/>
<b>Info on how to create this email, and the special characters to use for substituting various fields in the email.</b>	
Registration email header	<input type="text"/>
Registration confirmation email template	<input type="text"/>
<b>Info on how to create this email, and the special characters to use for substituting various fields in the email.</b>	
Reference email header	<input type="text"/>
Reference email template	<input type="text"/>
<b>Text about the different substitutes you can use in the email template. (like %n for the person's name etc)</b>	
Lost password email header	<input type="text"/>
Lost password email template	<input type="text"/>
<input type="button" value="change"/>	

Kuva 29: Järjestelmäasetuslomake

Ylläpitäjä voi muuttaa erilaisia järjestelmän asetuksia tällä sivulla. Järjestelmäasetuslomake on kuvattu kuvassa 29. Lomakkeen kolme vapaata tekstimuotoista kenttää ovat sähköpostimallit kolmelle järjestelmästä lähetettävälle sähköpostille: rekisteröinnin hyväksymissähköposti (registration

confirmation), suosittelijalle lähetettävä kysely rekisteröijästä (reference email) sekä kadonneen salasanan resetointi (lost password email).

Järjestelmäasetusten muutokset lähetetään järjestelmälle painamalla ”change”-nappia.

### 8.13 eventhistory.jsp – Järjestelmän tapahtumahistoria

Järjestelmän tapahtumahistoriassa näytetään kaikki ylläpitäjälle mielenkiintoiset järjestelmässä tapahtuneet tapahtumat hänen viimeisen sisäänkirjautumisensa jälkeen. Tapahtumat ovat listattu määrittelydokumentissa. Sivulla tapahtumat jaetaan kolmeen eri tyyppiin, valokäyriin, asteroideihin sekä käyttäjätietoihin kohdistuviin tapahtumiin. Jokaiselle eri tapahtumaryhmälle on oma taulunsa. Ylläpitäjä voi selata tapahtumahistoriaa ”Start date” sekä ”End date” kenttien avulla. Aloituspäivämäärän jädessä tyhjäksi järjestelmä hakee kaikki tapahtumat, jotka ovat tapahtuneet lopetuspäivämäärään mennessä. Lopetuspäivämäärän jädessä tyhjäksi haetaan kaikki tapahtumat aloituspäivämäärästä nykyhetkeen. Kummankin ollessa tyhjiä haetaan kahden viimeisen kuukauden tapahtumat.

Karkea hahmotelma tapahtumahistoriasivusta on alla.

Start time: ..  
End time: ..

Event time	Event type	Username	Asteroid	Lightcurve identity	Changed data
01.01.2003 14:50	NEW LC	mcd	Ida (243)	McDonald 23.1.1988	
24.02.2003 00:45	MODIFY LC	nvaris	Ida (243)	McDonald 23.1.1988	Detector: [CCD => Photoelectric] DATAPoint(time): [9.9528 => 9.9599] DATAPoint(visual): [14.8621 => 15.2215]
24.02.2003 00:45	MODIFY LC	nvaris	Ida (243)	McDonald 23.1.1988	Detector: [CCD => Photoelectric] DATAPoint(time): [99.9528 => 9.99528]
24.02.2003 00:50	REMOVE LC	nvaris	Ida (243)	McDonald 23.1.1988	

Event time	Event type	Username	Asteroid
24.02.2003 00:30	NEW ASTEROID DATA	mcd	Ida (243)

Event time	Event type	Username	Account username	Changed data
24.01.2003 00:30	NEW ACCOUNT	nnash	nnash	
24.02.2003 00:30	CHANGE ACCOUNT	nvaris	nnash	Userlevel:["Pending approval" => "Registered user"]
25.02.2003 12:25	CHANGE ACCOUNT	mcd	nnash	Userlevel:["Registered user" => "Administrator"]
15.01.2003 17:17	REMOVE ACCOUNT	mcd	iinesankka	

Kuva 30: Tapahtumahistoria



## 9 Atlas-syötin

Atlas-syötin on järjestelmän ulkoinen osa, jonka avulla Atlas-muodossa olevat valokäyrät voidaan siirtää tietokantaan. Atlas-syötin käyttää tiedon syöttämiseen Tahiti-Apin tarjoamia palveluja ja kommunikoi Tahiti-Apin kanssa käyttämällä container-rajapintaa. Tässä luvussa kerrotaan, miten Atlas-syötin toimii, ja kuinka se on tarkoitus toteuttaa.

### 9.1 Toiminta

Atlas-syötin on komentoriviltä ajettava jäsenysohjelma. Käynnistettäessä Atlas-syöttimelle annetaan syötteenä käyttäjänimi ja salasana, sekä syötettävien Atlas-tiedostojen nimet. Syötin kutsuu Tahiti-Apin login-metodia käyttäjän antamalla käyttäjänimellä ja salasanalla. Mikäli sisäänkirjautuminen onnistuu, Atlas-syötin alkaa seuraavaksi jäsentää Atlas-tiedostoja. Jos sisäänkirjautuminen epäonnistuu, käyttäjän ruudulle tulostetaan Tahiti-Apin antama virheilmoitus. Atlas-syöttimen käyttöön tarvitaan siis vähintään syöttäjätason oikeudet.

Valokäyrän syöttövaiheessa Atlas-syötin luo LightcurveFormContainer-olioita Atlas-tiedostojen pohjalta määrittelydokumentissa esitettyjen vastaavuuksien perusteella. Atlas-syötin käy tiedostoja läpi ylhäältä alas muodostaen ja syöttäen valokäyriä järjestelmään yksi kerrallaan. Mikäli LightcurveFormContainerin pakollisia kenttiä ei pystytä täyttämään, Atlas-syötin tulostaa käyttäjän ruudulle ja luomaansa virhetiedostoon virheilmoituksen, jossa kerrotaan, mitä tiedostoa käsiteltäessä virhe tapahtui, monesko tiedoston valokäyrä oli käsiteltävänä, monennellako rivillä virhe tapahtui ja mitkä kentät puuttuivat. Atlas-syötin myöskin tarkistaa ennen valokäyrän syöttämistä, onko järjestelmään jo tallennettu identtinen valokäyrä. Tarkistus tehdään hakemalla valokäyriä Tahiti-Apin getLightcurves-metodilla ja vertaamalla sitä muodostettuun LightcurveFormContaineriin. Identtisyys tarkoittaa sitä, että kaikki kentät lukuunottamatta information-kenttää ovat samanlaiset. Tällaisen virheen sattuessa Atlas-syötin tulostaa ruudulle ja virhetiedostoon, mitä tiedostoa käsiteltäessä virhe tapahtui, monesko valokäyrä tiedostossa oli käsiteltävänä, käsiteltävän rivin numeron sekä ilmoituksen valokäyrien identtisyydestä.

### 9.2 Toteutus

Atlas-syötin koostuu yhdestä luokasta, joka sisältää main()-metodin, sekä sisäisestä valokäyrien jäsentäjämetodista. Kun Atlas-syötin ajetaan, se luo käyttöönsä ilmentymän Tahiti-Apista, ja kutsuu Tahiti-Apin login-metodia parametrina saamallaan käyttäjätunnuksella ja salasanalla. Kun yhteys on muodostettu, main-metodi jäsentää parametrinaan saamansa Atlas-tiedostot erillisiksi valokäyriksi ja kutsuu sisäistä parseLightcurve-metodia, joka jäsentää valokäyrän ja palauttaa LightcurveFormContainer- sekä AsteroidQueryContainer-luokat, tai null-arvot, mikäli valokäyrätiedot olivat puutteelliset. Tämän jälkeen main-metodi tarkastaa Tahiti-Apin getLightCurves-metodin avulla, antaen syötteenä muodostetun AsteroidQueryContainerin, onko järjestelmässä jo syötettävän valokäyrän kanssa identtinen valokäyrä. Mikäli ei ole, valokäyrä talletetaan järjestelmään käyttäen Tahiti-Apin insertLightcurve metodia.

## Lähteet

- Sun03 Sun Microsystems, <ftp://ftp.javasoft.com/docs/codeconv/CodeConventions.pdf>
- Tah03 Tahiti-ohjelmistotuotantoprojektiryhmä, määrittelydokumentti v.1.0 Final, <http://www.cs.helsinki.fi/group/tahiti/maarittelydokumentti10.pdf>. [4.3.2003]
- Kar00 Karttunen et al, Tähtitieteen perusteet, Tähtitieteellinen yhdistys Ursa, Vaasa, 2000