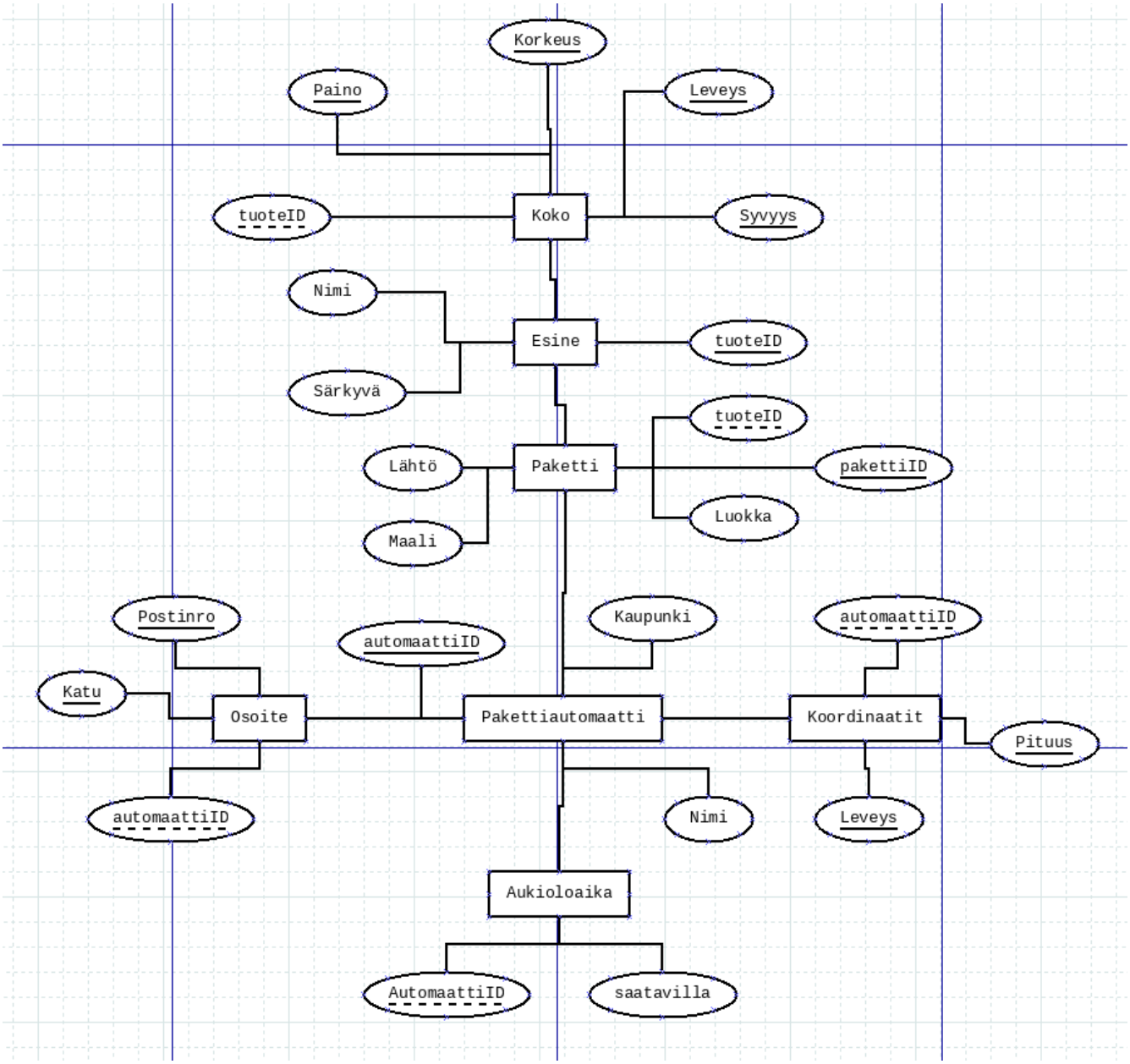
Harkkatyö, TYYLI-projekti
Camilla Piskonen, Late, 0451801

1. Tehtävänannon kuvaus ja työn rajaukset.
Tehtävänä oli suunnitella ja toteuttaa tietokanta yksinkertaista postijärjestelmää varten. Toteutin tietokannan SQLiteä käyttäen jolloin se ei oikein sovellu todelliseen yrityskäyttöön mutta se havainnollistaa hyvin, kuinka tietokanta voisi todellisuudessa toimia.

2. Tietokannan käsitemalli ja eheyssäännöt.
Tein tietokannasta ensin jonkinlaisen raakaversion, josta lähdin sitten miettimään mahdollisimman 3. normaalimuodon tietokantaa. Tietokantaan tuotteille, pakettiautomaateille sekä paketeille on jokaiselle oma uniikki ID josta ne voidaan tunnistaa päällekkäisyyksien estämiseksi. Kokotietoihin ja muihin pakollisiin olemassaolon kannalta oleviin tietoihin määrittelin check-lauseilla ja positiivisuusmääreillä eheysehtoja. Myöskään mikään ID ei saa esiintyä kuin kerran, joten se on määritelty uniikiksi.

Tietokannan ER-malli:



Postiautomaatti – Aukioloaika 1:1

Postiautomaatti – Koordinaatit 1:1

Postiautomaatti – Osoite 1:1

Postiautomaatti – Paketti 1:n

Paketti – Esine 1:1

Esine – Koko 1:1

3. Ohjelman kuvaus

Ohjelmassa on kaksi toiminnallisempaa ikkunaa ja kaksi infoikkunnaa. Käyttäjä hoitaa kaikki toiminnat graafisen käyttöliittymän kautta. Ohjelma hakee suurimman osan tiedostaan tietokannasta, joka on ohjelmoitu SQLitellä. Käyttäjä pystyy lisäämään ja poistamaan tietokannasta esineitä ja paketteja muttei automaattien tietoja.

Ohjelman ikkunat suunniteltiin Gluon-ohjelmalla ja tietokanta rakennettiin Database Browserilla. Kokonaisuudessaan ohjelma tehtiin NetBeans IDE 8.1.

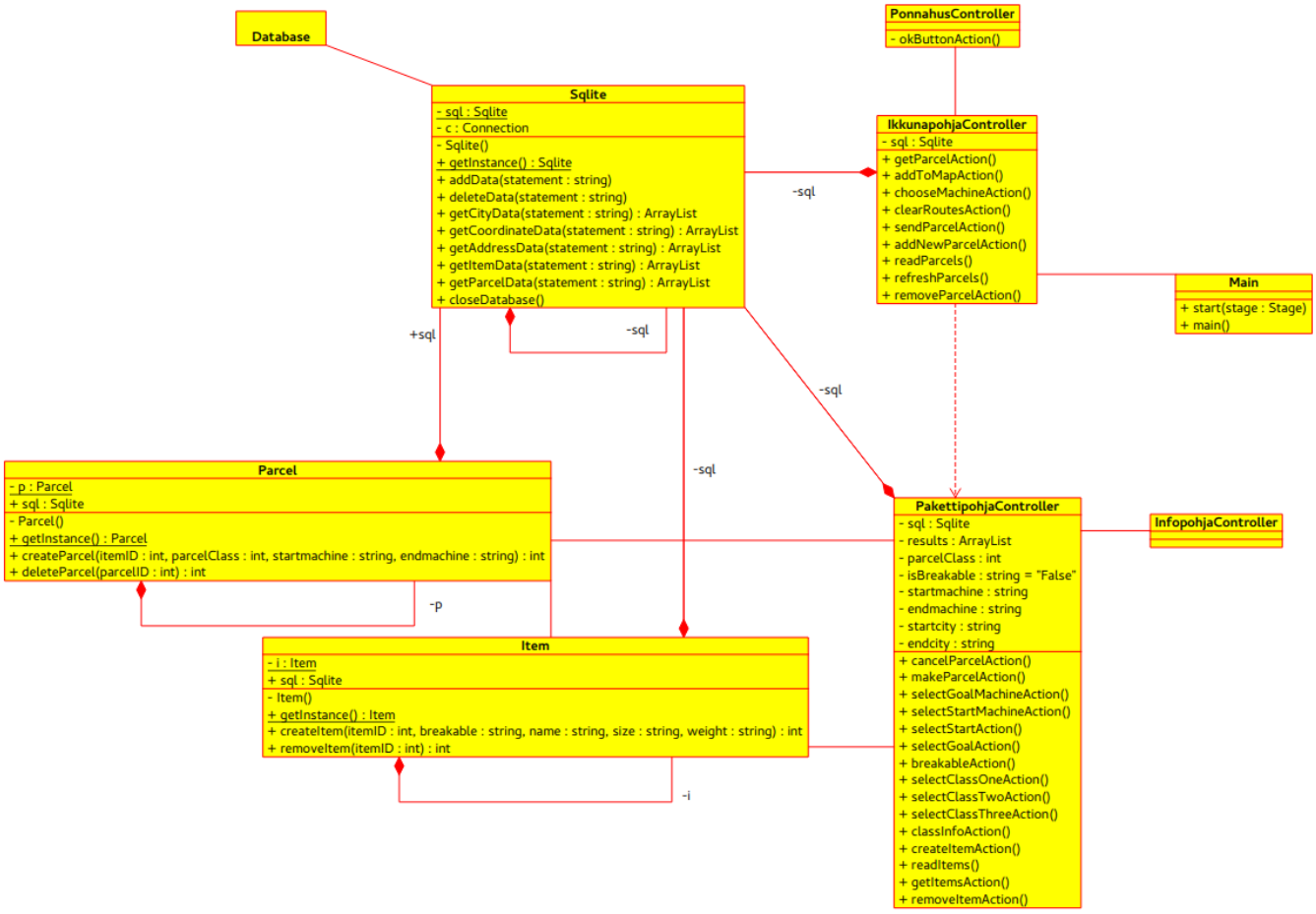
4. Raportti ohjelmallisesta toteutuksesta ja lista toiminnallisuuksista.

Ohjelma sisältää ikkunoiden controllerit sekä muutaman olion, joilla käsitellään tietokantaa, paketteja ja esineitä. Ohjelmaan on myös sisällytetty automaattien tietojen parsimista varten olio, mutta sitä käytettiin vain silloin, kun haluttiin lisätä automaattien tiedot ensimmäistä kertaa tietokantaan. Tietokannan eheyttä ylläpidetään sekä tietokannan check- ja not null -lauseilla että koodin puolella try-lohkoilla ja else-if rakenteilla. Virheenkäsittely pyrittiin pitämään jokaisen olion omana tehtävänä, paitsi jos olion täytyi palauttaa jotain arvoja kutsuvaan olioon. Tällöin olio palautti virhetilanteessa tietyn arvon ja kutsuvassa ohjelmassa verrattiin paluuarvoa ja tulostettiin arvoa vastaava virheilmoitus tai jatkettiin ohjelman toimintaa normaalisti.

Ohjelmassa on mahdollista lisätä kartalle automaatteja kaupunkien mukaan, lähettää paketteja, lisätä uusia paketteja, poistaa lähettämättömiä paketteja, lisätä uusia esineita, poistaa esineitä sekä poistaa kaikkien lähetettyjen pakettien reitit kartalta.

Tietokannan käsittely pyrittiin pitämään yhden olion vastuulla, jolloin muiden olioiden tarvitsi vain kutsua tietokanta-oliota tarpeen vaatiessa. Tämä selventää hieman muiden ohjelmien koodia kun ei tarvitse kirjoittaa samoja sql-toimintoja uudelleen jokaiseen metodiin vaan voidaan vain kirjoittaa esim. Sql.addData(statement).

5. Ohjelman täydellinen luokkakaavio



Laitan mallista vielä tiedostoversion mukaan.
Mallista nähdään että muut ohjelmat kutsuvat aina Sqlite-oliota kun haluavat yhteyden tietokantaan. Metodeja tuli todella paljon koska ikkunatoiminnallisuudet ovat kaikki aina yksi olio. Infoikkunoihin ei tullut käytännössä yhtään metodeja koska ne vain näyttävät jotain tietoa kutsuttaessa. Main lähinnä vain käynnistää ohjelman ja jos ohjelmassa olisi loppulokitoiminnallisuus, se kirjoittaisi sen. IkkunapohaController on lähinnä vastuussa kaikessa ohjelman toiminnallisuudesta.

6. Yhteenveto. Mitä ongelmia jouduttiin ratkomaan tai mitä erityistä havaittiin ja opittiin tietokannan toteutuksen yhteydessä.

Pääongelmina oli ehkä tietokannan ja Javan välisen keskustelun toteutus sekä esineiden ja pakettien toteutus. Sql:n yhteyteen tuli paljon toistuvia käskyjä mutta niitä ei oikein voinut yleistää jolloin toistaminen oli melkeinpä pakollista. Toistoa sai karsittua sillä, että keskitti kaikki Sql-toiminnot yhteen olioon jolloin muu koodi pysyi hieman siistimpänä.