

Aki Koppinen

Projektityö – Ruokala simulaattori

Ohjelmointiprojekti - Simulointiprojekti

Metropolia Ammattikorkeakoulu
Tieto- ja viestintätekniikka
Ohjelmistotuotanto
Ohjelmointiprojekti
18.10.2020

Sisällys

1	Johdanto	1
2	Visio	1
3	Käsitteet, määritelmät	1
4	Käsitteellinen malli	3
4.1	Tavoite	3
4.2	Syötteet	3
4.3	Tulosteet	3
4.4	Sisältö	4
4.5	Oletukset ja yksinkertaistukset	5
4.6	Mallin kuvaus	6
4.6.1	Komponenttilista	6
4.6.2	Prosessikaavio	6
5	Mallin ohjelmointitekhninen toteutus	7
5.1	Käytetyt ohjelmointikielet ja kirjastot (ulkoiset API:t).	7
5.2	Arkkitehtuuri	7
5.3	Käyttöliittymän kuvaus	11
5.4	Sisäisen logiikan kuvaus	15
5.5	Ulkoisten tietovarastojen (tiedostot, tietokannat) kuvaukset	16
5.6	Testaus	17
6	Simulaattorin käyttöohje	17
7	Tehdyt simulointikokeet	19
7.1	Koe 1	20
7.2	Koe 2	21
7.3	Koe 3	22
7.4	Koe 4	23
7.5	Koe 5	24
8	Jatkokehitys suunnitelmat	25
8.1	Syötteet	25

8.2	Tietokanta	26
8.3	Käyttöliittymä	26
9	Yhteenveto	27

Liitteet

Linkki simulaattorin lähdekoodiin SVN-versionhallinnassa

Linkki simulaattorin JavaDoc dokumentaatioon SVN-versionhallinnassa

1 Johdanto

Tämä dokumentti on laadittu osana Ohjelmointiprojekti kurssin Simulointiprojektia. Sen tarkoitus on auttaa lukijaa ymmärtämään toteutetun simulaation tarkoitus, käytötapa ja toteutustapa. Se toimii myös projektin dokumentaationa ja muistiinpanoina itse simulaattorin ohjelmoijalle.

2 Visio

Tavoitteena projektille oli tuottaa Java-kielellä ohjelmoitu simulaattori, joka simuloi kouluruokalan toimintaa. Ruokalassa asiakkaat jonottavat eri linjastoille, joista sitten siirtyvät kassalle ja edelleen ruokalaan ruokailemaan. Simuloinnilla pystytään seuraamaan asiakas ja -palvelupiste määrien vaikutusta prosessin toimintaan.

3 Käsitteet, määritelmät

Simulointi on reaali maailman tapahtuman jäljittelyä. Simulaatio on simuloinnin ilmentymä, tässä tapauksessa tietokoneella tuotettu ohjelmisto, joka jäljittelee oikean ruokalan toimintaa.

Asiakkaalla tarkoitetaan ruokalassa asioivaa henkilöä, hän voi olla oppilas, opettaja tai muu koulun ruokalan palveluita käyttävä henkilö.

Palvelupiste on yksittäinen osa simulaatio kokonaisuutta. Asiakkaat saapuvat jonottamaan palvelupisteisiin ja saavat niistä palvelua. Asiakkaat jatkavat palvelupisteestä toiseen palvelupisteeseen tai poistuvat simulaatiosta.

Palveluajalla tarkoitetaan aikaa, joka asiakkaalla kuluu palvelupisteessä palveltavana.

Tapahtuma on yksittäinen simuloitu asia, joka saa aikaan muiden asioiden käynnistymisen tai loppumisen simulaatiossa. Se voi olla saapuminen, poistuminen tietylle palvelupisteelle, tai poistuminen simulaatiosta.

Tapahtumalista on sisältää tapahtumia ja mahdollistaa niiden läpikäymisen tärkeys järjestyksessä. Tapahtumalistan avulla simuloidaan asiakkaiden reaaliaikaista saapumista ja poistumista palvelupisteiltä.

Jono on tapahtuma, missä asiakas on saapunut palvelupisteelle palveltavaksi, mutta joutuu odottamaan palvelun alkamista. Syynä tähän voi olla, että palvelupisteen kapasiteetti on sillä hetkellä täynnä.

Jakauma on mittaustuloksellista tietoa, jota simulaattorissa tuotetaan simulaation aikana saatavasta datasta.

Satunnaisuutta käytetään jäljittelemään reaali maailman satunnaisuutta. Satunnaisuutta simulaatiossa toteuttavat erilaiset valmiit satunnaisluku generaattorit.

Kello on simulaattorin oma itsenäinen aikajärjestelmä ja joka kasvattaa aikaa samalla kun simulaatio etenee.

Moottori on se ohjelman osa, joka vastaa simulaation toiminnallisuudesta. Sen kautta luodaan alkutapahtumat ja suoritetaan A, B ja C tapahtumia.

Generaattori on arvoja tuottava ohjelma tai ohjelman osa. Generaattori tuottaa arvoja siihen kirjoitetun koodin perusteella.

Kapasiteetti on palvelupisteille annettu maksimimäärä asiakkaita, joita se voi ottaa palveltavaksi yhtä aikaa. Tämän määrän yli menevät asiakkaat pistetään jonoon.

A-vaihe on yksi ohjelman toistamista simulaatio vaiheista. A vaihe suoritetaan aina toiston alussa ja siinä kello siirretään vastaamaan seuraavan tapahtuman aikaa.

B-vaihe on A vaihetta seuraava simulaation vaihe. Siinä käydään läpi olemassa olevaa tapahtumalistaa ja suoritetaan seuraavaksi vuorossa oleva tapahtuma. Tapahtumien suorittaminen generoi uusia toisen tyyppisiä tapahtumia.

C-vaihe seuraa B vaihetta tai toista C vaihetta. Sitä suoritetaan niin pitkään, kunnes C vaiheita ei enää voida suorittaa. C vaihe tarkistaa onko palvelupisteiden jonossa asiakkaita ja onko kyseisessä palvelupisteessä kapasiteettia tälle asiakkaalle. Jos on, asiakas siirretään palveltavaksi.

MVC on ohjelmistoarkkitehtuuri, jonka tarkoituksena on jakaa ohjelma kolmeen osaan: malliin, näkymään ja käsittelijään/ohjaimeen.

DAO eli Data Access Object-suunnittelumalli kuvaa tietokannan ja ohjelmiston keskinäisen vuorovaikutuksen toteutustapaa. Sen tavoitteena on rajata kaikki tietokantaan liittyvät operaatiot tiettyihin luokkiin.

4 Käsitteellinen malli

4.1 Tavoite

Ruokalasimulaattorin tavoitteena on simuloida ruokalan toimintaa tuomalla simulaatioon asiakkaita ja seuraamalla heidän toimintaansa, kun he etenevät järjestelmän läpi. Simulaatio kerää dataa asiakkaiden toiminnasta ja tällä informaatiolla saadaan selville ruokalan toimintamahdollisuudet tietyille asiakasmäärille, ja tuloksia voidaan pyrkiä parantamaan muuttamalla simulaation lähtötietoja.

4.2 Syötteet

Muutettavia lähtötietoja ovat asiakkaiden määrät ja palvelupisteiden määrät sekä palvelupisteiden kapasiteetit. Simulaatiolle voidaan myös asettaa kesto, jonka tultua täyteen simulaation suoritus lakkaa.

4.3 Tulosteet

Simulaatio kerää simulaation aikana dataa suorituskykyksuureiden muodossa, ja laskee niistä myös uusia suureita simulaation loputtua. Kaikki suureet esitetään käyttäjälle simulaation päätyttyä. Suureiden perusteella voi tehdä päätelmiä simulaation tehostamisesta, ja ajaa simulaatio läpi uusilla lähtötiedoilla eroavaisuuksien huomaamiseksi. Tämän simulaation keräämät ja laskemat suureet on esitetty alla.

A: Saapuvien asiakkaiden kokonaismäärä

C: Palveltujen asiakkaiden määrä. Erotus A:han kertoo, kuinka paljon asiakkaita jäi palvelematta.

B: Palvelupisteen aktiivinen aika, eli ajat yhteensä milloin palvelupiste on palvellut asiakkaita. Aktiivisen ajan ulkopuolelle jäävänä aikana palvelupiste on jouten.

T: simuloinnin kokonaisaika, eli milloin simulointi todellisuudessa loppui.

U: palvelupisteen käyttöaste. Jos käyttöaste on turhan pieni ei palvelupistettä välttämättä tarvita, jos sille on toinen vaihtoehto. Jos taas liian iso, on palvelupiste luultavasti ruuhkautunut. Lasketaan yhtälöstä B/T .

X: suoritusteho. Kertoo palvelupisteelle saapuvien asiakkaiden määrän suhteessa aikaan. Lasketaan yhtälöstä C/T .

S: palvelupisteen keskimääräinen palveluaika. Lasketaan yhtälöstä B/C .

Ri: Palvelupisteen läpimeno aika yhdelle asiakkaalle (sisältää jonotusaika + palveluaika), vaatii ajanottoa, kun asiakas saapuu ja poistuu.

W: kaikkien asiakkaiden läpimeno aikojen summa (kaikkien R_i aikojen summa).

R: Keskimääräinen läpimeno aika. Saadaan selville kuinka kauan yksittäinen asiakas keskimäärin viettää simulaatiossa. Lasketaan yhtälöstä W/C .

N: Keskimääräinen jonon pituus. Saadaan selville, kuinka pitkään ruokalaan saapuva asiakas joutuu keskimäärin odottamaan. Lasketaan yhtälöstä W/T .

4.4 Sisältö

Reaalimaailman ruokaloista on yleensä käytössä erilaisia linjastoja, joilla monesti tarjotaan erilaisia ruokavaihtoehtoja. Asiakkaat aloittavat näin ollen ruokalakäyntinsä valitsemalla mieluisen ruuan, ja asettumalla sitten haluamaansa jonoon. Simulaation kannalta tämä toimii helposti, sillä asiakkaat voidaan jakaa linjastoille oletetun käyttäytymismallin mukaan.

Linjastot mahdollistavat reaalimaailmassa useiden asiakkaiden liikkumisen palvelupisteessä yhtä aikaa, joten linjastot voisivat hyötyä siitä, että niillä olisi oma kapasiteetti.

Reaalimaailman ruokalan kassojen määrä on yleensä suhteessa asiakkaiden määrään, jota palvellaan. Kassoja ei kuitenkaan yleensä ole mahdollista avata paria kassaa enemmän. Tässä simulaatiossa kassojen määrää ei säädellä ajon aikana, vaan sitä voi säätää simulaation alkutiedoissa.

Monien reaali maailman ruokaloiden yhteydessä toimii oma kahvila, jolla on oma linjasto ja samassa yhteydessä myös oma kassa. Kahvilan kassa ja ruokalan kassat eivät yleensä anna maksaa toistensa hyödykkeitä, joten sitä ei sallita myöskään simulaatiossa. Kahvila ja ruokala kuitenkin yleensä jakavat ruokala tilan, jossa asiakkaat nauttivat ruuan ja/tai juoman. Ruokalalle annetaan näin ollen kapasiteetti, ja kapasiteetin täytyessä asiakkaat joutuvat jonottamaan syömään pääsemistä.

4.5 Oletukset ja yksinkertaistukset

Simulaation toiminta ei perustu oikeille luvuille, joita olisi kerätty olemassa olevilta ruokaloilta. Sen sijaan kaikki luvut, kuten asiakkaiden määrä, palveluajat, jonotusajat ja kapasiteetit, ovat pelkkiä arvioita. Asiakasmäärän ja saapumisaikojen satunnaisuutta simuloidaan käyttämällä satunnaislukuja, joille kuitenkin asetetaan näennäiset ala ja ylärajat. Nämä ala- ja ylärajat varmistavat, ettei simulaatio tuota täysin mahdottomia tuloksia.

Myös asiakkaiden käyttäytymistä kuvataan oletetuilla luvuilla. Simulaatiossa annetaan oletuksena että 20 % ihmisistä valitsee kahvilan palvelut, 30 % kasvisruuan ja 10 % erikoisruuan. Loput 40 % valitsevat perusruuan. Kasvisruoka, erikoisruoka ja perusruoka ovat vain oletuksia linjastojen tarjonnalle. Simulaatiossa linjastoja ei ole nimetty, vaan niitä kutsutaan nimillä Linjasto 1, Linjasto 2 ja Linjasto 3. Prosenttiosuudet mukailevat silti edellä mainittuja. Linjasto 1 vastaa perusruokaa, Linjasto 2 kasvisruokaa ja Linjasto 3 erikoisruokaa.

Simulaatio saa odotusarvoiset palveluajat, jotka kuitenkin vaihtelevat satunnaisesti asiakkaan mukaan. Linjastolle 1 odotusarvo on 5, Linjasto 2 odotusarvo on 5, Linjasto 3 odotusarvo on 12, Kahvila odotusarvo on 4, Kassa 1 odotusarvo on 3, Kassa 2 odotusarvo on 3 ja Ruokalan odotusarvo on 20.

Reaali maailmassa kahvila ja ruokala saattaisivat omata eri aukiolo aikoja, mutta tässä simulaattorissa sitä mahdollisuutta ei käyttäjälle tarjota, vaan aloitusaika ja loppuaika on molemmille palveluille sama.

Reaali maailmassa linjasto on usein omatoimista palvelua, ja hidas asiakas pystyy näin hidastamaan palvelua myös muille linjastossa. Tätä ei kuitenkaan oteta suoranaisesti simulaatiossa huomioon.

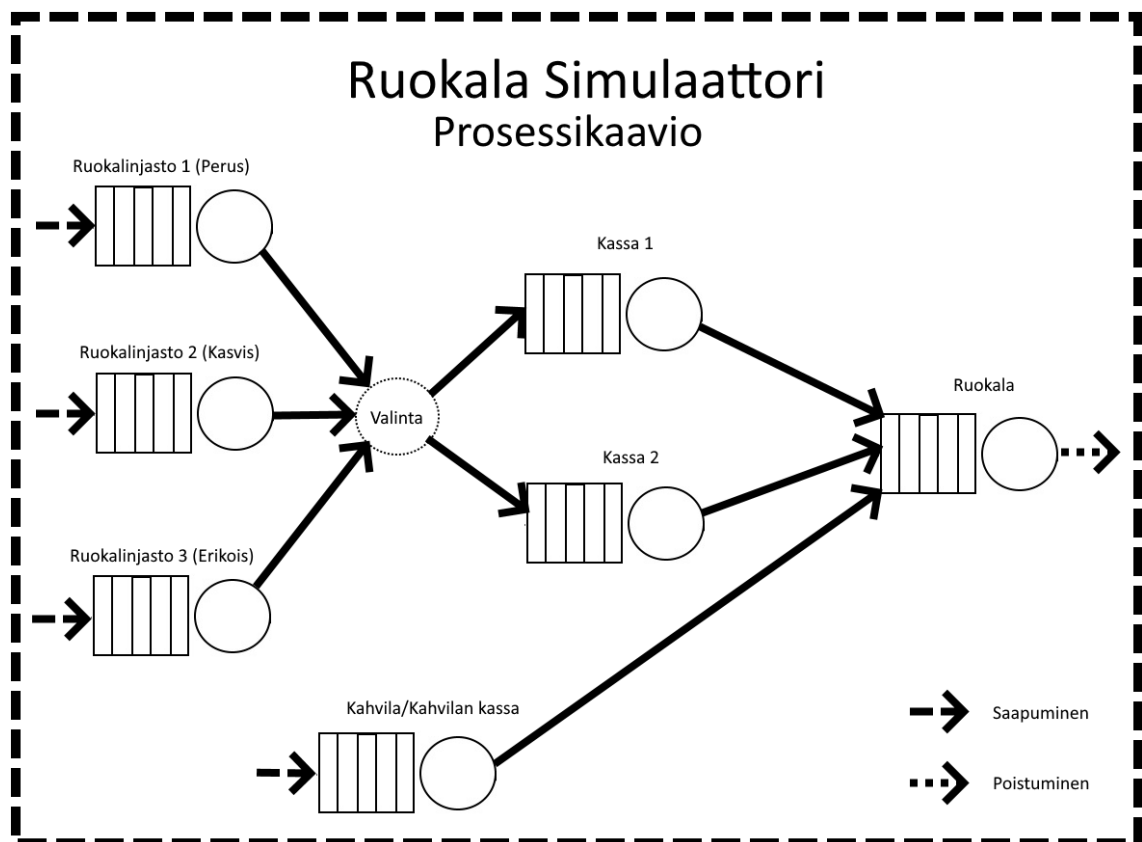
4.6 Mallin kuvaus

Simulaatiossa yritetään jäljitellä reaali maailman ruokalan toimintaa käyttäen hyväksi todellisista ruokaloista asiakkaana tehtyjä huomioita. Reaali maailman ruokaloissa on kuitenkin runsaasti variaatiota, eikä tämä simulaatio jäljittele täysin mitään tiettyä ruokalaa. Simulaation toimintamalli on valittu yhdistelemällä simulaation tekijän omia kokemuksia Metropolian Leiritie ja Myllypuron toimipisteistä, sekä UniCafen ravintoloista.

4.6.1 Komponenttilista

Komponentti	Yksityiskohtia
Asiakas	saapumisväliaikojen jakauma
Palvelupiste	palveluajan jakauma
Ruokalinjastot	alku jonotus, kassan valinta
Kahvila	alku jonotus, oma kassa
Ruokala	Kapasiteetin riittävyys, poistuminen

4.6.2 Prosessikaavio



Kuva 1. Ruokala-simulaattorin prosessikaavio

Prosessikaaviosta (*Kuva 1*) voidaan nähdä asiakkaan kulkureitti simulaatiossa. Asiakas saapuu joko linjastolle 1,2,3 tai Kahvilaan. Jos asiakas saapui linjastojen palveltavaksi, hänen tulee tehdä valinta, kummalle kassalle hän päätyy. Simulaatiossa asiakas valitsee aina kassan, jossa on lyhyempi jono. Kassojen jälkeen asiakas jatkaa Ruokalaan ruokailemaan. Jos asiakas taas aloitti palvelunsa kahvilasta, jatkaa hän suoraan ruokalaan, jossa hän jakaa ruokalan kapasiteetin linjastojen kassojen kautta tulleiden asiakkaiden kanssa. Kun ruokalan palvelu päättyy, eli asiakas on lopettanut syömisen, hän poistuu simulaatiosta.

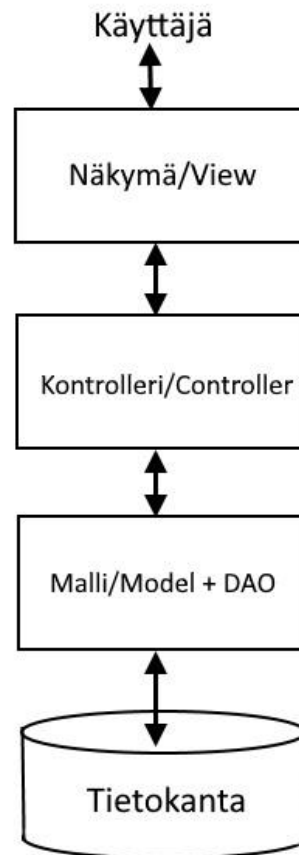
5 Mallin ohjelmointitekkinen toteutus

5.1 Käytetyt ohjelmointikielet ja kirjastot (ulkoiset API:t).

Simulaatio on toteutettu kokonaisuudessaan Java-ohjelmointikielellä. Käytetty versio JDK:sta on 1.8.0_261. Ohjelmisto käyttää ulkoisia kirjastoja apunaan tietokannan toteuttamisen ja satunnaisluku jakaumien laskemisen yhteydessä. Tietokanta on toteutettu Hibernate ORM -työkalun avulla, ja satunnaisluku jakaumat eduni.distributions paketin sisältämien Java luokkien avulla. Käyttöliittymän ja visualisoinnin tuottamisessa ohjelmisto käyttää apunaan JavaFX komponentteja, jotka kuitenkin kuuluvat käytettyyn JDK versioon.

5.2 Arkkitehtuuri

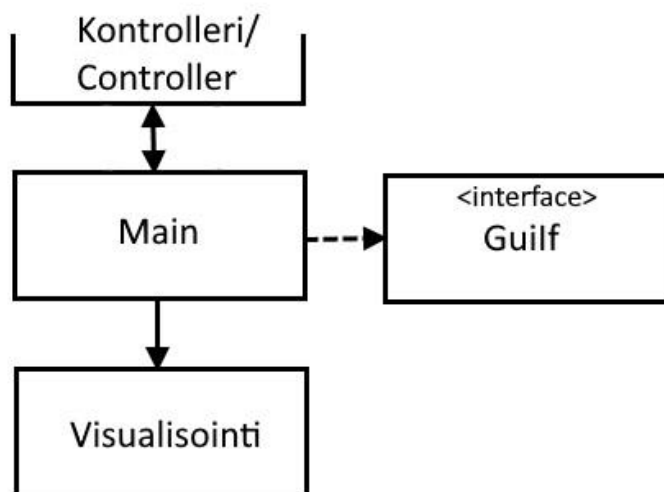
Alla olevissa kuvissa on esitetty korkean tason komponentit ja niiden väliset yhteydet.

MVC - YLEISARKKITEHTUURI

Kuva 2. Ohjelmistossa toteutettu MVC arkkitehtuuri

Ohjelmiston yleisarkkitehtuuri (*Kuva 2*) on kasattu MVC mallin mukaisesti. Näkymä määrittää käyttöliittymän ulkoasun ja tietojen näytön esityksen käyttöliittymässä. Malli kuvaa järjestelmän tiedon tallentamisen, ylläpidon ja käsittelyn. Tässä tapauksessa malliin on myös sisällytetty DAO suunnittelumallia mukaillen omat luokat tietokantatoiminnoille. Käsittelijä/ohjain vastaanottaa käyttäjältä tulevat käskyt sekä muuttaa mallia ja näkymää vastauksena niihin.

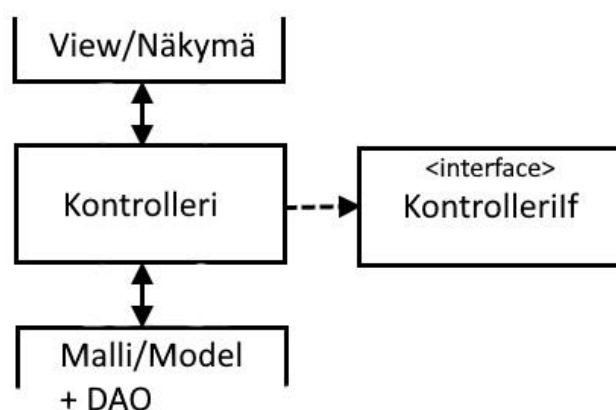
Näkymä/View



Kuva 3. Näkymän sisältämät luokat ja yhteydet

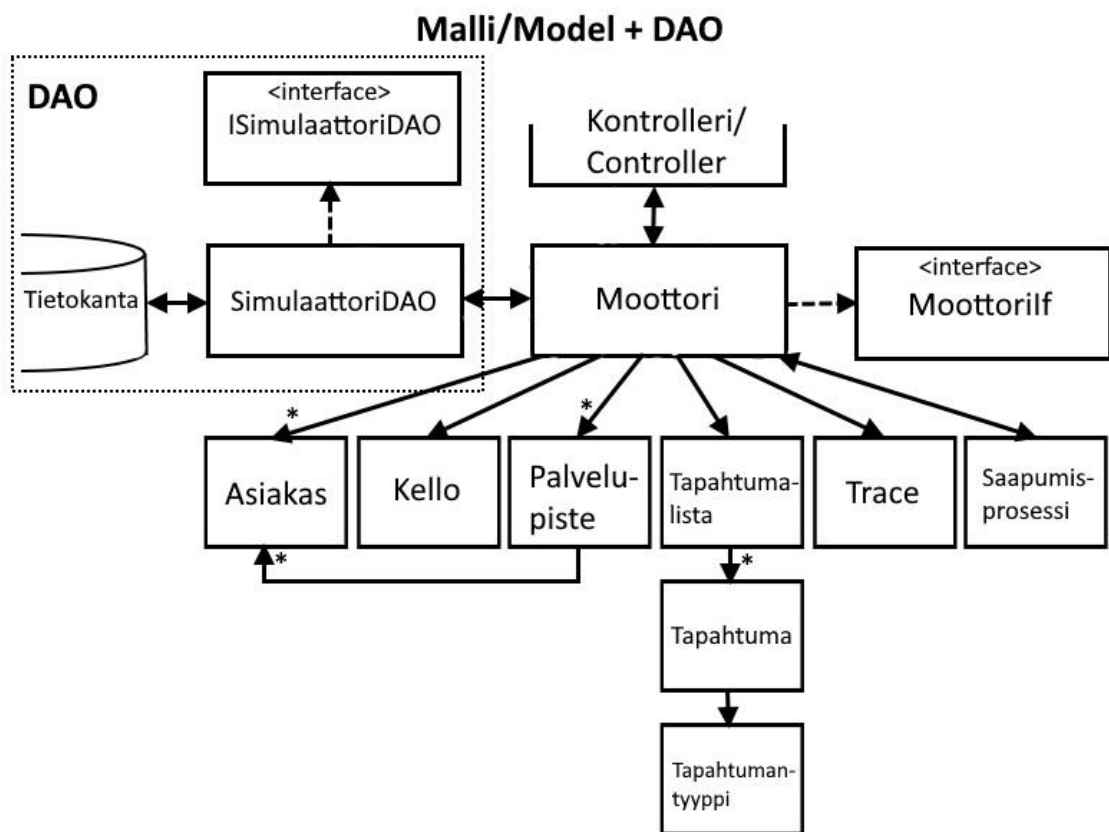
Näkymän (*Kuva 3*) pääluokkana on Main, joka toteuttaa GuIf rajapinnan. Näin ollen Main luokkaa voidaan kutsua GuIf rajapinnan kautta. Main luokka käyttää Visualisointi-luokkaa toteuttamaan simuloinnin graafisen esityksen ja on yhteydessä muuhun koodin osiin Kontrollerin kautta. Ohjelmiston käynnistyminen myös tapahtuu Main luokan kautta.

Kontrolleri/Controller



Kuva 4. Kontrollerin sisältämät luokat ja yhteydet

Kontrolleri (*Kuva 4*) luokka välittää tietoja Näkymän Main-luokan ja Mallin Moottori-luokan välillä. Näin toteutuu MVC mallin mukainen toimintojen eriyttäminen. Kontrolleri toteuttaa KontrolleriIf rajapintaa, jonka kautta Kontrolleria voi kutsua.

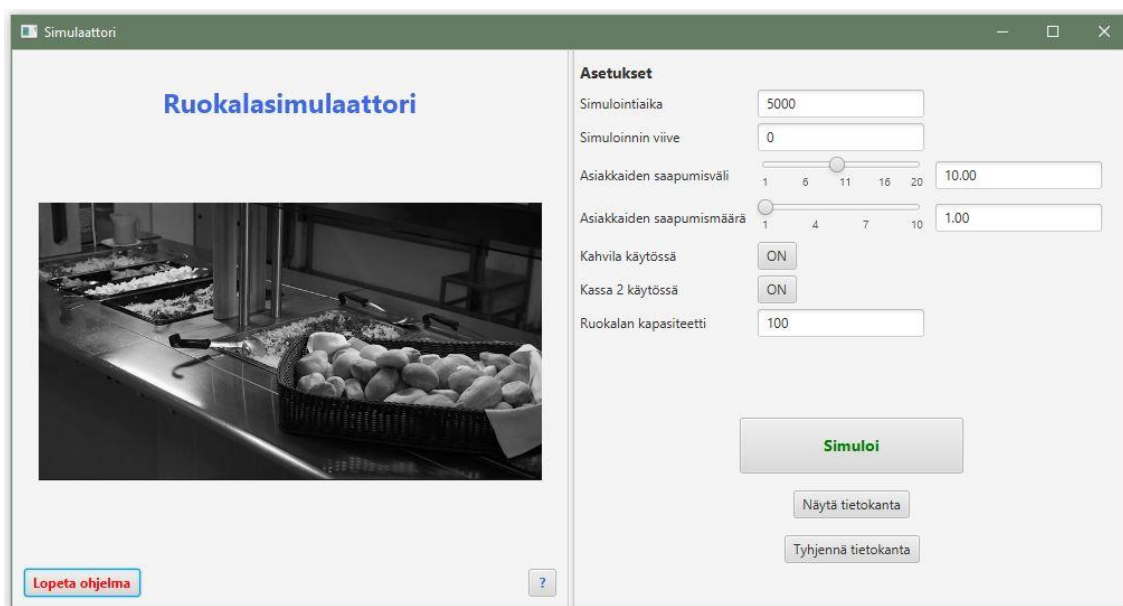


Kuva 5. Mallin sisältämät luokat ja yhteydet

Malli osion pääluokkana on Moottori (*Kuva 5*), joka toteuttaa lähes kaiken ohjelmiston tiedon käsittelyn, vain tietokantaan liittyvä toiminnallisuus on eriytetty omaan luokkaansa nimeltä SimulaattoriDAO. Sekä Moottori että SimulaattoriDAO toteuttavat omaa rajapintaansa ja näin ollen niitä voidaan kutsua rajapinnan kautta. SimulaattoriDAO luokan toimintoja käytetään vain Moottori luokan kautta ja toiminnot ovat joko tietokannasta lukeminen, lisääminen tai poistaminen. Koska toiminnot ovat melko yksinkertaisia, ei DAO suunnittelua ole eriytetty omaksi kokonaisuudekseen käytetyssä MVC mallissa.

5.3 Käyttöliittymän kuvaus

Alla olevissa kuvissa ja niiden selitteissä on esitelty ohjelman käyttöliittymä. Käyttöliittymä on yritetty rakentaa sisällöltään mahdollisimman siistiksi ja käyttäjälle helposti ymmärrettäväksi. Tyyliiltään käyttöliittymä on pidetty harmaa sävyisenä ja pelkistettynä, ja tällä korostetaan sitä, että simulaattori on tarkoitettu pääasiassa työkaluksi. Värejä on lisätty vain pienissä määrin nappeihin ja selitteisiin. Tässäkin tapauksessa värit viestivät vain toiminnallisesta tärkeydestä. Simulaation visualisoinnissa on käytetty värejä enemmän, jotta voitaisiin tuoda niillä esiin palvelupisteiden erilaisuudet ja kytkökset.

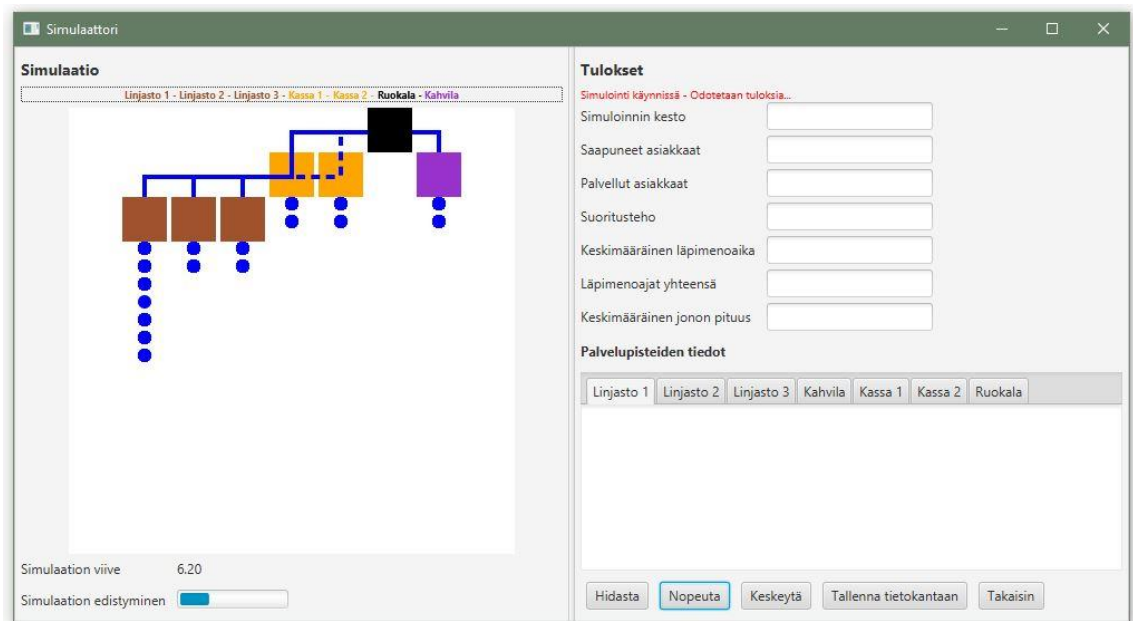


Kuva 6. Käyttöliittymän aloitusnäkömä

Ohjelman käynnistyessä aloitusnäkömä (*Kuva 6*) on ensimmäinen käyttäjälle avautuva ikkuna. Ikkuna on jaettu sivuttain kahteen osaan, jossa vasen puoli antaa visuaalista ilmettä ohjelmalle ja tarjoaa lopetus painikkeen, joka sulkee ohjelman, sekä info painikkeen, josta painettaessa aukeaa Tietoa ohjelmasta infolaatikko (*Kuva 10*).

Aloitusnäkömän oikea puoli "Asetukset" antaa käyttäjälle mahdollisuuden muuttaa simulaation toimintaa antamalla haluttuja syötteitä. Syöte laatikoille on annettu minimi ja maksimiarvot sekä haluttu syötteen tyyppi. Jos syötteet ovat näistä poikkeavia, ohjelma näyttää virheilmoitus ikkunan. Painamalla "Simuloi" -painiketta, simulointi käynnistyy annetuilla syötteillä ja näkömä vaihtuu simulaationäkömään (*Kuva 7*).

”Näytä tietokanta” -painike näyttää tietokannassa, sillä hetkellä olevan sisällön (Kuva 9). ”Tyhjennä tietokanta” -painike poistaa kaiken datan olemassa olevasta tietokannasta.



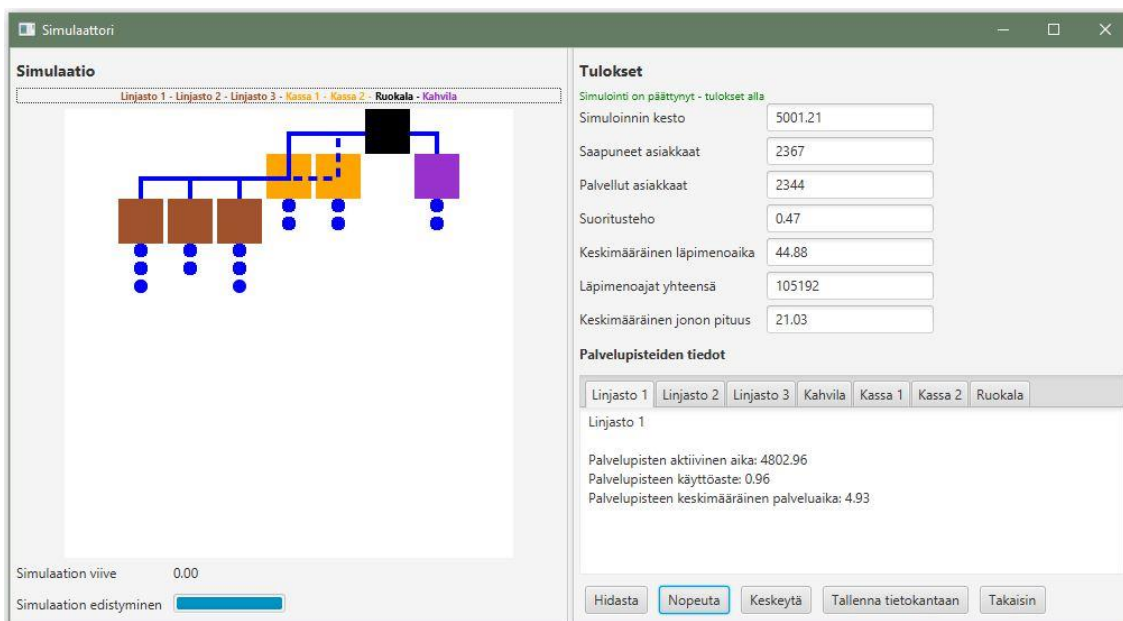
Kuva 7. Käyttöliittymän simulaationäkymä simulaation ollessa käynnissä

Simulaationäkymän vasen puoli esittää simuloinnin graafisen esityksen (Kuva 7). Siinä väritetyt laatikot edustavat eri palvelupisteitä ja siniset pallot palvelupisteelle jonottavia asiakkaita. Ruskeitten laatikoiden eli linjastojen läpi asiakkaat jatkavat oransseihin laatikoihin eli kassoilta. Kassoilta asiakkaat jatkavat mustaan laatikkoon eli ruokalaan, joka on simuloinnin päätepiste. Kun ruokalan palvelu päättyy, asiakas poistuu järjestelmästä. Purppura laatikko on kahvila joka omaa oman kassan ja siksi asiakkaat siirtyvät kahvilasta suoraan ruokalaan.

Jonottavien asiakkaiden määrä vaihtelee annettujen syötteiden perusteella. Suuret saapumismäärät ja pienet saapumisvälit lisäävät jonoja. Suuri kapasiteetti voi aiheuttaa sen, ettei jonoja pääse syntymään. Jos palvelupisteitä on otettu asetuksissa pois käytöstä, voivat jonot myös sen takia helposti kasvaa.

Simulaationäkymän vasemmassa nurkassa on esitetty ”Simulaation viive”, joka kertoo sen, kuinka nopeasti simulaatio edistyy. Pieni viive tarkoittaa suurempaa nopeutta. Edistymisestä kertoo viiveen alapuolella oleva ”Simulaation edistyminen” palkki.

Simulaation oikealla puolella on napit ”Hidasta” ja ”Nopeuta” jotka kasvattavat tai vähentävät viiveen määrää. ”Keskeytä” -nappi lopettaa meneillään olevan simulaation.

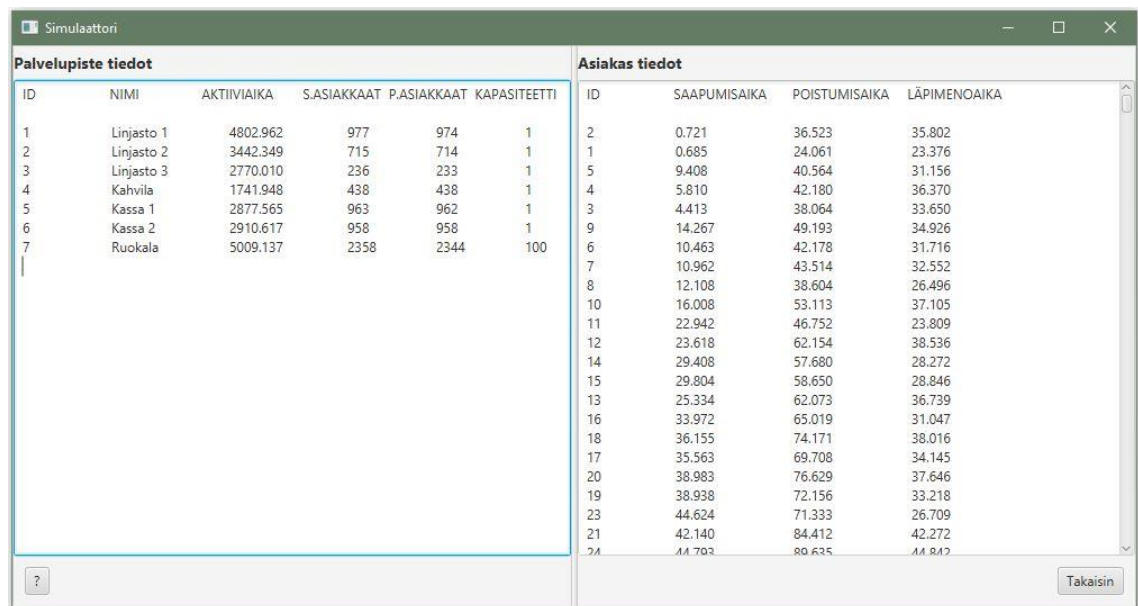


Kuva 8. Käyttöliittymän simulaationäkymä simulaation päätyttyä

Kun simulaatio on ajanut itsensä loppuun tai simulaatio on keskeytetty ”Keskeytä”-napilla, simulaatiosta kerätyt tulokset esitetään käyttäjälle simulaationäkymän oikeanpuoleisessa näkymässä (*Kuva 8*). Myös ”simulointi käynnissä” -teksti muuttuu punaisesta vihreäksi ja kertoo että simulointi on päättynyt.

Kaikkia palvelupisteitä koskevat tiedot ja niistä kerätyt yhteiset keskimääräiset tiedot esitetään yksittäisissä tietueissa kuten Simuloinnin kesto ja Keskimääräinen jonon pituus. Palvelupisteille ominaiset tiedot esitetään nimetyissä sarake tekstikentissä. Näistä sarakkeista valitsemalla pääsee näkemään tietyn palvelupisteen tiedot.

Simulaationäkymä tarjoaa myös mahdollisuuden tallettaa saadut tulokset mySQL tietokantaan. Tämä onnistuu painamalla ”Tallenna tietokantaan” -painiketta. Tietokantaan tallentuu kunkin palvelupisteen ja jokaisen asiakkaan tiedot simuloinnin ajalta. Painamalla ”Takaisin” -painiketta, käyttäjä pääsee takaisin aloitusnäkömään (*Kuva 6*) ja voi esimerkiksi muuttaa simulaation syöte asetuksia ja ajaa simulaation uudestaan.

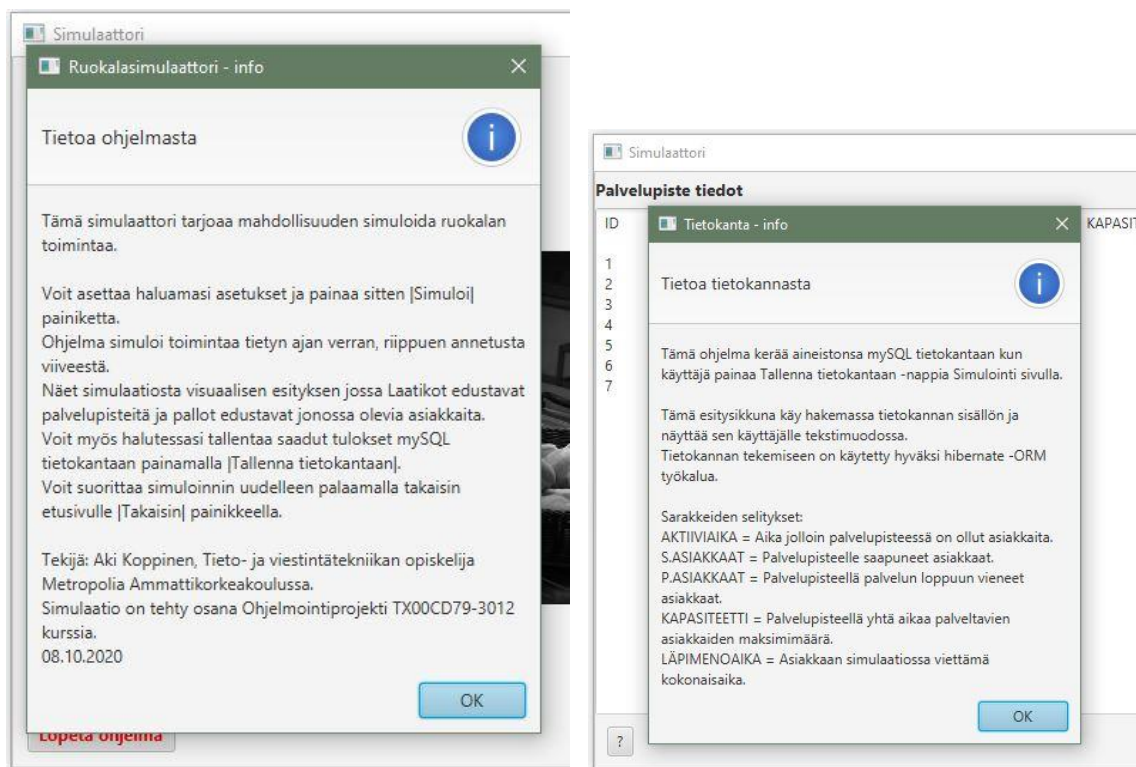


The screenshot shows a software window titled 'Simulaattori'. It contains two side-by-side tables. The left table, 'Palvelupiste tiedot', lists service points with columns for ID, Name, Active Time, S-Customer Count, P-Customer Count, and Capacity. The right table, 'Asiakas tiedot', lists customers with columns for ID, Arrival Time, Departure Time, and Waiting Time. Both tables have a vertical scrollbar on the right. At the bottom left is a help icon (?) and at the bottom right is a 'Takaisin' button.

Palvelupiste tiedot						Asiakas tiedot			
ID	NIMI	AKTIIVIAIKA	S-ASIAKKAAT	P-ASIAKKAAT	KAPASITEETTI	ID	SAAPUMISAIKA	POISTUMISAIKA	LÄPIMENOAIKA
1	Linjasto 1	4802.962	977	974	1	2	0.721	36.523	35.802
2	Linjasto 2	3442.349	715	714	1	1	0.685	24.061	23.376
3	Linjasto 3	2770.010	236	233	1	5	9.408	40.564	31.156
4	Kahvila	1741.948	438	438	1	4	5.810	42.180	36.370
5	Kassa 1	2877.565	963	962	1	3	4.413	38.064	33.650
6	Kassa 2	2910.617	958	958	1	9	14.267	49.193	34.926
7	Ruokala	5009.137	2358	2344	100	6	10.463	42.178	31.716
						7	10.962	43.514	32.552
						8	12.108	38.604	26.496
						10	16.008	53.113	37.105
						11	22.942	46.752	23.809
						12	23.618	62.154	38.536
						14	29.408	57.680	28.272
						15	29.804	58.650	28.846
						13	25.334	62.073	36.739
						16	33.972	65.019	31.047
						18	36.155	74.171	38.016
						17	35.563	69.708	34.145
						20	38.983	76.629	37.646
						19	38.938	72.156	33.218
						23	44.624	71.333	26.709
						21	42.140	84.412	42.272
						24	44.703	89.635	44.842

Kuva 9. Käyttöliittymän tietokanta näkymä

Painamalla ”Näytä tietokanta” -painiketta aloitussivulla (Kuva 6), ohjelma hakee tietokannasta sinne talletetut tiedot ja näyttää ne tekstimuodossa tietokantanäkymässä (Kuva 9). Tietokanta näkymässä näkyy vasemmalla jokaisen palvelupisteen ID, Nimi ja muut tiedot. Oikealla näkyy jokaisen asiakkaan ID sekä erilaiset simulaation asiakkaalle kirjaamat kellonajat simulaation edetessä. Tietokantanäkymässä voi painaa joko ”info” -painiketta, joka avaa ”Tietoa tietokannasta” infolaatikon (Kuva 11), tai sitten ”Takaisin” -painiketta, joka vaihtaa näkymän takaisin aloitusnäkyymään (Kuva 6).



Kuva 10. (vasen) Tietoa ohjelmasta – info laatikko

Kuva 11. (oikea) Tietoa tietokannasta – info laatikko

Info laatikot luotiin auttamaan käyttäjää ohjelman käytössä kertomalla nappien toiminnosta sekä avaamalla lyhenteiden merkitystä. *Kuva 10* kertoo myös simulaation tarkoituksen ja tekijän. *Kuvassa 11* on myös kerrottu hieman tietokannan toiminnallisuudesta.

5.4 Sisäisen logiikan kuvaus

Kun käyttäjä painaa "Simuloi" painiketta, simuloinnille asetetut asetukset viedään MVC mallin mukaisesti näkymän kautta kontrolleriin, ja kontrolleri luo model osiossa olevan Moottori luokan ilmentymän ja asettaa tälle näkymän kautta saadut aloitusarvot. Sitten kontrolleri kutsuu Moottori-luokan run-metodia ja käynnistää tälle samalla oman säikeen. Säikeen kautta tapahtuva ohjelman ajaminen mahdollistaa sen, että käyttäjä voi ohjelman edistyessä vaikuttaa sen toimintaan.

Moottorin sisällä simulaation toiminta alkaa alustamalla vaaditut palvelupisteet sekä tapahtumalistan. Alustuksen jälkeen se toistaa A, B ja C vaiheita, kunnes listatut tapahtumat loppuvat, tai ohjelma kohtaa ennalta määrätyn lopetusajan.

A vaihe asettaa kellonajan vastaamaan nykyistä ja B vaihe suorittaa tapahtumalistalta kaikki A vaiheessa asetettua kellonaikaa vastaavat tapahtumat. Tapahtuma voi olla asiakkaan saapuminen palvelupisteelle tai asiakkaan poistuminen palvelupisteeltä. Saapumiset generoivat uusia saapumisia tapahtumalistalle ja poistumiset generoivat siirtymisen seuraavalle palvelupisteelle.

Jos poistuminen tapahtuu simulaation viimeisellä palvelupisteellä, suoritetaan siirtymisen sijaan kuitenkin palvelun lopettaminen. Poistumista saattaa myös seurata valintatilanne, jos siirtyminen on mahdollista useampaan kuin yhteen palvelupisteeseen. Simulaatiossa näin tapahtuu kassanvalinnassa. Toiminnallisesti siirtyminen tapahtuu tällöin aina sille kassa -palvelupisteelle, jossa on pienempi jono. Jos jono on yhtä suuri, siirtyminen on satunnainen.

Kun suoritettavia B tapahtumia ei enää ole, siirrytään suorittamaan C tapahtumia. C tapahtumassa suoritetaan varsinainen palvelu palvelupisteissä. Palvelua ei kuitenkaan voida aloittaa, ellei palveltavia ole jonossa, tai heitä ei mahdu palveltavaksi. Koska joillakin palvelupisteillä on kapasiteetti ja ne voivat palvella samanaikaisesti useita, pitää C vaihetta toistaa niin kauan, kunnes kapasiteettia ei ole enää tarjolla.

Näitä kolmea vaihetta toistetaan Moottorin run-metodissa niin pitkään kunnes simuloinnille asetettu loppuaika saavutetaan, tai "Keskeytä" painike kertoo, että simulointi tulee lopettaa. Run metodin päätteeksi moottori vie tulostiedot kontrollerin kautta näkymälle ja ne voidaan esittää käyttäjälle.

5.5 Ulkoisten tietovarastojen (tiedostot, tietokannat) kuvaukset

Tietokannan toiminta on tehty Hibernate ORM työkalun avulla. Hibernate kääntää tietoa oliomuodosta relaatiomuotoon tallentaessaan tietoja MySQL tietokantaan, ja relaatiomuodosta olio muotoon tuodessa tietoja relaatiokannasta ohjelman luettavaksi. Tietokantaan liittyvä toiminta on sisällytetty model-kansioon MVC mallissa nimellä SimulaattoriDAO. Se saa kuitenkin oman rajapinnan (ISimulaattoriDAO), jonka kautta se keskustele model kansiossa olevan Moottori luokan kanssa.

Huomioitavaa on, että tietokanta on olemassa ohjelmasta erillään, eivätkä tietokanta toiminnot toimi, ellei tietokantaa ole luotu tietokoneelle lokaalisti tai salasana ja käyttäjänimeä ei ole vaihdettu hibernate.cfg.xml dokumenttiin. Dokumentissa tulee myös olla "validate" komennon sijasta "create" jos kyseessä on uusi tietokanta. Dokumentti löytyy ohjelman src kansioista.

Ohjelma luo tapahtumia käyttäen satunnaisluku generaattoreita Edinburghin yliopiston tekemästä "eduni.distributions" pakkauksesta. Pakkaus löytyy ohjelmiston src -kansioista. Nämä pakkauksen jakaumat saavat syötteenä arvoja, jotka määrittelevät satunnaislukujen suuruuden.

Ohjelmiston Javadoc dokumentaatio on löydettävissä ohjelmiston projektikansion (OhPro_Simulaattori_Final_w_JavaDoc) alikansioista nimeltä doc. Täältä pääsee selaamaan index.html tiedoston kautta Javadoc dokumentaatiota, joka selittää enemmän luokkien, muuttujien ja metodien tarkoitusta ja toimintaa. Linkki suoraan index.html tiedostoon SVN-versionhallinnassa löytyy tämän dokumentin liitteistä. Liitteissä on myös linkki ohjelmaan itsessään.

5.6 Testaus

Ohjelmiston testaus on tapahtunut ohjelmoijan toimesta syötteiden antamisella ja lukemisella ohjelmiston rakentuessa. Lopullinen Simulaattori on todettu toimivaksi manuaalisesti varmistamalla laskutoimitusten oikeellisuus ja ohjelmiston looginen käyttäytyminen saaduista tuloksista ja visuaalisesta toiminnasta. Varsinaisia JUnit testejä ei ohjelmalle ole tehty.

6 Simulaattorin käyttöohje

Ohjelman käynnistyessä aukeaa aloitusnäky (Kuva 6). Käyttäjä voi nyt aloitusnäkyän oikealla puolella, eli Asetuksissa, asettaa simulaatiolle lähtötiedot. Lähtötiedot on selitetty alla.

Simulointiaika kertoo simuloinnin halutun keston. Simulaattori käsittelee niin monta tapahtumaa kuin ehtii annetussa ajassa.

Simuloinnin viive mahdollistaa simuloinnin seuraamisen hidastetusti graafisesta näkymästä. Jos viiveen asettaa nolaksi, ohjelma vetää sen loppuun hyvin nopeasti.

Asiakkaiden saapumisväli ja *Asiakkaiden saapumismäärä* kertovat yhdessä, kuinka monta asiakasta keskimäärin saapuu annetuin väliajoin.

Kahvila käytössä on ON/OFF näppäin. Jos Kahvila käytössä painetaan OFF tilaan, ei Kahvila voi palvella asiakkaita. Kahvilan asiakkaat eivät valitse toista palvelupistettä, eivätkä he täten tule osaksi simulaatiota.

Kassa 2 käytössä on myös ON/OFF näppäin. Kun Kassa 2 on OFF tilassa, asiakkaiden on pakko valita Kassa 1 siirtyessään Linjastoilta 1-3 eteenpäin.

Ruokalan kapasiteetti kertoo sen, kuinka monta asiakasta mahtuu yhtä aikaa palveltavaksi kyseisellä palvelupisteellä.

Simuloi nappia painamalla simulaatio alkaa ja näkymä vaihtuu simulaationäkymään (Kuva 7). Käyttäjä voi vaikuttaa simulaation edistymiseen sen vielä käynnissä ollessa painamalla Hidasta ja Nopeuta näppäimiä. Keskeytä näppäin lopettaa simuloinnin.

Kun simulaation suoritus on päättynyt, tulospäkymään tulee näkyviin simulaation keräämät tulokset (Kuva 8). Ylemmät tulokset ovat kaikille palvelupisteille yhteisiä ja alemmat sarakkeisiin saadut tiedot ovat palvelupisteille yksilöllisiä. Tulostiedot on selitetty alla, tulosarvojen selitteet löytyvät kappaleesta 4.3.

Simuloinnin kesto on arvo T.

Saapuneet asiakkaat ovat arvo A.

Palvellut asiakkaat ovat arvo C.

Suoritusteho on arvo X, se on laskettu jakamalla palveltujen asiakkaiden määrä simuloinnin kestolla.

Keskimääräinen läpimenoaika on arvo R, eli Läpimenoajat jaettuna Palvellut asiakkaat.

Läpimenoajat yhteensä on arvo W.

Keskimääräinen jonon pituus on arvo N, eli Läpimenoajat jaettuna Simuloinnin kestolla.

Palvelupisteen aktiivinen aika on arvo B.

Palvelupisteen käyttöaste on arvo U, eli Palvelupisteen aktiivinen aika jaettuna Simuloinnin kestolla.

Palvelupisteen keskimääräinen palveluaika on arvo S, eli Palvelupisteen aktiivinen aika jaettuna Palvellut asiakkaat.

Käyttäjä voi halutessaan tallentaa simulaation keräämiä tietoja mySQL tietokantaan painamalla Tallenna tietokantaan -painiketta. Tietokantaan tallentuu palvelupisteiden ja palveltujen asiakkaiden yksilölliset tiedot. Tietokannan sisältöä voi tarkastella paremmin tietokantanäkymästä (Kuva 9). Tänne pääsee painamalla simulaationäkymästä Takaisin ja aloitusnäköymästä Näytä tietokanta. Käyttäjä voi myös halutessaan tyhjentää tietokannan sisällön Tyhjennä tietokanta -painikkeella.

Kun käyttäjä on palannut takaisin aloitusnäköymään, voi hän halutessaan asettaa uudet lähtötiedot ja ajaa ohjelman uudestaan.

7 Tehdyt simulointikokeet

Tässä kappaleessa esitellään viisi simulointikoetta, jotka on ajettu Ruokalasimulaattori ohjelmalla läpi. Kussakin kokeessa on kuvat alkuasetuksista ja saaduista tuloksista. Kaikki kokeet on ajettu kokeiden nopeuttamiseksi 0 viiveellä ja simulointiajalla 5000 vertailun vuoksi. Eri palvelupisteistä on otettu ylös vain käyttöaste sillä se kuvastaa saatuja eroja parhaiten.

Koska yksittäinen simulaation ajokerta saattaa satunnaisuuden vuoksi tuloksiltaan vaihdella hyvinkin paljon, on varmuuden vuoksi sama testi ajettu uudestaan useamman kerran samoilla arvoilla ja näistä ajokerroista on otettu silmämääräisesti vähiten keskiverrosta poikkeava ajokerta varsinaiseksi esiteltäväksi kokeeksi.

7.1 Koe 1

Asetukset

Simulointiaika: 5000

Simuloinnin viive: 0

Asiakkaiden saapumisväli: 10.00

Asiakkaiden saapumismäärä: 1.00

Kahvila käytössä: ON

Kassa 2 käytössä: ON

Ruokalan kapasiteetti: 100

Simuloi

Näytä tietokanta

Tyhjennä tietokanta

Tulokset

Simulointi on päättynyt - tulokset alla

Simuloinnin kesto: 5002.69

Saapuneet asiakkaat: 524

Palvellut asiakkaat: 521

Suoritusteho: 0.10

Keskimääräinen läpimenoaika: 28.32

Läpimenoajat yhteensä: 14753

Keskimääräinen jonon pituus: 2.95

Palvelupisteiden tiedot

Linjasto 1 Linjasto 2 Linjasto 3 Kahvila Kassa 1 Kassa 2 Ruokala

Linjasto 1

Palvelupisten aktiivinen aika: 1045.35

Palvelupisteen käyttöaste: 0.21

Palvelupisteen keskimääräinen palveluaika: 4.93

Hidasta Nopeuta Keskeytä Tallenna tietokantaan Takaisin

Kuva 12. (vasen) Koe 1, asetukset

Kuva 13. (oikea) Koe 1, tulokset

Kokeessa 1 asiakkaita on saapunut yksi asiakas 10 aikayksikön välein. Suoritusteho on jäänyt varsin pieneksi (0,10), sillä asiakkaita on tullut palvelupisteiden palveltavaksi varsin harvakseltaan. Näin ollen aktiivinen aika on jäänyt kaikissa palvelupisteissä hyvin pieneksi. Linjaston 1 käyttöaste on toiseksi korkein mutta silti varsin alhainen. Kaikkein korkein käyttöaste on Ruokalalla (0,88) mutta tämä selittyy Ruokalan varsin pitkällä palveluajalla, ja sillä että Ruokalalle oli annettu varsin suuri kapasiteetti (100), ja aktiiviseksi ajaksi lasketaan se aika, kun ruokalassa on edes yksi asiakas. Seuraavaan kokeeseen voisi lisätä saapuvien asiakkaiden määrää.

Linjasto 2 käyttöaste on 15 %, Linjaston 3 käyttöaste on 12 %, Kassa 1 käyttöaste on 12 %, Kassa 2 käyttöaste on 13 %, ja Kahvilan käyttöaste on 9 %.

7.2 Koe 2

Asetukset

Simulointiaika: 5000

Simuloinnin viive: 0

Asiakkaiden saapumisväli: 10.00

Asiakkaiden saapumismäärä: 2.50

Kahvila käytössä: ON

Kassa 2 käytössä: ON

Ruokalan kapasiteetti: 100

Simuloi

Näytä tietokanta

Tyhjennä tietokanta

Kuva 14. (vasen) Koe 2, asetukset

Tulokset

Simulointi on päättynyt - tulokset alla

Simuloinnin kesto: 5000.28

Saapuneet asiakkaat: 1317

Palvellut asiakkaat: 1310

Suoritusteho: 0.26

Keskimääräinen läpimenoaika: 29.99

Läpimenoajat yhteensä: 39292

Keskimääräinen jonon pituus: 7.86

Palvelupisteiden tiedot

Linjasto 1 | Linjasto 2 | Linjasto 3 | Kahvila | Kassa 1 | Kassa 2 | Ruokala

Linjasto 1

Palvelupisten aktiivinen aika: 2678.87

Palvelupisteen käyttöaste: 0.54

Palvelupisteen keskimääräinen palveluaika: 5.10

Hidasta Nopeuta Keskeytä Tallenna tietokantaan Takaisin

Kuva 15. (oikea) Koe 2, tulokset

Koe 2 suoritettiin muuten samoilla luvuilla, paitsi että asiakkaiden saapumismäärää lisättiin yhdestä 2,5:een. Tuloksista on heti nähtävissä, että suoritusteho on kasvanut huomattavasti, vaikka onkin silti melko alhainen (0,26). Keskimääräinen läpimenoaika on kasvanut vain hieman, joten suurta ruuhkaa ei lisäys ole vielä aiheuttanut. Seuraavaan kokeeseen voisi lisätä edelleen saapuvien asiakkaiden määrää.

Linjasto 1 käyttöaste on 54 %, Linjaston 2 käyttöaste on 39 %, Linjasto 3 käyttöaste on 33 %, Kassa 1 käyttöaste on 33 %, Kassa 2 käyttöaste on 29 %, Kahvilan käyttöaste on 21 %, ja Ruokalan käyttöaste on 100 %.

7.3 Koe 3

Asetukset

Simulointiaika: 5000

Simuloinnin viive: 0

Asiakkaiden saapumisväli: 1 6 11 16 20 10.00

Asiakkaiden saapumismäärä: 1 4 7 10 5.00

Kahvila käytössä: ON

Kassa 2 käytössä: ON

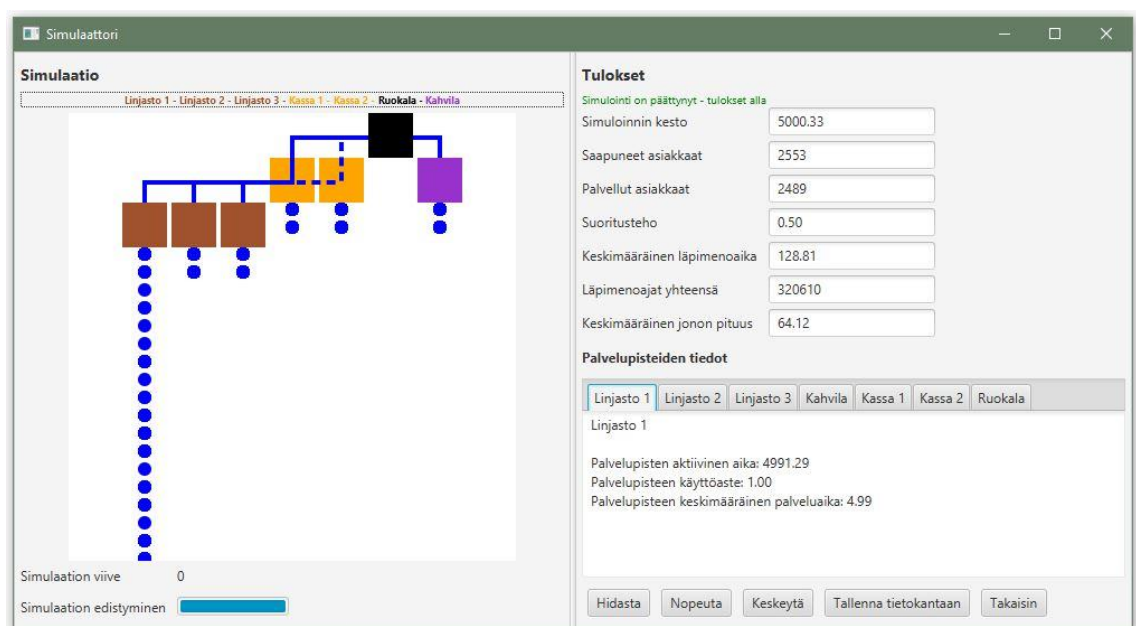
Ruokalan kapasiteetti: 100

Simuloi

Näytä tietokanta

Tyhjennä tietokanta

Kuva 16. Koe 3, asetukset



Kuva 17. Koe 3, tulokset

Koe 3 suoritettiin myös samoilla luvuilla, paitsi että saapumiset tuplattiin viiteen per 10 aikayksikköä. Nyt on heti huomattavissa, että simulaatio on päässyt ruuhkaantumaan, sillä jo graafisesta liittymästä on nähtävissä, kuinka Linjaston 1 jonon pituus on kasvanut todella pitkäksi. Samaa tarinaa kertoo myös se, että keskimääräiset läpimenoajat

ovat yli nelinkertaistuneet. Linjasto 1 on kuitenkin ainut, jossa käyttöaste on kivunnut 100 %. Muut linjastot ovat selvinneet siihen nähden paljon paremmin. Näin ollen Linjasto 1 on muodostunut melkoiseksi pullonkaulaksi simulaatiolle. Seuraavaan kokeeseen voisi saapumisia yrittää laskea hieman, jos Linjaston 1 saisi sillä rullaamaan paremmin.

Linjasto 1 käyttöaste on 100 %, Linjaston 2 käyttöaste on 76 %, Linjasto 3 käyttöaste on 59 %, Kassa 1 käyttöaste on 61 %, Kassa 2 käyttöaste on 60 %, Kahvilan käyttöaste on 41 %, ja Ruokalan käyttöaste on 100 %.

7.4 Koe 4

Asetukset

Simulointiaika: 5000

Simuloinnin viive: 0

Asiakkaiden saapumisväli: 10.00

Asiakkaiden saapumismäärä: 4

Kahvila käytössä: ON

Kassa 2 käytössä: ON

Ruokalan kapasiteetti: 100

Simuloi

Näytä tietokanta

Tyhjennä tietokanta

Tulokset

Simulointi on päättynyt - tulokset alla

Simuloinnin kesto: 5000.47

Saapuneet asiakkaat: 2065

Palvellut asiakkaat: 2048

Suoritusteho: 0.41

Keskimääräinen läpimenoaika: 34.93

Läpimenoajat yhteensä: 71530

Keskimääräinen jonon pituus: 14.30

Palvelupisteiden tiedot

Linjasto 1 Linjasto 2 Linjasto 3 Kahvila Kassa 1 Kassa 2 Ruokala

Linjasto 1

Palvelupisten aktiivinen aika: 4224.26

Palvelupisteen käyttöaste: 0.84

Palvelupisteen keskimääräinen palveluaika: 5.16

Hidasta Nopeuta Keskeytä Tallenna tietokantaan Takaisin

Kuva 18. (vasen) Koe 4, asetukset

Kuva 19. (oikea) Koe 4, tulokset

Koe 4 suoritettiin muuten samoilla luvuilla mutta saapumisten määrä vähennettiin neljään per 10 aikayksikköä. Tuloksista voidaan todeta, että valittu saapumisten määrä vaikuttaisi varsin toimivalle ratkaisulle. Pullonkaloja ei pääse syntymään ja suoritusteho saadaan pysymään melko korkeana. Myöskään läpimenoajat eivät päässeet huomattavissa määrin kasvamaan. Pienillä muutoksilla tuleviin kokeisiin olisi varmasti mahdollista löytää kaikkein optimaalisin asiakkaiden palvelumäärä. Käyttämättä siihen kuitenkaan

sen enempää aikaa, kokeillaan viimeiseen kokeeseen kassojen määrien vähentämisen ja Ruokalan kapasiteetin laskemisen vaikutusta.

Linjasto 1 käyttöaste on 84 %, Linjaston 2 käyttöaste on 60 %, Linjasto 3 käyttöaste on 50 %, Kassa 1 käyttöaste on 49 %, Kassa 2 käyttöaste on 48 %, Kahvilan käyttöaste on 34 %, ja Ruokalan käyttöaste on 100 %.

7.5 Koe 5

Asetukset

Simulointiaika: 5000

Simuloinnin viive: 0

Asiakkaiden saapumisväli: 10.00

Asiakkaiden saapumismäärä: 4

Kahvila käytössä: ON

Kassa 2 käytössä: OFF

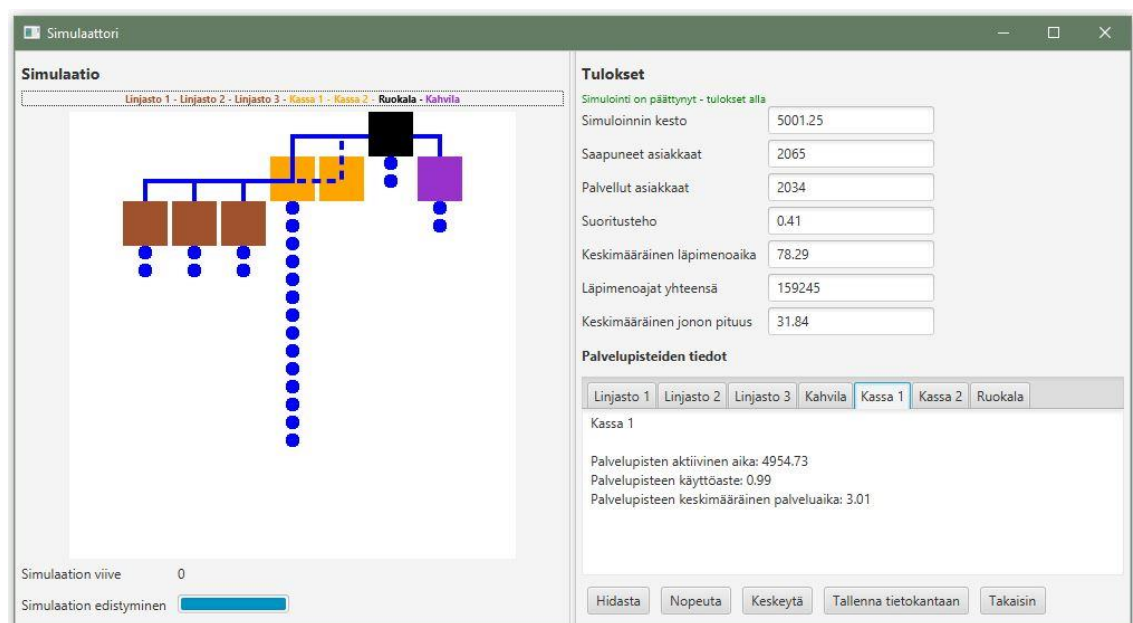
Ruokalan kapasiteetti: 10

Simuloi

Näytä tietokanta

Tyhjennä tietokanta

Kuva 20. Koe 5, asetukset



Kuva 21. Koe 5, tulokset

Koe 5 suoritettiin muutoin samoilla luvuilla kuin koe 4, mutta nyt poistettiin Kassa 2 käytöstä ja vähennettiin Ruokalan kapasiteetti 100:sta 10:een. Nyt eri ajokerrat vaihtelivat huomattavasti mikä johtunee asiakkaiden saamista satunnaisista palveluajoista, joista erityisen pitkä palveluajat nyt pakottavat ihmiset jonottamaan joko ainoalle kassalle, tai ainoalle syömispaikalle, jossa on paikkoja vain kymmenelle henkilölle. Kassan poistumisen vaikutus vaikuttaisi olevan kuitenkin vielä paljon suurempi sillä sen käyttöaste oli 100 % tai hyvin lähelle 100 % joka ajokerralla. Kaikeksi onneksi kassojen palveluaika on varsin lyhyt, joten täyttä seisahdusta ei näillä saapumismäärillä vielä nähty. Läpimenoajat kasvoivat kuitenkin silti yli kaksinkertaisiksi.

Linjasto 1 käyttöaste on 84 %, Linjaston 2 käyttöaste on 60 %, Linjasto 3 käyttöaste on 49 %, Kassa 1 käyttöaste on 99 %, Kassa 2 käyttöaste on 0 %, Kahvilan käyttöaste on 31 %, ja Ruokalan käyttöaste on 100 %.

8 Jatkokehitys suunnitelmat

Simulaattorille oli suunniteltu paljon ideoita, jotka deadlineen lähestyttyä täytyi kuitenkin priorisoida. Näin ollen monet simulaattorille kaavaillut ideat jäivät toteutumatta, mutta liitetään nyt olennaisimmat osaksi tätä dokumenttia.

8.1 Syötteet

Syötteiden määrä jäi priorisoinnissa seuraavaksi asiaksi mikä olisi ollut toteutettavana. Syötteiden määrän lisääminen itsessään on varsin yksinkertainen nykyiseen koodiin sillä jo toteutetut "Asiakkaiden saapumisväli" ja "Asiakkaiden saapumismäärä" syötteet toimivat periaatteeltaan täysin samalla tavalla. Ne viedään simuloinnin käynnistyessä kontrollerille ja alustetaan moottorissa muuttujiin.

Palautteen saamisen syötteiden määrästä ja kokeiden kirjaamisen jälkeen syötteiden prioriteetti olisi ollut korkeampi. Kokeiden tuloksista voi varsinkin huomata, että niin sanottu pullonkaula efektiä ei voi kiertää mitenkään nykyisessä käyttöliittymässä, vaan on vain pakko tiputtaa koko simulaation asiakasmäärää ongelman selvittämiseksi. Yksittäisillä syötteillä tämän olisi voinut kiertää ja saada lisää informaatiota simulaatiosta.

Etenkin valittava keskimääräinen palveluaika jokaiselle palvelupisteelle ja sekä valittava kapasiteetti jokaiselle palvelupisteelle olisivat olleet seuraavia syöte lisäyksiä simulaattoriin.

8.2 Tietokanta

Tietokanta oli kaikkein viimeisin osio mikä tuli osaksi Ruokalasimulaattoria. Sen toimintaan saaminen aivan viime hetkillä tarkoitti kuitenkin sitä, ettei ihan kaikkia toivottuja ominaisuuksia kerennyt liittää osaksi tietokannan toimintoja. Ohjelma on tällä hetkellä valmis tallentamaan kyseisestä ajosta Palvelupisteiden ja Asiakkaiden tiettyjen muuttujien arvot tietokantaan ja myös lukemaan ne sieltä ja palauttamaan ne tekstimuodossa käyttäjän luettavaksi. Ohjelma myös tarjoaa mahdollisuuden tyhjentää tietokannan sisältö.

Puuttumaan jäi mahdollisuus tallettaa useita tietueita yhden sijaan, sekä tulostaa käyttäjän toimesta jonkinlainen tekstitiedosto, johon olisi talletettuna kaikki informaatio jokaisesta suoritetusta ajosta. Tällä hetkellä viimeisin ei olisi kovin hankala toteuttaa, sillä Text Area johon tietoja talletettiin, sisälsi jokaisen ajokerran tekstit, se kuitenkin pistettiin manuaalisesti tyhjentymään joka ajon yhteydessä sillä se ei sillä hetkellä vastannut haluttua toiminnallisuutta.

Viimeisenä asiana tietokantaan olisi voinut tallettaa enemmän tietoa kuin nykyisellään. Tämä olisi vaatinut hieman enemmän muutoksia tiedon tallennusmenetelmiin Moottorissa sekä tietokanta oliossa. Etenkin lähtötiedot olisi hyvä ottaa omaksi datakseen ja yhdistää aina tiettyyn ajokertaan vaikkapa ajankohdalla.

8.3 Käyttöliittymä

Käyttöliittymän parantelemiseen olisi voinut käyttää helposti huomattavan paljon enemmän aikaa useisiin erilaisiin isompiin ja pienempiin parannuksiin. Jossakin vaiheessa oli kuitenkin todettava, että simulaattori oli jo tarpeeksi toimintakykyinen esiteltäväksi. Tässä kuitenkin muuta erityisesti mieleen jäänyt parannusehdotus.

Palvelupisteillä voisi olla Visualisoinnissa tai sen yläpuolella näkyvissä numeroesitys asiakkaiden määrästä, jotka ovat kulkeneet sen läpi tai ovat tällä hetkellä palveltavana. Jälkimmäinen olisi etenkin siinä tapauksessa toivottava, jos kapasiteetti asetettaisiin useammalle kuin yhdelle palvelupisteelle.

Palvelupisteiden tietueiden dataa voisi kasvattaa. Tällä hetkellä toteutin ajan puitteissa vain vaaditut suureet. Lisää suureita olisi voinut kuitenkin laskea olemassa olevasta datasta ja esittää se näin käyttäjälle.

Käyttöliittymän sovittaminen ruudulle. Tarkoituksena oli saada käyttöliittymä toimimaan missä tahansa ruudun koossa kasvattamalla tietueiden ja muiden näkymien kokoa vaikkapa ikkunaa venytettäessä. Tämä oli jo mahdollista esimerkiksi Visualisoinnin puolesta, joka nojaa Canvas piirroksiltaan täysin sille annettuun kokonaisleveyteen ja kokonaispiuuteen.

9 Yhteenveto

Ruokalasimulaattori on sinällään toimiva kokonaisuus, jossa sitä ohjaava logiikka on saatu toimimaan ja käyttöliittymä tarjoaa näkymän käyttäjälle sen testaamiseen ja toteuttamiseen. Simulaattorilla on mahdollista tehdä varsin toimivia testejä ja saada loogisesti toimivia tuloksia, ottaen kuitenkin huomioon, että vielä tällä hetkellä useat muuttujat ohjelman toiminnassa ovat ennalta asetettuja eivätkä ole käyttäjän muutettavissa.

Pitkäikäisenä simulaattorina käyttäjälle tulisi antaa mahdollisuus lisättyjen syötteiden antamiseen. Lisätyt syötteet ja muut parannukset voisivat olla rakentamassa simulaattorista varsin hyvin toimivan työvälineen hyvinkin erilaisissa tilanteissa. Simulaattori on rakennettu pohjalle, joka mahdollistaa kaikki edellä esitetyt parannusehdotukset, ja monet niistä myös hyvin vähäisellä panostuksella.

Liitteet

Simulaattorin lähdekoodi linkki:

https://innoscm.edusrv.metropolia.fi/svn/akikop/trunk/OhPro_Simulaattori_Final_w_JavaDoc

JavaDoc dokumentaatio linkki:

https://innoscm.edusrv.metropolia.fi/svn/akikop/trunk/OhPro_Simulaattori_Final_w_JavaDoc/doc/index.html