LATVIJAS UNIVERSITĀTE

ĢIRTS KARNĪTIS

INFORMĀCIJAS SISTĒMU INTEGRĀCIJAS PROBLĒMAS

Promocijas darbs

Rīga - 2004

LATVIJAS UNIVERSITĀTE

Fizikas un matemātikas fakultāte

ĢIRTS KARNĪTIS

INFORMĀCIJAS SISTĒMU INTEGRĀCIJAS PROBLĒMAS

Promocijas darbs

Datorzinātņu doktora (Dr.sc.comp.) zinātniskā grāda iegūšanai

Nozare: datorzinātnes

Apakšnozare: datu apstrādes sistēmas un datortīkli

Zinātniskais vadītājs

Profesors, Dr.sc.comp.

J. BIČEVSKIS

Rīga - 2004

SATURS

Ievads

20. gadsimta beigās iezīmējās jauns cilvēces attīstības virziens – informācijas laikmets. Informācija kļūst par galveno vērtību un tās daudzums katru gadu palielinās. Atbilstoši ASV Bērklija universitātes zinātnieku pētījumam tikai 2002. gadā vien pasaulē kopumā bija radīts un uzkrāts vairāk nekā $5x10^{18}$ baitu informācijas. Salīdzinājumam: viens metrs grāmatu plauktā satur apmēram 100 megabaitu (10^8 baiti) informācijas [LV03]. Arvien vairāk cilvēku nodarbojas ar datu apstrādi un jaunas informācijas ražošanu. Cilvēce virzās uz jaunu sabiedrības formu – informācijas sabiedrību. Latvijas ceļu uz informācijas sabiedrību iezīmē Nacionālā programma "Informātika" [MoT98a, MoT98b]. Virzība uz informācijas sabiedrību ir kļuvusi iespējama, pateicoties informācijas tehnoloģiju (IT) attīstībai un plašai pieejamībai. Būtiska loma informācijas apmaiņā ir arī interneta attīstībai un komunikāciju pieejamībai. Praktiski visās iestādēs pastāv dažādas informācijas sistēmas (IS), kurās ir uzkrāta iestādes funkcionēšanai bieži vien vitāli nepieciešama informācija, piemēram, finanšu informācija grāmatvedības un finanšu uzskaites sistēmās, klientu informācija gan vienkāršos klientu sarakstos, gan kompleksās klientu attiecību vadības *(Customer relationship management)* sistēmās, kā arī lietvedības, projektu vadības sistēmas un citas iestādes specifiskajai darbībai domātas sistēmas.

Viena no nozarēm, kurai ir raksturīga daudzu specifisku IS nepieciešamība, ir valsts pārvalde. Daudzām valsts iestādēm ir savas specifiskās sistēmas, kas ir nepieciešamas gan konkrētās iestādes, gan arī citu iestāžu darbam. Ja visu valsts pārvaldei nepieciešamo informāciju iztēlojas kā lielu mozaīku, tad katra valsts iestāde uztur kādu gabaliņu no šīs kopējās mozaīkas jeb informācijas. Lai novērstu lieku resursu patēriņu, kāds neizbēgami rodas, atkārtoti apstrādājot vienu un to pašu informāciju vairākās iestādēs, ir nepieciešama informācijas apmaiņa starp dažādām iestādēm un dažādām informācijas sistēmām. Latvijā ir definēta integrētā valsts nozīmes informācijas sistēma jeb Megasistēma, kas ir valsts pārvaldei nepieciešamo informācijas sistēmu kopums. 1998. gadā tika sākts mērķtiecīgs darbs pie Megasistēmas izveides projekta, kura gaitā tika izveidots Megasistēmas ideoloģiskais pamats, kā arī tika apzināti un integrēti 5 primārie valsts nozīmes reģistri:

- Iedzīvotāju reģistrs (IR),
- Uzņēmumu reģistrs (UR),
- Transportlīdzekļu reģistrs (TR),
- Valsts ieņēmumu dienesta (VID) IS,
- Kadastra reģistrs.

Runājot par moderno tehnoloģiju pielietošanu valsts pārvaldē, bieži vien runā par e-pārvaldi, ar to saprotot iespēju klientam vienkāršāk sazināties ar valsts pārvaldes iestādēm. Megasistēma ir viena no e-pārvaldes sastāvdaļām – tā ir parastam iedzīvotājam neredzama jeb tā saucamais *back-office*, kuru iedzīvotājs parasti neredz un pat nenojauš par tāda eksistenci, bet kas ir vitāli svarīga e-pārvaldes eksistencei.

Promocijas darbā pētītās problēmas apgabals ir sistēmu integrācija un datu apmaiņa Megasistēmas ietvaros. Autors apraksta dažādus informācijas uzglabāšanas, apstrādes un apmaiņas teorētiskos aspektus, kā arī praktiski realizētos projektus, kuros šīs idejas ir pielietotas:

- darba pirmā nodaļa ir ievads problēmas apgabalā – Megasistēmā,
- otrajā nodaļā autors veicis teorētisko apskatu par dažādām sistēmu integrācijas metodēm,
- trešā un ceturtā nodaļa satur autora skatījumu uz datu apmaiņu starp sistēmām,
- piektajā un sestajā nodaļā aprakstīti autora praktiski realizētie iepriekšējās nodaļās apskatītie risinājumi – Komunikāciju serveris (KS) un Iedzīvotāju reģistra informācijas sistēma,
- nobeigumā ir autora viedoklis par to, kas praktiski būtu jāveic Megasistēmas tālākai attīstībai un kādus teorētiskos pētījumus būtu lietderīgi veikt KS attīstībai.

1.nodaļā ir ievads problēmas apgabalā - Megasistēmā, kuras izveidē autors piedalījās. Par Megasistēmas ideoloģisko pamatu tika izveidots materiāls "Prasības primārajiem reģistriem". Minētajā dokumentā ir aprakstīti Megasistēmas principi, uzbūve un prasības pret Megasistēmā iesaistītajiem reģistriem. Tika izveidots valsts 5 primāro reģistru konceptuālais datu modelis, kā arī aprakstītas datu plūsmas starp šiem reģistriem. Šis materiāls vēl joprojām kalpo par neoficiālu stratēģisko dokumentu nozīmīgu valsts sistēmu izstrādē un integrācijā.

Megasistēmas izveides projekta laikā tika veikta 5 primāro valsts reģistru - IR, UR, TR, VID IS un Kadastra reģistra analīze. Šie reģistri tika izvēlēti par Megasistēmas kodolu, jo tie satur informāciju par valsts funkcionēšanai vissvarīgākajiem objektiem – fiziskajām personām, juridiskajām personām, teritoriju (nekustamajiem īpašumiem), transportlīdzekļiem, kā arī valsts finansēm. Analīzes rezultāti ir apkopoti piecos dokumentos, par katru reģistru atsevišķi aprakstot tā darbību regulējošo normatīvo aktu kopumu, reģistrā esošo ziņu apjomu un datu plūsmas uz un no reģistriem. Šie dokumenti ir atrodami [MEGA98+].

2.nodaļā ir aprakstītas dažādas IS integrēšanas metodes, apskatīti dažādi datu bāzu modeļu integrācijas veidi – globālā modeļu integrācija, federālās *(federated)* datu bāzes, multidatu bāzes. Visām šim metodēm ir vairāki trūkumi – jo ciešāka integrācija, jo mazāka autonomija paliek konkrētajai datu bāzei. Lai veiktu modeļu integrāciju, ļoti labi jāpārzina katras sistēmas semantika, kā arī nepieciešama diezgan brīva pieeja pie integrējamajām datu bāzēm. Pēdējā laikā arvien plašāk parādās metodes, kuras atļauj integrēt augsti autonomas sistēmas. Šādos gadījumos parasti tiek veidotas starpnieksistēmas *(mediated systems)*. Darbā apskatītas dažādas metodes, kā veidot saites starp datu avotu modeļiem un globālo modeli. Apskatītas pieejas "globāls kā skats" (*Global as view* jeb GAV) un "lokāls kā skats" (*Local as view* jeb LAV), to priekšrocības un trūkumi, kā arī apskatīts, kā veikt pieprasījumu izpildi katrā no šiem modeļiem.

3. nodaļa ir veltīta dažādu datu apmaiņas protokolu priekšrocību un trūkumu analīzei - gan datu bāzu līmenī (piemēram, SQL*Net, ODBC), gan starpprogrammatūras *(middleware)* līmenī (DCOM, CORBA), gan pēdējā laikā popularitāti ieguvušajiem SOAP un Web Services protokoliem. Datu apmaiņa no tehnoloģiskā viedokļa sagādā salīdzinoši vismazāk problēmu, jo pastāv daudzi un dažādi tehnoloģiskie risinājumi, kas balstīti gan uz vispārpieņemtiem, gan konkrētu ražotāju izstrādātiem protokoliem. Jo augstāka līmeņa protokols tiek izmantots, jo plašākas iespējas paveras pašas datu apmaiņas realizācijā, bet sarežģītāka kļūst tās implementācijai nepieciešamā infrastruktūra.

Ja datu apmaiņas arhitektūra balstās uz pašas IS arhitektūru, tad datu apmaiņu ir viegli implementēt, jo lietotāju autorizācijai, pieprasījumu reģistrācijai un bieži vien arī datu atlasei un ievadam var izmantot jau izstrādātu infrastruktūru, savukārt visas datu apmaiņas izveide pilnībā no jauna var izrādīties ļoti darbietilpīga un sarežģīta, kā arī, modificējot pārējo sistēmu, var gadīties, ka jāveic nopietnas izmaiņas arī datu apmaiņas sistēmā.

4. nodaļā ir aprakstīti dažādi datu apmaiņas modeļi, sākot ar datu nolasīšanu no viena reģistra un beidzot ar datu ievadi vairākos reģistros. Šie datu modeļi ir izkristalizējušies izpētes laikā, pētot, kāda veida datu apmaiņa starp sistēmām ir aktuāla. Tiek apskatītas dažādas situācijas, aprakstot un pamatojot, kad un kurš datu apmaiņas modelis būtu piemērotāks, kā arī raksturoti katra modeļa priekšrocības un trūkumi.

Daži no šiem modeļiem ir acīmredzami un bieži pielietojami, piemēram, konkrētu datu atlase no vienas sistēmas. Citi modeļi ir diezgan skaidri saprotami, taču sarežģītāk implementējami un tāpēc retāk pielietoti – piemēram, datu ierakstīšana no vienas sistēmas otrā. Šāda nepieciešamība var rasties gadījumā, ja dati rodas vienā sistēmā, bet uzglabāti tiek citā sistēmā, piemēram, dati par personas dzīvesvietu rodas pašvaldībās, bet

glabājas Iedzīvotāju reģistrā, līdz ar to pašvaldības IS jānodrošina personas dzīvesvietas deklarēšanas notikuma ievadīšana Iedzīvotāju reģistra IS.

Lai gan modeļi ir ļoti līdzīgi, taču bieži vien nevar viennozīmīgi pateikt, kuru modeli labāk izmantot - piemēram, gadījumā, ja vienai sistēmai regulāri ir nepieciešami izmainījušies dati no citas sistēmas, piemēram, Uzņēmumu reģistrā interesē dzīvesvietas izmaiņas personām, kas ir uzņēmumu īpašnieki vai amatpersonas. Šādos gadījumos pastāv vismaz divas dažādas iespējas. Pirmkārt, tā sistēma, kurā dati mainās, publicē visas izmaiņas, un sistēma, kurai šie dati ir nepieciešami, tās paņem, izanalizē un atlasa datus par interesējošiem objektiem. Minētā piemēra gadījumā Iedzīvotāju reģistrs publicētu ziņas par visām personām, kurām ir mainījusies dzīvesvieta laika periodā (piemēram, pēdējā mēneša laikā), bet Uzņēmumu reģistrs atlasītu informāciju par to interesējošām personām. Otrs variants - sistēma, kurā dati mainās, monitorē šīs izmaiņas un piegādā otrai sistēmai tikai to interesējošo objektu izmainījušos datus. Piemēram, Iedzīvotāju reģistrā būtu zināms, kuras personas interesē Uzņēmumu reģistru, un Iedzīvotāju reģistrs piesūtītu Uzņēmumu reģistram datus par dzīvesvietas maiņu tikai šīm personām.

5. nodaļā ir aprakstīts vēl viens interesants datu apmaiņas modelis - saistīto datu atlase no vairākām sistēmām un tā praktiskā realizācija – Komunikāciju serveris. Reizēm ir nepieciešami viena objekta dati, kas atrodas vairākās sistēmās, vai arī ar vienu objektu saistīto citu objektu dati no citām sistēmām. Autors šo datu apmaiņas modeli realizēja 1999. un 2000. gadā Baltijas valstu valdību datu pārraides tīkla projekta ietvaros, kad tika veikta KS izstrāde. KS ir aparatūras un programmatūras komplekss, kas nodrošina sistēmas metadatu apskati - kādi dati kurā sistēmā tiek glabāti, kā arī informācijas pieprasīšanu no dažādām sistēmām un dažādu saistīto objektu datu skatīšanu. KS sastāv no divām daļām – Reģistru reģistra un Universālā pārlūka (UP).

Reģistru reģistrs ir reģistrs, kurā tiek reģistrētas valsts pārvaldei nepieciešamās informācijas sistēmas. Reģistru reģistrs satur informāciju gan cilvēkam saprotamā formā, gan arī metadatu bāzi jeb repozitoriju, ko izmanto datorizētas sistēmas. Cilvēkam saprotamā informācija ir vispārējā informācija par IS – nosaukums, vispārīgs satura apraksts, sistēmas turētājs un izstrādātājs, likumdošanas akti, kas attiecas uz šo sistēmu, tehniskā realizācija un cita informācija. Repozitorijs satur sistēmas konceptuālo modeli, kur ir informācija par sistēmas entītijām un relācijām starp tām un saitēm starp sistēmām, kā arī ar kura iepakotāja (*wrapper*) palīdzību piekļūt datiem un kādā formātā tos attēlot. Reģistru reģistrs ir pieejams interneta adresē http://www.mega.lv/nls/lv/Rr_pub/RegMain.asp.

Universālais pārlūks ir īpaši izstrādāta datu atlases sistēma no dažādām IS, kuru var lietot bez nepieciešamības iedziļināties šo IS tehniskajā uzbūvē un datu modeļos. Šis pārlūks

sākotnēji tika veidots BVVDPT projekta ietvaros, lai interesenti no citām valstīm varētu piekļūt Latvijā esošai informācijai, izmantojot vienotu resurspunktu, taču vēlāk izrādījās, ka tas ir spēcīgs rīks lietošanai valsts iekšpusē un pat konkrētas iestādes ietvaros, lai varētu atlasīt saistītu informāciju no daudzām iestādes IS un datu bāzēm.

Autors pilnībā izstrādāja universālā pārlūka 1.versiju (2.versijas realizācijā piedalījās vēl vairāki Latvijas Universitātes speciālisti). UP ir uz metadatiem bāzēts datu pārlūks un realizēts kā interneta resurspunkts, kuram var pieslēgties autorizēti lietotāji. Tā kā datu avoti var saturēt konfidenciālu informāciju, tad piekļuvei izmanto HTTPS protokolu. Lietotāji tiek identificēti, un identifikācijai var izmantot privāto/publisko atslēgu mehānismu. Lietotājam, atkarībā no viņa tiesībām, var būt pieejama plašāka vai ierobežotāka informācija.

UP darbībai tiek izveidoti datu avotu konceptuālie modeļi, kurus UP izmanto, lai datus atlasītu, attēlotu un nodrošinātu navigāciju pa datiem. Izmantojot konceptuālo modeļu datu bāzi, UP lietotājam piedāvā izvēlēties meklējamo objektu tipu, kā arī formulēt pieprasījuma kritērijus. UP veic datu pieprasījumu un parāda rezultātu. Lietotājam ir iespēja apskatīt detalizētāku informāciju, kā arī sameklēt un apskatīt saistītos objektus. Konceptuālie datu modeļi tiek veidoti kā rīka GRADE [GRA00] klašu diagrammas. GRADE izveidoto modeli eksportē uz eksporta-importa formāta (EIF) failu, kuru speciāla programma analizē un ieliek UP metadatu bāzē.

Piekļuvei datu avotiem tiek izmantots specifiski moduļi – iepakotāji. Iepakotāji tiek veidoti katram konkrētajam datu avotam, translē UP pieprasījumu datu avotam saprotamā formā un noraida uz datu avotu izpildei. Pēc atbildes saņemšanas iepakotājs atbildi pārveido UP saprotamā formātā un nodod UP. Iepakotāji nav UP sastāvdaļas, bet pilda starpnieku lomu starp UP un datu avotu, nodrošinot iespēju viegli pievienot jaunu datu avotu, kā arī ieviest izmaiņas esošajos datu avotos. UP iekšienē visi dati tiek glabāti un apstrādāti XML *(eXtensible Markup Language)* formātā.

Autors veidoja UP, izejot no pieņēmuma, ka datu avoti ir specifiskām vajadzībām veidoti reģistri un bieži vien satur ierobežotas pieejamības informāciju, tāpēc ārējo sistēmu (t.sk. UP) iespējas piekļūt datu avotam ir ļoti ierobežotas – parasti datu avots atļauj izpildīt tikai kaut kādas stingri definētas funkcijas. Atbilstoši šim pieņēmumam, autors izstrādāja paņēmienu, kā veidot datu avotu metamodeļus un saites starp objektiem, balstoties uz tām funkcijām, kuras datu avots atļauj izpildīt. Autors ieviesa šādus UML sintakses paplašinājumus:

- speciālu saites tipu starp objektiem – meklēšanas saiti,
- speciālu atribūtu tipu – meklēšanas atribūtu,
- virsklasi ar paplašinātu semantiku.

6. nodaļā ir aprakstīts Iedzīvotāju reģistrs (IR) un IR IS. Pašreiz IR tiek lietota 1996. gadā veidotā IS, kas gadu gaitā ir papildināta un modernizēta, bet saglabājusi 1996. gadā izstrādāto ideoloģiju. Tā kā šī sistēma ir novecojusi, tad 1999.gadā tika nolemts izveidot jaunu IS, kas atbilstu mūsdienu prasībām pēc funkcionalitātes, tehnoloģijas, drošības u.c. aspektiem. 2000. gadā tika uzsākts jaunas IR IS izstrādes projekts, kuru vadīja šī darba autors. Šī sistēma ir mērķtiecīgi veidota kā pilotprojekts atbilstoši visiem Megasistēmas principiem un prasībām.

Iedzīvotāju reģistrs ir viens no 5 primārajiem reģistriem, kas ir vitāli svarīgi Megasistēmas un valsts pārvaldes funkcionēšanai. Viens no šiem principiem nosaka elektronisku datu apmaiņu ar citām sistēmām. Darbā apskatītas dažādas datu apmaiņas tehnoloģijas, kas tika implementētas IR jaunajā sistēmā. Viena no šādām tehnoloģijām ir specifiska UP versija, kas nodrošina IR IS, meklēšanā esošo personu datu bāzes un nederīgo dokumentu datu bāzes integrētu apskati.

UP ir domāts, lai IR datus varētu apskatīt lietotājs - cilvēks. Lai datus varētu saņemt citas IS, tika izveidots uz metamodeļiem balstīts datu atlases rīks. Ar šī rīka palīdzību var definēt ārējās sistēmas, tām pieejamos pieprasījumus un atbilžu datu kopu apjomus un izskatu. Caur WWW saņemot pieprasījumu XML formātā, rīks autorizē pieprasījuma veicēju, atlasa datus atbilstoši pieprasītāja - sistēmas tiesībām, noformē atbildi un atdod pieprasītājai sistēmai.

Jaunajā IR sistēmā tika iestrādāts personas datu monitorings. Tika izstrādāta speciāla apakšsistēma, ar kuras palīdzību IR personāls var nodefinēt monitorējamo datu kopu. Ārējās sistēmas var definēt savu monitorējamo personu loku. Ja monitorējamai personai mainās kādi no monitorējamiem datiem, tad izmaiņu fakts tiek atzīmēts monitoringa datu bāzē. Kad šī ārējā sistēma pieslēdzas IR IS, tā saņem informāciju, kurām personām dati kopš pēdējās datu saņemšanas reizes ir mainījušies, un ārējā sistēma var saņemt šo personu jaunākos datus.

Vēl viens specifisks datu apmaiņas veids tika izstrādāts datu saņemšanai no ārējām sistēmām. Tādas sistēmas pašlaik ir Pasu IS (informācija par izsniegtajām pasēm) un Pašvaldību vienotā IS (personu dzimšana, laulība, uzvārda maiņa, adreses maiņa). Notikumi no šīm sistēmām tiek saņemti caur WWW XML formātā un pēc tam speciāls iepakotājs tos pārveido IR starpprogrammatūrai saprotamā formātā, un, izmantojot starpprogrammatūru, šie notikumi tiek saglabāti IR IS datu bāzē. Problēmu gadījumā informācija tiek reģistrēta un ārējai sistēmai tiek nosūtīts atbilstošs paziņojums.

Darba nobeigumā autors īsi uzskaita Megasistēmas, kā arī UP tālākai attīstīšanai veicamos darbus -

- Jaunu sistēmu iesaistīšana datu apmaiņā Megasistēmas ietvaros,

- Reģistru reģistra uzturēšana un paplašināšana, papildus saskarņu pievienošana,
- XML kā standarta ieviešana datu apmaiņā starp sistēmām,
- Komunikāciju servera funkcionalitātes paplašināšana, lai varētu KS izmantot kā vienu resurspunktu informatīvajām sistēmām,
- Domēnspecifiskas valodas pielietošana UP WWW lapu izskata un funkcionalitātes aprakstīšanai,
- Papildus pētījumu veikšana par iespējām izmantot DATALOG valodu kompleksu pieprasījumu veikšanai UP,
- Papildus pētījumu veikšana, lai noskaidrotu saistību starp UP metamodeli un lokālajiem datu avotu modeļiem un kā tas saistās ar LAV un GAV pieejām, un par kompleksu pieprasījumu pārformulēšanu no UP modeļa uz datu avotu lokālajiem modeļiem.

Promocijas darbā aprakstītie rezultāti ir iegūti pēdējo 6 gadu laikā un ir tapuši vairāku projektu laikā (Megasistēma, Komunikāciju serveris, Reģistru reģistrs, Iedzīvotāju reģistrs), kā arī ir aprakstīti vairākās publikācijās un izklāstīti dažādās konferencēs (Baltic DB&IS, Baltic IT&T, IS'2002 (Slovēnija), TIBO (Baltkrievija), LU zinātniskā konference). Tālāk tekstā ir uzskaitīti nozīmīgākie projekti, kuros praktiski pielietotas un realizētas teorētiskās idejas, risinot informācijas uzglabāšanas, apstrādes un datu apmaiņas problēmas informatīvajās sistēmās.

- Autors 1998.-1999.gadā ir piedalījies Megasistēmas projektā, kura laikā tika izstrādāti vairāki valsts mēroga konceptuālie dokumenti. Autors sniedza ieguldījumu dokumenta "Prasības primārajiem reģistriem" izstrādē. Šī darba rezultātā tika formulēti Megasistēmas principi, kuri joprojām ir aktuāli un tiek izmantoti nozīmīgu informatīvo sistēmu izveidē. Ar šo tēmu ir saistīti sekojošie autora darbi [ABKK02, ABK00a, BK02, ABKK01],
- Autors 1998.gadā piedalījās Iedzīvotāju reģistra sistēmanalīzes projektā, kura laikā tika izanalizēta reģistra funkcionalitāte, sadarbība ar citām sistēmām un datu kvalitāte. Balstoties uz iegūtajiem analīzes rezultātiem, tika izstrādāta IR IS tālākās attīstības koncepcija,
- Autors piedalījās Baltijas valstu valdību datu pārraides tīkla (BVVDPT) projekta izstrādē, kura laikā tika pieņemti un realizēti vairāki Baltijas mēroga lēmumi par vienota resurspunkta – KS izveidi saskaņā ar Eiropas datu apmaiņas starp administrācijām projektu IDA un IDA II [IDA00]

koncepciju. Nozīmīgs autora ieguldījums BVVDPT projektā ir vienotā resurspunkta - Komunikāciju servera izstrāde. Autors izpētīja pasaulē lietotās starpnieksistēmu izveides pieejas un risinājumus un secināja, ka piemērota risinājuma ar nepieciešamo funkcionalitāti nav, tāpēc autors izveidoja jaunu pieeju starpnieksistēmas izveidei un realizēja to praktiski, izveidojot Universālo pārlūku. Sistēma ir izstrādāta un nodota pasūtītājam. Ar šo tēmu ir saistīti sekojošie autora darbi [AK02a, AK02b, AK00, AK01, ABK99, BK98, ABK00b],

- Autora vadībā laikā no 2000. līdz 2003.gada beigām saskaņā ar IR IS koncepciju tika izstrādāta Iedzīvotāju reģistra jaunā IS, kurā tika iestrādāti arī Megasistēmas principi. Datu sniegšanai tiešsaistes režīmā tika izmantots autora izstrādātais Universālais pārlūks, modificējot un adaptējot šo rīku IR specifiskajām vajadzībām un veicot pāreju uz XML formāta izmantošanu. Pēc autora iniciatīvas un idejiskā risinājuma tika izveidots uz metamodeli bāzēts universāls un elastīgi konfigurējams rīks datu apmaiņai ar ārējām sistēmām un datu monitorēšanai. Sistēma ir pilnībā izstrādāta un nodota pasūtītājam. Ar šo tēmu ir saistīti sekojošie autora darbi [Kar02, Kar03a, Kar03b, Kar04].

1. Integrētā valsts nozīmes informācijas sistēma – Megasistēma

Šajā nodaļā ir apskatīta valsts nozīmes informācijas sistēmu integrācijas nepieciešamība un Latvijas pieeja - Megasistēmas izveide, kā arī pārskaitīti Megasistēmas principi.

1.1. Problēmas nostādne

Straujais tehnoloģiju progress veicina plašu IT izmantošanu valsts pārvaldē gan *back-office* nodrošināšanā, gan arī tiešajā saskarsmē ar iedzīvotājiem. E-pārvaldes nodrošināšanai ir nepieciešamas trīs galvenās lietas – dokumenti, dati un procedūras [Kar01].

Dokumenti ir galvenais informācijas aprites veids valsts institūcijās. E-pārvaldes ietvaros ir jārunā par elektroniskajiem dokumentiem, ko paraksta ar elektroniskajiem parakstiem un kā ieviešanai ir nepieciešama atbilstoša infrastruktūra elektronisko parakstu izsniegšanai un pārbaudei *(Public Key Infrastructure)*.

Otra būtiskā lieta valsts pārvaldes nodrošināšanai ir dažādi nepieciešamie dati, tā sauktā publiskā sektora informācija, kas rodas un ir nepieciešama valsts pārvaldes un pašvaldību institūcijās - gan dažādi reģistri, gan dažādas ziņas utt. Publiskā sektora informācijas un tās atrašanās vietas apzināšana (reģistri, datu bāzes utt.) un to integrēšana ir integrētās valsts nozīmes informācijas sistēmas jeb Megasistēmas projekta mērķis.

Līdz ar dokumentiem un datiem trešais valsts pārvaldes aspekts ir procedūras. Tās nosaka, kā valsts funkcionē, kas un kā ir jādara iedzīvotājiem, kā darboties valsts institūcijām un ierēdņiem. Procedūras definē dažādos normatīvajos aktos, tomēr bieži vien likumdošana procedūras apraksta neskaidri un pat pretrunīgi. Līdz ar to precīzu automatizējamu procedūru izveidei ir nepieciešams pamatīgs likumdošanas aktu analīzes darbs. Procedūras automatizējot, izveidojas darba plūsma *(workflow)*. Jāpiebilst, ka procedūru modernizācijai Latvijā praktiski nav pievērsta uzmanība.

Šajā promocijas darbā ir apkopots autora veikums Megasistēmas sastāvdaļu izveidē, galvenokārt pievēršot uzmanību datu apmaiņai starp dažādām sistēmām.

1.2. Megasistēmas principi

Veidojot Megasistēmu, izkristalizējās vairāki principi, kuri ir jāievēro, veidojot integrētu valsts nozīmes informācijas sistēmu [MEGA98b]:

- Megasistēma ir nevis viens liels "superreģistrs", kurā ir visu reģistru informācija, bet gan daudzu atsevišķi darbojošos, harmonizētu reģistru kopums. Visiem valsts nozīmes objektiem (tie ir fiziskas personas, juridiskas personas, zeme un nekustamais īpašums, transportlīdzekļi) ir jābūt reģistrētiem reģistros,

- Objekta reģistrācijas rezultātā ir jāizsniedz objekta reģistrācijas apliecība (pase, transportlīdzekļa reģistrācijas apliecība utt.),
- Reģistrācijas apliecība ir jādrukā no datu bāzes. Katru objektu ir atļauts reģistrēt tieši vienā reģistrā. Ir jābūt precīzi definētai informācijai par objektu atrašanās vietu, kā arī noteiktai atbildībai par informācijas pilnību un pareizību,
- Citām informācijas sistēmām ir jāizmanto par objekta datu pareizību atbildīgajā reģistrā esošā informācija par objektu. Nav pieļaujama atkārtota manuāla datu ievade. Ātrdarbības paaugstināšanai utml. nolūkiem var veikt elektronisku datu dublēšanu citā sistēmā,
- Informācija ir jāfiksē elektroniski tās rašanās vietā,
- Valsts nozīmes IS nav atļauta atkārtota informācijas ievade no reģistrācijas apliecības vai citiem dokumentiem,
- Paredzams, ka tuvākajos gados tradicionālos papīra dokumentus nomainīs elektroniskie dokumenti, un faktus un notikumus apliecinās ar datu bāzes informāciju, līdz ar to datu bāzēs katrs ieraksts kļūs par juridiski tiesīgu elektronisku dokumentu.

Pašlaik Latvijā speciālisti ir akceptējuši šos principus un, veidojot informatīvās sistēmas valsts pārvaldes vajadzībām, tie tiek lielā mērā ievēroti.

Megasistēmas projekts - saskaņotas valsts nozīmes reģistru sistēmas izveide ar precīzi definētu atbildību un datu savākšanas tehnoloģiju, ko apstiprina normatīvie akti, ir cieši saistīts ar Baltijas valstu valdību datu pārraides tīkla (BVVDPT) projektu. BVVDPT apzīmē modernu informācijas pakalpojumu kopumu, kas nodrošina Latvijas administratīvajām institūcijām ērtu un drošu informācijas apmaiņu ar Baltijas valstu un citu Eiropas valstu pārvaldes un administratīvajām institūcijām starpvalstu līmenī, ieskaitot speciālos balss/ video sakarus. BVVDPT ietvaros tika veikta Latvijas sistēmu integrācija ar Eiropas sistēmām – UR integrācija ar Eiropas uzņēmumu reģistru *European Business register* un TR integrācija ar Eiropas transportlīdzekļu reģistru EUCARIS.

Megasistēmas projekts tiek veidots kā Latvijas mēroga projekts, savukārt BVVDPT projekts saista Megasistēmu ar ārējiem datu avotiem un informācijas patērētājiem.

2. Ievads sadalītajās datu bāzēs

Šī nodaļa ir apskats par pasaulē pielietotām metodēm sistēmu integrēšanā un tajā ir sniegts gan vispārīgs apraksts par sistēmu integrēšanu, gan arī vairāku metožu sīkāks izklāsts.

2.1. Terminoloģija

Šajā nodaļā ir aprakstīti vairāki termini [ERS99], kas saistīti ar sadalītajām datu bāzēm.

2.1.1 Sadalīšana

Lielās sistēmās bieži vien datu bāzes un dati tajās ir fiziski sadalīti pa dažādām vietām., arī viena konkrēta IS var būt fiziski sadalīta pa vairākiem datoriem. Dati starp dažādām datu bāzēm var būt sadalīti gan vertikāli (viena objekta dažādi atribūti ir dažādās sistēmās, piemēram, personas kods, vārds un uzvārds atrodas vienā sistēmā, bet personas kods un dzīvesvietas adrese atrodas otrā sistēmā), gan horizontāli (dažādu objektu vieni un tie paši atribūti atrodas dažādās datu bāzēs, piemēram, katras pašvaldības informatīvā sistēma satur pilnu pašvaldībai nepieciešamo informāciju par personām, bet tikai par tām personām, kuras dzīvo konkrētajā pašvaldībā). Dati var būt replicēti vairākās sistēmās un tādā gadījumā ir jāuztur aktuālā stāvoklī visas datu kopijas.

Datu sadalīšanai ir vairākas priekšrocības – uzlabojas pieejamība datiem un samazinās pieejas laiks. Tajā pašā laikā citas operācijas – datu aktualizēšana, datu pieprasīšana no vairākiem informācijas avotiem - kļūst sarežģītākas.

2.1.2 Heterogenitāte

Sadalīta IS var būt vai nu homogēna, vai heterogēna. Praksē nereti vērojama divu jēdzienu "sadalīta datu bāze" un "heterogēna datu bāze" nekonsekventa lietošana. Homogēnā sadalītā sistēmā datu apstrāde visās vietās tiek veikta ar vienu un to pašu programmatūru, kā arī tiek izmantoti vienādi datu modeļi un ir vienāds pats apstrādes process. Līdzko sistēma kaut pēc viena no minētajiem parametriem neatbilst homogēnas sistēmas prasībām, to sauc par heterogēnu sistēmu. Heterogenitāte var pastāvēt visdažādākajos līmeņos, piemēram, operētājsistēmā, datu bāzu pārvaldības sistēmā (DBPS), datu modelī, datu apstrādes aplikācijā. Jo vairāk līmeņos ir atšķirības, jo sistēmu integrācija ir sarežģītāka.

2.1.3 Autonomija

Datu bāzes pārvaldošās organizācijas bieži vien ir neatkarīgas viena no otras un neatkarīgi viena no otras pārvalda savas datu bāzes. Līdz ar to IS kontrolējošās organizācijas citām organizācijām ļauj piekļūt saviem datiem tikai tādā gadījumā, ja

pārvaldošā organizācija saglabā kontroli par IS darbināšanu un piekļuvi tās datiem. Līdz ar to kļūst svarīgi dažādi autonomijas aspekti, kuri ietekmē sadarbību starp IS:

- Projektējuma autonomija: IS izstrādātāji paši izvēlas sistēmas datu modeli, pieprasījumu valodu, arhitektūru, datu semantisko interpretāciju, integritātes ierobežojumus u.c.,
- Komunikāciju autonomija: IS var izvēlēties, kad un kā atbildēt uz pieprasījumiem no citām IS,
- Izpildes autonomija: pieprasījumu izpildes kārtību nosaka IS, un ārējās IS to nevar ne ietekmēt, ne kontrolēt. IS neinformē ārējās sistēmas par iekšējo pieprasījumu izpildi un ietekmi uz ārējiem pieprasījumiem,
- Asociāciju autonomija: IS izstrādātāji var izlemt, cik un kādas operācijas (*project, select, join* u.c) piedāvās ārējiem lietotājiem. Izpildes statistikas apjoms, ko piedāvā ārējiem lietotājiem (pieprasījuma izpildes laiks, efektivitāte), arī ir atkarīgs no konkrētās datu bāzes izstrādātāju vēlmēm, līdz ar to globālu pieprasījumu optimizācija ir apgrūtināta.

2.1.4 Sadarbspēja

IS sadarbspēja (*interoperability*) nozīmē sistēmas spēju pieprasīt un saņemt servisus no citām sistēmām. Zemākā līmeņa sadarbspēja ir, piemēram, periodiska datu nosūtīšana citai sistēmai vai datu saņemšana no tās. Augstāka līmeņa sadarbspēja ir, piemēram, starpatkarība (*interdependency*), kas nozīmē, ka dažādu sistēmu dati un funkcijas ir atkarīgi viens no otra, tajā pašā laikā lietojumprogrammas par šo savstarpējo atkarību var pat nezināt, piemēram, vienas DB saglabātās procedūras izsaukšana no saglabātās procedūras citā DB. Starpatkarīgās sistēmās iezīmējas jauna problēma, kura parasti nepastāv vienas sistēmas ietvaros – kā nodrošināt datu integritāti starp vairākām, savā starpā saistītām sistēmām.

Vispārīgā gadījumā sistēmas var uzskatīt par sadarbspējīgām, ja izpildās šādi nosacījumi:

- tās spēj apmainīties ar paziņojumiem un pieprasījumiem,
- tās spēj saņemt ārējus pieprasījumus un šo pieprasījumu apstrādē darboties kā vienota sistēma.

Lai sistēmas spētu izpildīt šos nosacījumus, tām ir:

- jāspēj lietot vienai otras funkcionalitāti,
- jāspēj sadarboties, neraugoties uz iekšējās realizācijas atšķirībām,

- jābūt sadalītām sistēmām,
- jāvar iesaistīt sadarbībā citas sistēmas.

2.2. Sistēmu integrācijas veidi un to klasifikācija

Sistēmu integrāciju var veikt, veidojot vai nu materializētos skatus *(materialised views)*, vai virtuālos skatus. Materializēto skatu gadījumā dati tiek atlasīti no datu bāzes un saglabāti, lai no tiem vēlāk varētu atlasīt nepieciešamos datus. Materializēto skatu veidošana tiek pielietota, piemēram, datu noliktavās. Šajā promocijas darbā uzmanība galvenokārt pievērsta datu integrācijas metodēm, kurās tiek veidoti virtuālie skati – tiek veikta modeļu integrācija un pēc tam datu atlase, izmantojot integrēto modeli. Virtuālo skatu gadījumā nekādi iepriekš sagatavoti dati netiek veidoti un glabāti, bet dati no sistēmām tiek iegūti brīdī, kad tos pieprasa lietotājs.

Šo abu pieeju gadījumā integrācijas veicējam ir nepieciešams saprast datu sintaksi un semantiku dažādās sistēmas un veikt datu modeļa pārveidi starp integrējamo sistēmu sintaksēm un semantikām. Literatūrā var atrast dažādas metodes, kā veikt datu un datu modeļa integrāciju un pārveidošanu. Tāpat var atrast dažādus klasifikāciju veidus, kā šīs metodes tiek klasificētas.

Viens no klasifikāciju veidiem ir balstīts uz abstrakcijas līmeni, kurā tiek pielietota sistēmu integrācija. Otrs veids ir klasifikācija pēc izmantotā datu modeļa veida.

Vēl viens klasifikācijas veids ir pēc tā, kā tiek veidota saite starp integrēto modeli un lokālajām modeļiem. Izšķir divus visbiežāk lietotos paņēmienus – "lokāls kā skats" *(Local as View, LAV)*, kurā lokālie modeļi tiek izteikti kā skati uz globālo modeli, un "globāls kā skats" *(Global as View, GAV)*, kurā globālais modelis tiek izteikts kā skats uz lokālajiem modeļiem. Papildus parādās arī kombinētie veidi BAV *(Both as view)*, GLAV *(Global and local as view)*, BGLAV *(Both global and local as view)*, P2P *(Point-to-point)*.

2.2.1 Uz abstrakcijas līmeni balstītā klasifikācija

Ja tiek integrētas vairākas sistēmas, to var veikt vienā no šādiem līmeņiem un katrā no šiem līmeņiem var izdalīt dažādas integrācijas metodes:

- Lietotāju skatu līmenī: lietotāju skatu integrāciju metožu ideja un mērķis ir integrēt dažādu sistēmas lietotāju skatus uz sistēmu vienā kopīgā datu bāzes modelī. Skatu integrācija parasti ir datu bāzes projektēšanas procesa sastāvdaļa, līdz ar to tā tiek veikta pirms reālās datu bāzes izveides, tādējādi atvieglojot semantisko konfliktu un heterogenitāšu atrisināšanu, jo tādas rodas maz. Tāpat arī parasti nav nepieciešama modeļu translācija, jo

projektēšanas laikā visi lietotāju skati tiek veidoti vienā un tajā pašā formālismā.

- Konceptuālo modeļu līmenī: metodes, kas balstītas uz konceptuālajiem modeļiem, apraksta dažādu sistēmu modeļu integrēšanu. Lai to izdarītu, integrācijas laikā jātiek galā ar strukturālajām un semantiskajām heterogenitātēm.

- Datu līmenī: trešā tipa metodes galvenokārt darbojas datu līmenī. Šī līmeņa metodes balstās uz konkrētajiem datu bāzu datiem, lai veiktu integrāciju. Datu integrācijas metodēm ir jārisina divas galvenās problēmas:

    1. Entītiju identifikācija: kā identificēt ierakstus divās dažādās datu bāzēs, kuri pēc būtības reprezentē vienu un to pašu reālās pasaules objektu?

    2. Atribūtu vērtību konflikti: ko darīt situācijā, ja dati par vienu un to pašu reālās pasaules objektu dažādās datu bāzēs ir dažādi?

2.2.2 Uz izmantotā globālā modeļa tipu balstītās klasifikācijas

Cits klasifikācijas veids ir balstīts uz izmantotā globālā modeļa tipu. Parasti tiek lietoti četri modeļu tipi - relāciju modelis, semantiskais modelis (ER modelis u.c. formālismi), objektorientētie modeļi un ar pirmās pakāpes predikātu rēķiniem aprakstāmi modeļi (turpmāk predikātu modeļi).

- Relāciju modelis bija viens no pirmajiem, ko lietoja modeļu integrēšanai. Relāciju modeļa pozitīvā puse ir relāciju datu bāzu plašā izplatība, kā arī attīstītā pieprasījumu valoda. Diemžēl, relāciju modelis ir semantiski nabadzīgs, līdz ar to tajā nevar attēlot sarežģītus ierobežojumus. Relāciju modeli reizēm lieto sistēmās, kas nodarbojas ar sistēmu integrāciju datu līmenī.

- Semantiskais modelis tiek lietots biežāk, jo tajā iespējams definēt krietni plašāku semantiku nekā relāciju modelī un to var izmantot modeļu integrācijas laikā. Tā kā semantiskie modeļi parasti tiek lietoti konceptuālo modeļu attēlošanai, tad semantisko modeli parasti izmanto konceptuālo modeļu un lietotāju skatu integrācijas laikā.

- Objektorientētais modelis literatūrā tiek apskatīts atsevišķi, neraugoties uz to, ka tas pēc būtības ir semantiskais modelis ar paplašinājumiem. Parasti šo modeli izmanto tāpat kā semantisko modeli konceptuālo modeļu integrācijas

laikā, tomēr ir sistēmas, kuru integrēšanai tiek izmantots objektorientētais modelis, piemēram DISCO [TRV96].

- Predikātu modeļi nozares literatūrā parādās arvien biežāk, jo predikātu rēķinus *(first-order logic)* var izmantot, lai formalizētā veidā definētu relāciju datu bāzu semantiku. Tāpat arī predikātu modeļi ļauj nodefinēt tādu semantiku, kādu nav iespējams izteikt ar semantisko modeļu palīdzību, piemēram, lietotāja definētus integritātes ierobežojumus. Šajos modeļos parasti izmanto divus formālismus – predikātu rēķinus vai datu apstrādes valodu DATALOG.

2.2.3 Pēc saites tipa starp globālo un lokālajiem modeļiem

Lai varētu veikt pieprasījumu pārformulēšanu no globālā modeļa uz lokālajiem modeļiem, ir jānoformulē sakarības starp šiem modeļiem. Literatūrā ir aprakstīti divas pieejas šo sakarību formulēšanai:

- Globāls kā skats *(Global as View, GAV)*. Globālais modelis tiek definēts kā skats no lokālajiem modeļiem.
- Lokāls kā skats *(Local as View, LAV)*. Lokālie modeļi tiek definēti kā skati uz globālo modeli.

2.2.3.1. Piemērs, kā tiek definēti skati katrā no pieejām

Tiek pieņemts, ka ir šāds globālais modelis [Len02]:

```
movie(Title;Year;Director)

european(Director)

review(Title;Critique)
```

kur `movie` satur informāciju par filmām, `european` satur informāciju par Eiropas režisoriem un `review` satur filmu kritiku.

Tāpat arī ir divi šādi datu avoti:

1. datu avots:

   `r1(Title;Year;Director)` – informācija par Eiropas režisoru filmām kopš 1960. gada.

2. datu avots:

   `r2(Title;Critique)` – informācija par filmām kopš 1990. gada.

Interesē pieprasījums "1998. gada filmu nosaukumi un to kritika". Pārformulējot predikātu rēķinos, interesē raksti, kas atbilst šādam izteikumam:

```
∃D: movie(T;1998;D) ^ review(T;R);
```

jeb pārrakstot:

```
{ (T;R) | movie(T;1998;D) ^ review(T;R) }
```

LAV gadījumā lokālie datu avoti būtu izteikti šādā veidā ar globālā modeļa palīdzību:

```
r1(T;Y;D) :- { (T;Y;D) | movie(T;Y;D) ^ european(D) ^ Y
>= 1960 }

r2(T;R) :- { (T;R) | movie(T;Y;D) ^ review(T;R) ^ Y >=
1990 }
```

Pieprasījums tiktu pārformulēts šādi:

```
{ (T;R) | r2(T;R) ^ r1(T;1998;D) }
```

GAV gadījumā globālais modelis tiktu izteikts ar lokālo modeļu palīdzību:

```
movie(T;Y;D) :- { (T;Y;D) | r1(T;Y;D) }

european(D) :- { (D) | r1(T;Y;D) }

review(T;R) :- { (T;R) | r2(T;R) }
```

2.2.3.2. Pārformulēšana

Lai pārformulētu pieprasījumu lokālo modeļu valodā, ir tikai jāpārraksta pieprasījums, aizstājot globālās entītijas ar to definīcijām GAV (entītijām `movie` un `review`) un jāveic mainīgo substitūcija, un rezultātā tiek iegūta lokālo modeļu formālismā formulēta izteiksme.

2.2.3.3. Saites tipu starp lokālajiem un globālajiem modeļiem salīdzinājums

Tālāk ir salīdzinātas LAV un GAV pieeju priekšrocības un trūkumi.

GAV pieejas galvenā priekšrocība ir iespēja viegli pārformulēt globālajā modelī definētu pieprasījumu uz lokālajiem modeļiem, jo tas parasti aprobežojas ar mainīgo un formulu substitūciju. GAV lielākā problēma ir tā, ka sarežģīti veikt globālā modeļa izmaiņas datu avotu modeļu izmaiņu gadījumā. Pievienojot jaunu datu avotu, katrai globālā modeļa relācijai ir jāpievieno visas iespējamās formulas, kā atlasīt kortežus no lokālajiem modeļiem. Lai to izdarītu, ir jāpārbauda visas iespējamās esošo atlases nosacījumu un jauno nosacījumu kombinācijas un tas kļūst ļoti sarežģīti, ja ir daudz datu avotu [Lev00a].

LAV galvenā priekšrocība ir liela lokālo avotu autonomija, jo jauna avota pievienošana, noņemšana vai izmaiņa neiespaido globālo modeli, tā kā ir nepieciešams tikai noformulēt lokālos datu avotu modeļus globālā modeļa terminos. LAV pieeja atļauj formulēt precīzākus datu avotu ierobežojumus – lai to izdarītu, konkrētā datu avota relācijas formulā ir jāiekļauj tikai papildus ierobežojumi. LAV lielākais trūkums ir pieprasījumu izpilde, jo pieprasījumu pārformulēšana lokālo datu avotu terminos LAV modelī ir NP pilna problēma [Lev00a], kas pazīstama arī kā pieprasījumu pārformulēšana, izmantojot skatus [Lev00b].

Eksistē mēģinājumi apvienot LAV un GAV pieejas, lai no vienas puses viegli varētu pārformulēt pieprasījumus, kā tas ir GAV pieejā, un no otras puses būtu viegli modificēt lokālos modeļus, kā tas ir LAV pieejā. Tā ir radušās BAV, GLAV, BGLAV, P2P pieejas.

BAV *(Both as view)* [McBP03], GLAV *(Global and local as view)* [Cali03] un BGLAV *(Both global and local as view)* [XE02] pieejās tiek mēģināts, veidojot globālo modeli, veidot gan GAV translācijas aprakstus, gan LAV translācijas aprakstus, šādā veidā cenšoties saglabāt GAV formulējumu translācijas vieglumu un LAV elastīgumu attiecībā uz lokālo modeļu izmaiņām. P2P *(Point-to-point)* [HIST03] metodē tiek mēģināts iztikt vispār bez globālā modeļa, bet tiek veikta modeļu kartēšana starp tuvu stāvošiem modeļiem. P2P metodē, pievienojot jaunu datu avotu, tas tiek kartēts uz tam semantiski vistuvāko modeli. Lai veiktu pieprasījumus no vairākiem datu avotiem, nosaka un veic nepieciešamās translācijas, lai no pieprasījuma formulēšanas modeļa iegūtu datu avota formālismu. Piemēram, ja datu avotam D1 ir kartēšana ar D2 un D2 ir kartēšana ar D3, un tiek formulēts pieprasījums datu avotā D1, kas ir jāizpilda datu avotā D3, tad pieprasījumu dzinis atrod abas definētās translēšanas metodes un pārformulē, kā ir jātranslē no D1 formālisma uz D3 formālismu. Pēc tam rezultāts tāpat tiek translēts atpakaļ no D3 uz D1.

2.3. Modeļu integrācijas paņēmieni

2.3.1 Modeļu integrācijas veidi

Lai varētu veikt datu pieprasīšanu no vairākām sistēmām, kaut kādā mērā ir jāveic šo sistēmu integrācija. Literatūrā ir aprakstīti vairāki sistēmu integrācijas modeļi [SL90]. Daži no tiem ir DBPS un datu modeļu līmenī, citi atļauj integrējamo sistēmu plašāku autonomiju. Atkarībā no saistītajām IS piedāvātā autonomijas līmeņa, var izdalīt globālo modeļu integrāciju, federālo *(federated)* pieeju un multidatu bāzu sistēmas.

Globālo modeļu integrācijas pieejā tiek veidots viens globālais modelis (ER modelis, UML klašu diagramma u.c.), kurā ir apvienoti visu IS pilni datu modeļi. Globālā modeļa izveides laikā tiek mainīti lokālo datu bāzu modeļi, lai tie atbilstu globālajam modelim. Parasti tiek nodrošināta pieprasījumu izpilde šajā globālajā modelī [SP94].

Federālās pieejas gadījumā katra IS piedāvā tikai tās savas datu modeļa daļas, kuras tā ir ar mieru eksponēt publiski, un šīs pieejas gadījumā tiek veidoti pārejas modeļi no vienotā modeļa uz katras IS reālo modeli.

Multidatu bāzu sistēmās tiek izveidots pieprasījumu analīzes un izpildes dzinis, kurš pieprasījumu uz vairākām sistēmām analizē, sadala pa sistēmām, nosūta atsevišķos pieprasījumus uz konkrētām sistēmām, saņemtās atbildes apvieno un izrēķināto rezultātu nosūta atpakaļ pieprasītājam [LMR90, HBP94].

Starpnieksistēmās tiek izveidots pieprasījumu analīzes un izpildes dzinis, kā arī iepakotāji katram datu avotam, kas parasti nav DBPS ietvaros veidoti, bet ir specifiski veidota programmatūra. Starpnieksistēmas atļauj ļoti augstu sistēmu autonomiju, tanī pašā laikā piedāvājot lietotājam vienotu datu modeli.

Jāpiezīmē, ka jo ciešāka ir sistēmu integrācija, jo vienkāršāk ir strādāt ar integrēto sistēmu, tajā pašā laikā - jo ciešāka ir integrācija, jo vairāk autonomijas zaudē atsevišķās sistēmas.

2.3.1.1. Globālā modeļu integrācija

Globālā modeļu integrācija bija viena no pirmajām pieejām pasaulē, kas tika lietota, lai veiktu datu apmaiņu starp dažādām heterogēnām sadalītām datu bāzēm. Globālā modeļu integrācijā tiek izveidots viens kopīgs datu modelis, kur katra lokālā DB satur gabalu no šī modeļa bez izmaiņām. Šīs pieejas priekšrocība ir tas, ka lietotājam tiek piedāvāts vienots un nepretrunīgs skats un piekļūšana datiem. Daudzas datu bāzes tiek uztvertas kā viena kopīga datu bāze. Tomēr globālai modeļu integrācijai ir vairāki trūkumi:

- Modeļu integrācija ir sarežģīta un darbietilpīga, jo ir jāsaprot integrējamo datu bāzu semantika, līdz ar to šo procesu pilnībā automatizēt nav iespējams, lai gan eksistē rīki, kas atvieglo šo procesu;
- Modeļu integrācijas process nav vienreizējs pasākums. Integrētajiem modeļiem ir jāatspoguļo izmaiņas, kas ir notikušas integrējamajās datu bāzēs, līdz ar to modeļu integrācijas process lielākā vai mazākā mērā ir jāatkārto pēc katrām izmaiņām integrējamajās datu bāzēs;
- Modeļu integrācijas laikā var ciest atsevišķo datu bāzu autonomija. Visa konkrēto datu bāzu semantiskā informācija ir jāievieto kopīgajā repozitorijā, jo integrācijas procesā ir pilnībā jāpārzina datu bāzes semantika. Bieži vien, lai atvieglotu modeļu integrāciju, ir jāmaina lokālo datu bāzu datu modeļi;
- Modeļi parasti tiek integrēti pa soļiem, viens aiz otra, līdz ar to katrā solī tiek pielietota tikai daļa no visām zināšanām un pastāv iespēja, ka apvienotajā modelī pazudīs daļa no atsevišķo datu bāzu semantikas. Globālā modeļa

pareizību, t.i. to, vai tajā ir ietverta visa atsevišķo datu bāzu semantika, ir grūti pierādīt, jo semantika bieži vien ir atkarīga no izmantošanas konteksta.

2.3.1.2. Federālo datu bāzu sistēmas

Federālo datu bāzu sistēmas (FDBS) arhitektūras nolūks ir izvairīties no globālā modeļa izmantošanas un atļaut lokālajām datu bāzēm vairāk kontrolēt, kāda informācija tiek kopīgota (*shared*). Tas nozīmē, ka pastāv lielāka asociācijas autonomija starp lokālajām datu bāzēm, kas ļauj lietošanas kontroli decentralizēt un implementēt lokālajās datu bāzēs. FDBS integrācijas pakāpe ir atkarīga no konkrēto lietotāju prasībām, līdz ar to FDBS var būt gan cieši integrētas, gan arī vāji integrētas. Parasti FDBS satur kopējo datu modeli un iekšējo pieprasījumu valodu. FDBS var saturēt šādus modeļus un komponentes:

- *Lokālais modelis*: lokālās datu bāzes konceptuālais modelis, kas izteikts lokālās DBPS modeļa valodā (ER modelis, UML klašu diagramma u.c.),
- *Komponentes modelis:* lokālās DBPS modelis, izteikts FDBS modeļa valodā. Šāda translācija atvieglo modeļu integrāciju cieši integrētu FDBS gadījumā un skatu un pieprasījumu izstrādi vāji integrētu FDBS gadījumā. Modeļu translācijas laikā katrai lokālajai datu bāzei tiek izveidota kartēšana viens-pret-vienu starp lokālo modeli un komponentes modeli;
- *Transformācijas procesors:* transformācijas procesors izmanto kartēšanu viens-pret-vienu starp lokālo modeli un komponentes modeli, lai translētu pieprasījumus no iekšējās pieprasījuma valodas uz lokālo pieprasījumu valodu un datus no lokālā formāta uz kopējā datu modeļa formātu. Transformācijas procesors atrodas katrā lokālajā datu bāzē starp lokālo modeli un komponentes modeli;
- *Eksporta modelis:* katra datu bāze var noteikt, kuri datu objekti tiks kopīgoti ar citām FDBS datu bāzēm. Katra datu bāze satur pieejas kontroles informāciju (t.i. atsevišķai informācijai var piekļūt tikai atsevišķi FDBS lietotāji). Kopīgoto objektu modelis kopā ar pieejas kontroles informāciju veido eksporta modeli;
- *Filtrēšanas procesors:* filtrēšanas procesors izmanto pieejas kontroles informāciju, lai ierobežotu atļauto operāciju kopu, kuras var tikt izpildītas konkrētajā lokālajā datu bāzē. Tas veic pieejas kontroli un datu korektuma pārbaudes un atrodas starp komponentes modeli un eksporta modeli. Šāds procesors atrodas katrā lokālajā datu bāzē;

- *Federālais modelis:* šis modelis var būt vai nu statiski integrēts modelis, vai arī lietotāja specifisks skats uz vairākiem eksporta modeļiem. Ja FDBS ir cieši integrēta, tad statiski integrētu modeli uztur FDBS administrators. Ja FDBS ir vāji integrēta, tad lietotāja skatu uztur pats lietotājs. Katrai FDBS lietotāju klasei var tikt izveidots savs federālais modelis;

- *Konstruēšanas procesors* izmanto informāciju no federālās vārdnīcas un veic pieprasījumu dekompozīciju no federālā modeļa uz vienu vai vairākiem eksporta modeļiem un nosūta to izpildei lokālajam datu bāzēm, kā arī pēc rezultātu saņemšanas veic datu apvienošanu. Šis procesors atrodas FDBS;

- *Ārējais modelis:* ja federālais modelis ir ļoti liels un sarežģīts, tad ārējo modeli izmanto, lai konkrētai lietotāju klasei piedāvātu specifiski nepieciešamo federālā modeļa daļu. Ārējais modelis var saturēt papildus integritātes ierobežojumus un pieejas kontroles informāciju. Vāji saistītās FDBS šis modelis nav nepieciešams, bet cieši saistītās FDBS ārējais modelis ir ļoti svarīgs. Ārējais modelis var atšķirties no federālā modeļa, tāpēc var būt nepieciešams komandu un datu translācijas procesors no ārējā modeļa uz federālo modeli un atpakaļ;

- *Datu vārdnīca:* satur ārējo, federālo un eksporta modeļus. Cieši integrētā sistēmā datu vārdnīca reizēm satur arī komponenšu un lokālo modeļu objektus. Datu vārdnīca satur arī kartējumus starp modeļiem. Datu vārdnīca var saturēt arī citu informāciju, piemēram, pieprasījumu optimizēšanai nepieciešamo statistiku un heiristikas, funkcijas datu translēšanai starp dažādām mērvienībām un datu formātiem, lokālo bāzu tīkla adreses, komunikācijas protokolus utt.

FDBS pieclīmeņu arhitektūra dod iespēju efektīvāk uzturēt sadalīšanu, heterogenitāti un autonomiju. Komponenšu, eksporta un federālie modeļi tiek glabāti kopējā komponenšu datu bāzē, ko kontrolē un pārvalda komponenšu datu bāzes administrators. Komponenšu datu bāzes struktūra ir ļoti sarežģīta, jo satur vairāku tipu modeļus, kartējumus starp tiem, kā arī citu informāciju, un katra komponenšu datu bāzes realizācija var būt savādāka.

FDBS ir jāspēj komunicēt savā starpā šādu operāciju veikšanai:

- Apmaiņa ar datiem. Katrai lokālajai datu bāzei ir jāspēj piekļūt pie citu lokālo datu bāzu kopīgotajiem datiem. Tas ir galvenais FDBS pastāvēšanas iemesls, līdz ar to labs datu apmaiņas mehānisms ir obligāts;

- Kopīgas transakcijas. Var gadīties, ka tiek atļautas datu mainīšanas operācijas, kurām jānotiek starp vairākām lokālajām datu bāzēm;

- Kopējas aktivitātes. Lai gan nav lokālo sistēmu darbības centralizētas kontroles, tomēr reizēm ir nepieciešams veikt kopīgas aktivitātes, piemēram, izpildīt datu pieprasījumu no vairākām lokālajām datu bāzēm.

FDBS iedala vāji integrētās un cieši integrētās.

Vāji integrētas FDBS

Vāji integrētās FDBS lietotāji paši izveido un uztur federālos modeļus. FDBS neuzspiež nekādas papildus kontroles vai nosacījumus. Federālos modeļus veido kā skatus uz interesējošajiem eksporta modeļiem, kas nozīmē, ka katram lietotājam ir jābūt zināšanām par eksporta modeļu saturu un semantiku, lai varētu izveidot skatus. Federālie modeļi ir dinamiski un tos var jebkurā brīdī pievienot vai iznīcināt. Parasti tiek uzturēti vairāki federālie modeļi vienlaicīgi. Šādas sistēmas parasti satur ļoti autonomas lasāmrežīma datu bāzes un parasti neuztur skatu atjaunināšanu. Vāji integrētām FDBS ir šādas priekšrocības:

- Dažādi FDBS lietotāji var vieniem un tiem pašiem objektiem un atribūtiem piešķirt dažādu semantiku, lietojot dažādus objektu un atribūtu vārdus;
- Vāji integrētās FDBS ir vieglāk ieviest lokālo datu bāzu modeļu izmaiņas, jo atsevišķo skatu pārveide vai jaunu izveide ir vieglāka nekā visa globālā modeļa pārveidošana vai izveidošana no nulles. Lai interesenti varētu vieglāk konstatēt izmaiņas, ir nepieciešams mehānisms automātiskai izmaiņu nosūtīšanai.

Jāmin arī daži vāji integrētu FDBS trūkumi:

- Ja divi vai vairāk lietotāji grib piekļūt līdzīgai informācijai, tad katrs no tiem izveido savus skatus un kartējumus, nezinot, ka to kāds cits jau ir izdarījis. Līdz ar to pastāv iespēja, ka viens un tas pats darbs tiks izdarīts vairākas reizes. Ja lokālo modeļu ir daudz, tad var rasties grūtības ar to saprašanu un korektu integrēšanu;
- Tā kā relācijas starp objektiem var izteikt dažādu semantiku, tad skatu atjaunināšana ir apgrūtinoša un parasti pat neiespējama.

Cieši integrētās FDBS

Cieši integrētajās FDBS administratoriem ir pilna kontrole pār federālo modeļu izveidi un uzturēšanu, kā arī pār pieeju eksporta modeļiem. Tā mērķis ir nodrošināt atrašanās, replicēšanas un translēšanas caurspīdīgumu. Cieši integrētās FDBS var uzturēt vienu vai vairākus federālos modeļus. Viens federālais modelis palīdz uzturēt vienotu datu semantisko interpretāciju. Vairākus federālos modeļus ir grūtāk uzturēt, jo ierobežojumus

no vairākām lokālajām datu bāzēm ir grūtāk ievērot visos federālajos modeļos, līdz ar to var parādīties nekonsistence ar datu semantiku.

Viena federālā modeļa izveide patiesībā ir globālā modeļu integrācija. Šādā gadījumā vismaz daļēji var tikt uzturēta skatu atjaunināšana, ja FDBS administratori pilnībā pārzina lokālās datu bāzes, izprot un definē visus kartējumus un modeļu integrācijas laikā atrisina visus semantiskos konfliktus. Cieši integrēto FDBS trūkumi ir:

- Tā kā FDBS un komponenšu datu bāzu administratori vienojas veidot eksporta modeļus, tad šīs vienošanās laikā FDBS administratoram var būt tiesības lasīt komponentes modeļa metadatus bez pieejas datiem, kas pārkāpj autonomijas principus,
- Kad ir izveidots federālais modelis, tas ir statisks, t.i. tiek mainīts ļoti reti. Kad tiek pamainīti eksporta vai komponenšu modeļi, tad katram federālajam modelim integrācija ir jāveic no pašiem pamatiem.

2.3.1.3. Multidatu bāzu sistēmas

Multidatu bāzu sistēmas ir paredzētas gadījumiem, kad eksistē vairākas datu bāzes, bet neeksistē vienots datu modelis. Piemēram, daudzām organizācijām jau eksistē kaut kādas datu bāzes un ir nepieciešamība apvienot informāciju no šīm dažādajām datu bāzēm. Viens no iespējamajiem risinājumiem ir multidatu bāzu sistēmas (MDBS). Eksistējošās datu bāzes tiek apvienotas multidatu bāzes sistēmā bez modifikācijām. Faktiski šajā gadījumā nekāda modeļu integrācija nepastāv – ir tikai tehniska iespēja veikt pieprasījumu no vairākām tabulām vairākās datu bāzēs vienā pieprasījumā un multidatu bāzu sistēma māk šādus pieprasījumus izpildīt.

MDBS ir sistēmas, kuras sastāv no globālās datubāzes pārvaldības sistēmas (GDBPS) un vairākām lokālajām datu bāzēm (LDB). Katrai LDB ir sava lokālā DBPS, kas darbina lokālo datu bāzi. GDBPS satur savu DBPS un satur globālo DB modeli, kas tiek konstruēts, apvienojot LDB modeļus. Globālie pieprasījumi tiek prasīti GDBPS, lokālos pieprasījumus var prasīt tieši konkrētai LDB. GDBPS ir atbildīga par globālo pieprasījumu apstrādi, to sadali apakšpieprasījumos konkrētajām LDBS, apakšpieprasījumu izpildes koordināciju un rezultātu apvienošanu. MDBS piemīt šādas īpašības:

- LDBS ir iepriekš izveidotas datubāzu sistēmas, tāpēc parasti tās ir heterogēnas visdažādākajos līmeņos;
- GDBPS pieejamie resursi ir limitēti. Tas tā var būt vairāku iemeslu dēļ:

- LDBS nesniedz kādu daļu no GDBPS prasītās informācijas, kas nepieciešama globālai pieprasījumu izpildes optimizācijai (piemēram, pieprasījumu izpildes izmaksas);
- LDBS nenodrošina piekļuvi savām iekšējām struktūrām (piemēram, indeksu kokiem). Līdz ar to GDBPS ir jānosūta savi pieprasījumi uz LDBS kā SQL pieprasījumi, tāpēc LDBS tie ir atkārtoti jāanalizē un nav iespējas tiešā veidā izmantot LDBS pieprasījumu izpildes mehānismu.

- LDBS var saņemt lokālos pieprasījumus, apejot GDBPS. Šie lokālie pieprasījumi izpildes laikā var ietekmēt globālo pieprasījumu izpildi, tā sarežģījot globālo pieprasījumu optimizēšanu.

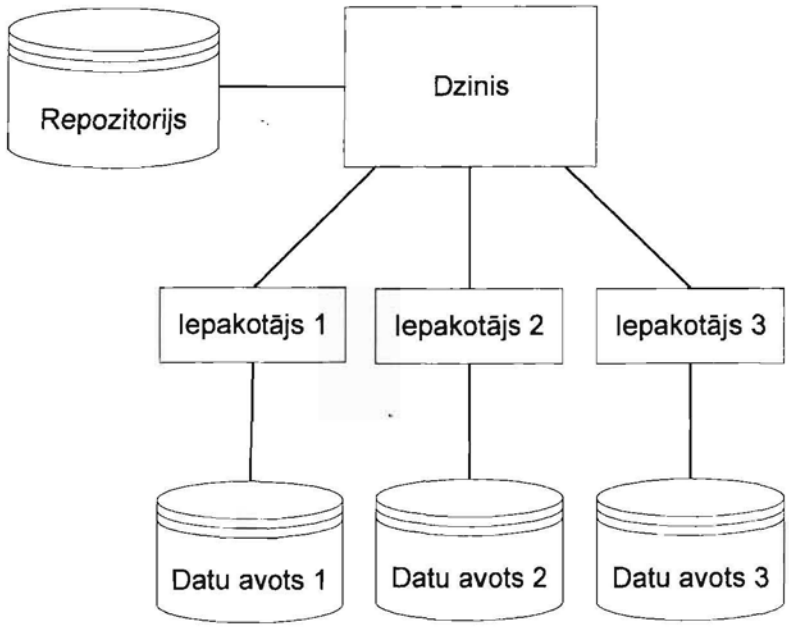
Galvenais MDBS trūkums ir tas, ka lietotājam pašam ir jāatrod nepieciešamo datu atrašanās vieta un pieprasīšanas veids, kā arī pašam jāmāk veikt dažādo datu avotu "integrāciju", formulējot datu pieprasījumu.

2.3.1.4. Starpnieksistēmas

Starpnieksistēmas *(mediated systems)* integrē dažādus heterogēnus datu avotus, piedāvājot virtuālu skatu uz tiem. Lietotājs veic pieprasījumus, neinteresējoties par datu avotu atrašanās vietu, datu modeļiem un pieejas metodēm, jo sistēma piedāvā globālu skatu uz datiem un lietotājs veic pieprasījumu šajā globālajā modelī. Starpnieksistēmas no globālās modeļu integrācijas pieejas un FDBS atšķiras ar šādām īpašībām [SL90]:

- Starpnieksistēmas arhitektūra var saturēt komponentes, kas nav datu bāzes;
- Datu avotu iespējas apstrādāt pieprasījumus var būt ierobežotas, un datu avoti var vispār neuzturēt SQL pieprasījumus;
- Datu avoti ir pieejami tikai lasīšanas režīmā, atšķirībā no FDBS, kur datu avoti ir pieejami arī izmaiņām (tas ir tāpēc, ka datu avoti starpnieksistēmās ir daudz autonomāki);
- Starpnieksistēmās datu avoti ir pilnībā autonomi, tas nozīmē, ka pielikt un noņemt kādu datu avotu ir viegli.

Attēlā Zīm. 1 ir redzama tipiska starpnieksistēmas arhitektūra, kuras galvenās sastāvdaļas ir starpnieks jeb dzinis un kur katram datu avotam ir savs iepakotājs.



Zīm. 1 Starpnieksistēmas arhitektūra

Starpnieks veic šādas darbības:

- Saņem no lietotāja datu pieprasījumus, kas formulēti kopējā modeļa valodā;
- Sadala šos pieprasījumus individuālās daļās, balstoties uz datu avotu aprakstiem;
- Optimizē pieprasījumu izpildes plānu;
- Nosūta pieprasījumus datu avota iepakotājiem, kas šos pieprasījumus pārtaisa par pieprasījumiem lokālajā modelī. Tad starpnieks saņem no iepakotājiem atbildes, tās kombinē kopā vienā atbildē un nosūta to lietotājam.

Iepakotāja uzdevums ir noslēpt no starpnieka tehniskās detaļas par piekļūšanu konkrētajam datu avotam. Iepakotājs saņem no starpnieka pieprasījumu starpnieka formātā, konvertē to datu avotam saprotamā formātā un nosūta datu avotam izpildei. No datu avota saņemto atbildi iepakotājs pārveido starpniekam saprotamā formātā, ja nepieciešamas veic papildus datu apstrādes darbības (piemēram, saņemto datu atlasi pēc kāda kritērija) un rezultātu nosūta starpniekam. Pasaulē ir izveidotas daudzas starpnieksistēmas, piemēram TSIMMIS [HGIP95], DISCO [TRV96], GARLIC [HMNT99], tomēr jāpiezīmē, ka lielākoties tās visas ir eksperimentālas izstrādnes.

2.3.2 Starpmodeļu relāciju identifikācija

Starpmodeļu relāciju identifikācijas mērķis ir identificēt relācijas starp dažādu modeļu objektiem un klasificēt tās. Starpmodeļu relāciju identifikācija var notikt vai nu balstoties uz konceptuālajiem modeļiem, vai arī balstoties uz sistēmas datiem.

Ja starpmodeļu relācijas tiek identificētas balstoties uz konceptuālajiem modeļiem, tad parasti tas ir process, kas sastāv no divām daļām:

1. Identificēt, kuri objekti ir saistīti,
2. Klasificēt attiecības starp šiem objektiem.

Pirmajā daļā tiek izmantotas integrējamo modeļu semantikas zināšanas. Acīmredzamākais veids ir, ka cilvēks definē, kuri objekti un kādā veidā ir saistīti. Tajā pašā laikā pastāv arī mēģinājumi izveidot automatizētas metodes, kas ir balstītas uz modeļos izmantotajām tipveida konstrukcijām. Tiek pieņemts, ka savā starpā ir saistīti objekti, kuru relācijas ir ar līdzīgām kardinalitātēm, atribūti ir no vieniem un tiem pašiem domēniem, kuriem ir līdzīgi ierobežojumi (*constraints*) un līdzīgi datu integritātes nosacījumi. Literatūrā iesaka analizēt objektus pēc to līdzības, piemēram, entītijām var analizēt lomas, nosaukumus, atribūtus utt., relācijām var analizēt nosaukumus, kardinalitātes, saistīto entītiju līdzību.

Otrajā daļā tiek klasificētas atrastās attiecības. Dažādi autori iesaka dažādas klasifikācijas. Piemēram, Larsons [LNEM89] piedāvā šādus četrus atribūtu attiecību tipus:

```
a EQUAL b,

a CONTAINS b,

a CONTAINED-IN b,

a OVERLAP b.
```

Tiek definēti arī četri attiecību tipi starp entītijām un relācijām, kas balstās uz atslēgas atribūtu attiecībām:

```
A EQUAL B,

A CONTAINED-IN B,

A OVERLAP B,

A DISJOINT B.
```

Uz datiem balstītas starpmodeļu relāciju identifikācijas mērķis ir identificēt, kuri ieraksti attiecas uz vienu un to pašu reālās dzīves objektu. Visprecīzākā identifikācija ir

iespējama tad, ja entītijas satur primāro atslēgu – tad ieraksti ar vienādām atslēgas vērtībām attiecas uz vienu un to pašu reālās dzīves objektu. Reizēm ir vēlams salīdzināt ne tikai atslēgas atribūtu vērtības, bet arī citu atribūtu vērtības. Tomēr šādos gadījumos ir jānosaka, ko darīt gadījumā, ja atslēgas atribūti sakrīt, bet citi atribūti, kuriem vajadzētu būt vienādiem, atšķiras. Piemēram, ir divas sistēmas, kurās ir informācija par personu – Iedzīvotāju reģistrs un Uzņēmumu reģistrs. Abās šajās sistēmas tiek glabāta informācija par personas kodu, vārdu un uzvārdu. Kad persona apprecas un nomaina uzvārdu, šīs izmaiņas tiek fiksētas Iedzīvotāju reģistrā, taču Uzņēmumu reģistram šādas izmaiņas nav jāfiksē. Problēma: kā rīkoties šādā gadījumā - mainīt UR datus, mainīt IR datus vai nemainīt neko? Ko darīt gadījumā, ja persona ir mainījusi uzvārdu un šī informācija nav nonākusi IR, taču ir nonākusi UR? Šāda tipa problēmas tiek risinātas katrā datu apmaiņas izveides gadījumā individuāli.

2.3.3 Modeļu integrācijas process

Modeļu integrācija ir process, kura rezultātā no apvienojamo datu bāzu modeļiem tiek ģenerēti viens vai vairāki apvienotie modeļi.

Modeļu integrācija sākotnēji var šķist līdzīga skatu integrācijai, ko pielieto datu bāzu projektēšanā, tomēr pastāv vairākas būtiskas atšķirības. Skatu integrācija ir process, kura laikā no dažādiem lietotāju skatiem uz vienu un to pašu sistēmu tiek veidota kopēja datu bāze. Šī integrācija tiek veikta sistēmas projektēšanas fāzē. Sākotnēji ir doti daudzi lietotāju skati uz sistēmu, kuri tiek integrēti vienā kopīgā datu bāzes modelī, kas nozīmē, ka skatu integrācija ir lejupejošs vienas datu bāzes ģenerēšanas process. Savukārt modeļu integrācija ir augšupejošs process, kura laikā no vairākiem eksistējošiem datu bāzu modeļiem tiek veidots viens kopīgais modelis, kurā integrē šīs datu bāzes.

Skatu integrācijā lietotāji definē skatus, lietojot vienu kopēju datu modeli. Modeļu integrācijā ir vairākas, parasti heterogēnas datu bāzes ar dažādiem datu modeļiem, kas var dažādi prezentēt datus.

Skatu integrācija tiek veikta laikā, kad vēl nav gatava datu modeļa – tas top šī procesa izstrādes laikā. Modeļu integrācija notiek, integrējot jau eksistējošas datu bāzes, līdz ar to iespējas variēt ir ierobežotas, jo izveidotie modeļi nedrīkst konfliktēt ar esošo datu bāzu semantiku. Skatu integrācijā ir lielāka brīvība semantikas interpretācijā.

2.3.3.1. Modeļu integrācijas ietvars

Modeļu integrācijas soļi

Parasti modeļu integrācijas metodoloģijas var sadalīt četrās fāzēs [RR95]:

1. Modeļu translācija. Šajā fāzē eksistējošo datu bāzu modeļi tiek translēti uz vienu kopēju datu bāzes aprakstīšanas formālismu. Parasti lietotie

formālismi ir ER modelis, UML modelis un pēdējā laikā popularitāti ieguvusī datu aprakstīšana ar predikātu rēķiniem. Ja, piemēram, viena ir relāciju datu bāze un otra ir objektorientēta datu bāze, tad lokālos modeļus translē uz kādu kopīgu formālismu. Bieži vien šo translāciju var veikt, izmantojot kāda rīka palīdzību, tomēr parasti ir rezultāts ir manuāli jākoriģē. Jāņem vērā, ka lietojamajam modelim ir jāatbilst šādiem parametriem:

- Jaunizveidotajam modelim ir jāsatur visa semantika, kas ir katrā konkrētajā datu bāzē;
- Jābūt iespējai translēt komandas no jaunizveidotā modeļa uz komandām sākotnējā modelī.

2. Starpmodeļu attiecību ģenerēšana. Šīs fāzes mērķis ir identificēt saistības starp objektiem dažādos modeļos. To var veikt, analizējot objektu semantiku un identificējot saistību, kas balstās uz šo semantiku. Analīze tiek balstīta uz entītiju, atribūtu, domēnu semantiku, kā arī uz zināšanām par konkrētajām aplikācijām un to pielietojumu. Rezultātā tiek iegūtas attiecības starp dažādu modeļu objektiem. Piemēram, entītija KLIENTS vienā modelī ir virsklase entītijai NOGULDĪTĀJS otrā modelī (ja runa ir par bankas aplikācijām). Var identificēt arī abu klašu kopīgos atribūtus;

3. Integrētā modeļa ģenerēšana. Šajā fāzē tiek izmantotas iepriekšējā fāzē izveidotās attiecības starp dažādiem objektiem, lai izveidotu kopīgu modeli. Šī procesa laikā ir jāņem vērā dažādas heterogenitātes un tās jāatrisina. Tiek izšķirtas piecas galvenās heterogenitāšu kategorijas:

   - domēnu definīciju,
   - entītiju definīciju,
   - datu vērtību,
   - abstrakcijas līmeņu
   - modeļu.

   Integrētā modeļa ģenerēšanas laikā šīs heterogenitātes tiek atrisinātas un tiek izveidots integrētais modelis, kas lietotājam noslēpj heterogenitātes;

4. Modeļu kartēšanas ģenerēšana. Šis solis parasti tiek veikts līdz ar iepriekšējo soli un tā rezultātā tiek izveidota atbilstība starp objektiem integrētajā modelī un oriģinālajos modeļos.

Īāpiezīmē, ka šos soļus var izpildīt interaktīvi, lai atrisinātu radušās heterogenitātes.

2.3.3.2. Modeļu translēšana

Integrētā modeļa izveides procesa laikā notiek vairākas modeļu pārveides – no lokālā modeļa uz eksporta modeli un pēc tam uz globālo (integrēto) modeli. Šīs pārveides sauc par modeļu translēšanu. Modeļu translēšanas laikā var būt nepieciešama papildus semantiskā informācija, kas nav ietverta pašā modelī.

Ja sistēmā tiek izmantota arī kopējā datu manipulācijas valoda (DML), tad ir nepieciešamas veikt arī valodu translēšanu. Pieprasījumi, kas tiek veikti kopējā DML, tiek sadalīti apakšpieprasījumos lokālajām datu bāzēm, un šie apakšpieprasījumi tiek translēti uz attiecīgās lokālās datu bāzes valodu. Lokālā datu bāze izpilda tai piesūtītos pieprasījumus un atgriež rezultātus, kas tiek translēti uz kopējiem terminiem.

Tālāk tiks apskatīti dažādi modeļi, kurus var izmantot integrētā modeļa izveidē – relāciju modelis, ER modelis, objektorientētais modelis un predikātu rēķinu izmantošana.

Modeļu translēšana uz relāciju modeli

Relāciju modelis ir viens no agrāk visbiežāk lietotajiem kopējiem modeļiem, lai definētu multidatu bāzu eksporta modeļus [DAT87]. Relāciju modelim ir vairākas priekšrocības – augsta līmeņa kopu operāciju izmantošana datu manipulācijai, kas atvieglo datu apstrādi, viegla relāciju dekompozīcija un apvienošana. No otras puses relāciju modeļa semantika ir nabadzīga un bieži vien neļauj izteikt visas zināmās modeļu sakarības. Lai varētu izteikt vairāk sakarību, ir jāizvēlas semantiski bagātāki modeļi – ER modelis, objektorientēts modelis u.c.

Modeļu translēšana uz ER modeli

ER modelis [Che76] ir viens no biežāk lietotajiem modeļiem multidatu bāzu sistēmās. Literatūrā var atrast diezgan precīzus algoritmus modeļu translēšanai starp ER modeli un citiem modeļiem, piemēram, CODASYL, IMS, relāciju, objektorientēto u.c.

Lai veiktu translāciju starp relāciju modeli un ER modeli, ir nepieciešams pārveidot tabulas par entītijām un relācijām. Ir jāatšķir tabulas, kas reprezentē entītijas, no tabulām, kas reprezentē relācijas. Bieži vien tas nav tik viegli izdarāms, jo vienu un to pašu semantisko jēgu var izteikt dažādos veidos, piemēram, vai tabula reprezentē atsevišķu entītiju, vai arī tabula reprezentē relāciju ar papildus atribūtiem? Ja ir tikai relāciju modelis, tad bez papildus semantikas zināšanām var būt grūti noteikt relāciju kardinalitātes (viens-pret-vienu, viens-pret-daudziem, daudzi-pret-daudziem). Ir nepieciešamas papildus semantiskās informācijas zināšanas, lai varētu veikt korektu translāciju.

Modeļu translēšana uz objektorientēto modeli

Objektorientēts modelis reizēm tiek lietots kā integrēto modeļu formālisms. Modeļu translēšanas laikā tiek lietots metamodelis, kurā glabājas papildus semantiskā informācija par lokālajiem datu modeļiem un saitēm starp tiem [CT92]. Lai pārveidotu ER modeli uz objektorientētu modeli, var pielietot divpakāpju translēšanu:

1. Katru entītiju pārveido par objektorientētā modeļa klasi,
2. Katru relāciju pārveido par ziņojumu starp šīm klasēm.

Modeļu translēšana uz predikātu rēķinu modeli

Ļoti lielu popularitāti pēdējā laikā iegūst predikātu rēķinu izmantošana, jo modelī, kas ir veidots izmantojot predikātu rēķinus un valodu DATALOG, var izteikt papildus datu avotu ierobežojumus, kurus nevar izteikt ar citiem formālismiem, tādiem kā ER modeli vai relāciju modeli.

2.3.3.3. Modeļu kartēšanas ģenerēšana

Modeļu kartēšanas ģenerēšana tiek veikta vienlaicīgi ar modeļu translēšanu un integrētā modeļa ģenerēšanu. Modeļu kartējums ir nepieciešams, lai varētu korekti translēt pieprasījumus no integrētā modeļa uz lokālajiem modeļiem un rezultātus translēt atpakaļ no lokālā modeļa uz integrēto modeli. Ja tiek pielietota modeļu restrukturizācijas pieeja, tad kartējums parasti tiek ģenerēts apvienotā modeļa izveides laikā un glabājas globālajā direktorijā/ vārdnīcā. Ja tiek pielietota skatu ģenerācija, tad kartējums tiek veidots kā daļa no skata pieprasījuma un arī tiek saglabāts globālajā katalogā.

2.3.3.4. Automatizēta modeļu ģenerēšana

Modeļu integrācija ir sarežģīts un darbietilpīgs process, līdz ar to aktuāla ir automatizēta modeļu integrācija. Tomēr pilnīga integrācijas procesa automatizācija nav iespējama, jo integrācijas procesa laikā jāsaprot integrējamo datu bāzu semantika, kā arī integrācijas rezultāts ir atkarīgs no tā, kā tas tiks izmantots. Līdz ar to ir nepieciešama cilvēka iejaukšanās modeļu integrācijas procesā. Tomēr automatizāciju var izmantot, lai veiktu vienkāršus rutīnas darbus un automatizētu pašu procesu. Ir izstrādāti vairāki rīki modeļu integrācijai, tomēr tie ir vairāk domāti zinātniskiem pētījumiem un izraisa akadēmisku interesi, nevis ir praktiski pielietojami. Parasti šajos rīkos kā ieejas dati tiek ņemti integrējamie modeļi. Daži rīki nodrošina tikai iespēju ērtāk norādīt ekvivalenci starp modeļu objektiem.

Daži rīki veic automātisku ekvivalences meklēšanu, balstoties uz dažādām metodoloģijām. Piemēram, rakstā [SLCN88] ir aprakstīts rīks, kurš veic analīzi, kā rezultātā lietotājam tiek iedots saraksts ar objektu pāriem un iespējamā attiecība starp

tiem (vienāds, ietilpst, satur, sadalīts-bet-integrējams, sadalīts-nav-integrējams). Lietotājs var norādīt, kādas ir patiesās attiecības starp objektiem.

Rīks, kas aprakstīts [RR95], veic analīzi, kas balstās uz entītijām, atribūtiem un relācijām. Rezultātā objektu pāriem tiek izdoti divi indeksi – līdzības koeficients un atšķirības koeficients. Līdzības koeficients var būt robežās starp 0 un 1 un atšķirības koeficients var būt robežās starp –1 un 0. Augsts līdzības koeficients norāda, ka objekti varētu būt saistīti, augsts atšķirības koeficients norāda, ka objekti varētu nebūt saistīti.

2.4. Semantiskā heterogenitāte un tās atrisināšana

Semantiku var definēt kā saiti starp simboliem un zīmēm un to nozīmi jeb jēgu [HM93]. Semantiskā heterogenitāte norāda uz dažādu datu nozīmi un lietojumu, kas apgrūtina saišu identifikāciju starp līdzīgiem vai vienādiem objektiem dažādās sistēmās.

Šajā pašā minētajā rakstā tiek definēti sekojoši heterogenitātes līmeņi:

1. Metadatu valoda. Katra lokālā DB var lietot atšķirīgu metadatu modeli un dažādas datu definēšanas valodas. Piemēram, viena sistēma datu manipulācijas aprakstam lieto SQL DDL, cita datu bāze izmanto funkcionālās apraksta valodas;

2. Ontoloģija/ terminoloģija. Ontoloģija – dažādu jēdzienu/ konceptu apraksts. Šajā līmenī heterogenitāte notiek, ja eksistē pretrunas ontoloģijās, piemēram, viens jēdziens apzīmē dažādas lietas dažādās sistēmās;

3. Metadatu specifikācija - kad eksistē pretrunas konceptuālajos modeļos, piemēram, vienā sistēmā objektam *Grāmata* ir atribūts *Autors*, kurā tiek ierakstīts autora vārds un uzvārds, vairāku autoru vārdus atdalot ar komatu, bet citā sistēmā ir objekts *Grāmata*, kas ar saiti daudzi-pret-daudziem ir saistīts ar objektu *Autori*;

4. Objektu salīdzināmība - kad ekvivalentus/ saistītus objektus nevar viegli identificēt, piemēram, atrast vienu un to pašu preci divos interneta veikalos bieži vien ir ļoti sarežģīti;

5. Zema līmeņa datu formāti - kad ekvivalentiem datu bāzu objektiem ir atšķirīgi atribūtu datu tipi, piemēram, vienā sistēmā personas kods glabājas kā simbolu virknīte, bet citā sistēmā - kā skaitlis;

6. Rīki - kad vairākas sistēmas lieto dažādas DBPS, līdz ar to piedāvājot atšķirīgas iespējas sistēmas līmenī, piemēram, viena DBPS uztur automātisku primārās atslēgas definēšanu, cita sistēma to neuztur.

Rakstā [SK93] apskatītie semantiskie konflikti ir sadalīti divās lielās grupās: domēnu definēšanas konflikti un entītiju definēšanas konflikti.

2.4.1 Domēnu definēšanas konflikti

Pie šiem konfliktiem var pieskaitīt nosaukumu konfliktus (sinonīmi un homonīmi), datu tipu konflikti, datu izmēru nesakritība, datu precizitāte, noklusētā vērtība un atribūtu integritātes ierobežojumi:

- Par nosaukumu konfliktiem sauc gadījumus, kad semantiski vienādi objekti tiek saukti dažādi (sinonīmi), vai arī semantiski dažādi objekti tiek saukti dažādi (homonīmi). Piemēram, atribūts GARUMS transportlīdzeklim un mākslas filmai nozīmē absolūti dažādas lietas. Lai šo problēmu risinātu, ir nepieciešams veikt translāciju no datu avota apzīmējumiem uz kopējiem apzīmējumiem;

- Ja dati tiek glabāti dažādos formātos – piemēram, Personas kods ar un bez svītriņas pa vidu, vai arī vieniem un tiem pašiem datiem ir dažādi datu tipi – piemēram, dzimums vienā sistēmā ir kodēts ar burtiem S/V, bet citā sistēmā ar skaitļiem ½, tad runā par datu tipu konfliktiem;

- Datu izmēru nesakritība iestājas gadījumā, ja vienus un tos pašus datus dažādi interpretē dažādās sistēmās. Galvenokārt tas notiek ģeogrāfisku, kultūras un organizatorisku atšķirību dēļ. Piemēram, atribūts alga ar vērtību 100 Latvijā tiek interpretēts kā 100 Lati, bet Vācijā kā 100 Eiro. Līdzīgi ir ar atšķirībām garuma un svara mērvienībās valstīs, kur ir SI sistēma, un Anglijā un ASV, kur ir galoni un jūdzes. Šādas atšķirības var novest pie diezgan bēdīgām sekām. Ir zināms gadījums, kad ASV lidmašīna ceļā no Francijas uz ASV bija spiesta veikt ārkārtas nosēšanos degvielas trūkuma dēļ. Izrādās, ka no ASV uz Franciju bija atsūtīts pieprasījums pēc lidmašīnai nepieciešamās degvielas galonos, bet Francijā tas tika interpretēts kā litri. Tā kā 1 galons ir aptuveni 4 litri, tad lidmašīnas bākas tika aizpildītas tikai par vienu ceturtdaļu;

- Datu precizitātes problēmas ir gadījumos, ja vienu un to pašu lielumu dažādās sistēmās glabā ar dažādu precizitāti;

- Atslēgu atšķirība rodas, ja vieniem un tiem pašiem datiem dažādās sistēmās ir dažādas primārās atslēgas. Tā kā sistēmās nav vienas kopējas primārās atslēgas, tad var rasties problēmas atlasīt vienus un tos pašus datus no šīm sistēmām;

- Apvienojumu savietojamības problēma rodas, ja ekvivalentām relācijām ir dažāds atribūtu skaits un nepastāv iespēja veikt kartēšanu starp atribūtiem;

2.4.2 Entītiju definēšanas konflikti

Entītiju definēšanas konflikti sadalās atslēgu ekvivalencē, apvienojumu savietojamība, modeļu izomorfismā un neesošo datu vienumu problēmā:

- Modeļu izomorfisms rodas, ja semantiski identiskām entītijām ir dažāds atribūtu daudzums, piemēram, vienā sistēmā ir lauks *vārds*, bet otrā sistēmā ir lauki *vārds* un *uzvārds*. Dažādi datu modeļi var rasties, vienus un tos pašus datus attēlojot dažādi. Piemēram, adresi var vienā sistēmā attēlot kā vienu vai vairākus atribūtus, bet citā sistēmā tā ir atsevišķa entītija. Var gadīties, ka vienā datu bāzē tiek lietotas konstrukcijas, kas nav paredzētas citā DBPS, piemēram, mantojamība.

- Ja semantiski identisku datu atribūti dažādos avotos ir atšķirīgi vai iztrūkst, tad var runāt par datu nekonsistenci vai neesošo datu vienumu problēmu.

- Var atšķirties integritātes nosacījumi. Piemēram, vienā DBPS var būt specificēts nosacījums *constraint cascade delete*, ka, izdzēšot pēdējā klases skolēna datus, arī klases dati tiek izdzēsti, bet citā sistēmā klases dati, veicot šādu pašu darbību, saglabājas. Sistēmā var būt lietoti trigeri, kas veic vēl sarežģītākas darbības.

- Abstrakcijas līmeņa nesavietojamība rodas ģeneralizācijas un agregācijas pielietošanas rezultātā. Piemēram, entītija *publikācija* vienā sistēmā tiek attēlota kā tabula

  ```
  Publ(publ#, autors, nosaukums,...)
  ```

  bet otrā sistēmā kā divas tabulas

  ```
  Gramata(isbn, autors, nosaukums,...)

  Zurnals(issn, autors, nosaukums,...)
  ```

  Pirmajā sistēmā tiek lietots augstāks abstrakcijas līmenis.

- Modeļu nesavietojamības problēma rodas, kad dati vienā sistēmā atbilst metadatiem otrā sistēmā. Šādu problēmu atrisināšana ir netriviāla, jo nepieciešams pieprasījumā vienlaicīgi vērsties gan pie datiem, gan pie metadatiem. [KLK91]

2.4.3 Heterogenitāšu atrisināšanas metodes

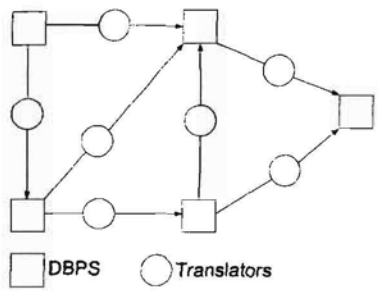
[ERS99] ir aprakstīti četri sadarbības veidi, kas var tikt lietoti heterogenitāšu atrisināšanai – translāciju pieeja, integrētā pieeja, pilnībā decentralizētā pieeja un brokeru bāzētā pieeja.

2.4.3.1. Translācijas pieeja

Translācijas pieeju Zīm. 2 parasti lieto darba plūsmu scenārijos. Tā tiek pielietota gadījumos, kad ir precīzi zināms, kādi dati un no kādām sistēmām būs nepieciešami. Tāpat arī konkrēto LDBPS modeļiem (eksporta modeļiem vai datu pieejas saskarnēm) ir jābūt stabilām un zināmām. Šie nosacījumi parasti izpildās dažādos Megasistēmas reģistros. Šādai pieejai ir divas pozitīvas īpašības:

1. Specifiskās zināšanas par katru translāciju ir iekļautas konkrētajā translatorā;
2. Šādu arhitektūru var bez problēmām pēc vajadzības paplašināt, iekļaujot jaunas sistēmas un jaunas datu apmaiņas.

Lai gan teorētiski datu apmaiņu skaits pieaug kvadrātiski pret sistēmu skaitu, tomēr prakse rāda, ka praktiski nepieciešamo apmaiņu skaits saglabājas pieņemamā līmenī. Megasistēmas ietvaros liels datu apmaiņu skaits ir pieciem Megasistēmas pamatreģistriem (īpaši Iedzīvotāju reģistram), tomēr tas ir pietiekoši unificēts un salīdzinoši viegli implementējams un uzturams.

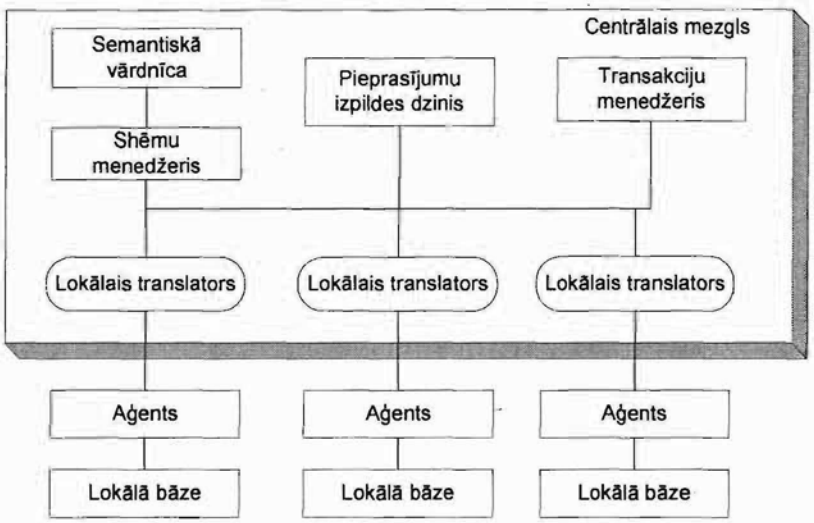


Zīm. 2 Translāciju pieeja

2.4.3.2. Pilnībā integrētā pieeja

Pilnībā integrētā pieejā visa informācija par lokālajām un globālajām semantikām tiek glabāta vienā centrālajā mezglā (datu bāzē, punktā). Pilnībā integrētās pieejas arhitektūra shematiski attēlota Zīm. 3. Galvenie moduļi šajā arhitektūrā ir modeļu menedžeris, multipieprasījumu procesors, globālais transakciju menedžeris, katrai LDBS nepieciešamais lokālais translētājs.

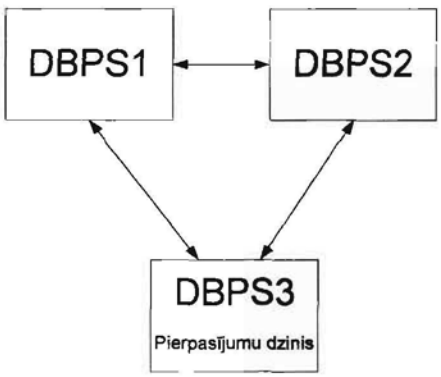
Katra LDBS, pievienojoties šai sistēmai, reģistrē savu modeli modeļu menedžerī.



Zīm. 3 Integrētā pieeja

2.4.3.3. Pilnībā decentralizētā pieeja

Kā pretstats pilnībā integrētai pieejai kalpo pilnībā decentralizētā pieeja (arhitektūra shematiski attēlota Zīm. 4), kur neeksistē centralizētais mezgls un kontrolieris, līdz ar to katra LDBPS satur multidatu bāzes pieprasījumu izpildes procesoru un komunikāciju moduli. Semantiskos konfliktus risina, formulējot pieprasījumus pieprasījumu izpildes laikā. Multidatu bāzu valoda parasti ir paplašinājums standarta valodām, piemēram, SQL un satur paplašinājumus, kas ir nepieciešami pieprasījumu izpildei no vairākām DBPS. Lai gan integrāciju parasti izpilda katrai LDBPS atsevišķi, tomēr nekas neaizliedz vairākām LDBPS veidot kopēju ontoloģiju, lai izteiktu vienādās semantikas.



Zīm. 4 Decentralizētā pieeja

2.4.3.4. Brokeru bāzētā pieeja

Brokeru bāzētā arhitektūra [DGH+95] satur konfliktu atrisināšanas moduli, kas lieto kopīgās lokālās ontoloģijas. Kad tiek iesūtīts pieprasījums, tad konfliktu atrisināšanas modulis, izmantojot lokālās ontoloģijas, ģenerē konfliktu tabulu, atrisina konfliktus un

ģenerē pieprasījumus pieprasījumu semantikā. Datu bāze, kurai pieprasa datus, šo pieprasījumu apstrādā un nodod pieprasījumu brokerim, kurš rezultātu konvertē atpakaļ uz pieprasītāja semantiku un nodod to pieprasītājam.

Pozitīvs aspekts šādai arhitektūrai (skat. Zīm. 5) ir tas, ka semantiskie konflikti kļūst caurspīdīgi lietotājam. Negatīvais aspekts ir tāds, ka šādu ontoloģiju izveide ir ļoti darbietilpīgs process, kas līdz šim nav automatizēts, pie tam, ja mainās kādas LDBPS modelis, tad šīs ontoloģijas ir jāmodificē. Līdz ar to, ja ir daudz LDBPS ar daudz sarežģītiem modeļiem, tad šī ontoloģija un translēšanas vārdnīca var būt ļoti liela. Papildus tam, atsevišķus modeļu elementus, piemēram datu integritātes ierobežojumus, var būt grūti izteikt ar šādu ontoloģiju palīdzību.

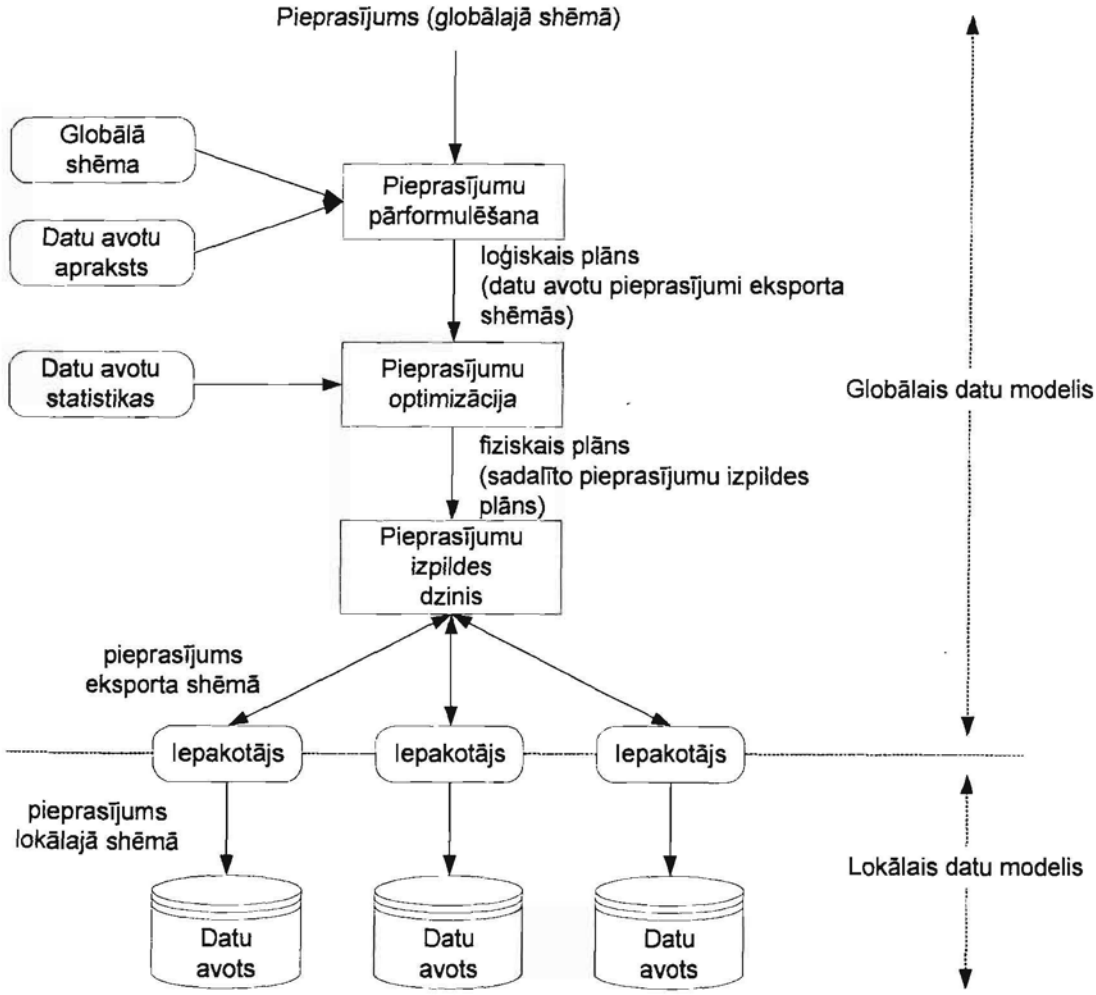
Zīm. 5 Brokeru bāzēta pieeja

2.5. Pieprasījumu apstrāde

Datu bāzu integrācijas veikšanai ir nepieciešams mehānisms, kas spētu apstrādāt kopīgā modeļa formālismā formulētu pieprasījumu. Kā redzams shematiskajā attēlā Zīm. 6, pieprasījuma apstrāde sākas ar pieprasījumu pārformulēšanu no globālā modeļa uz datu avotu eksporta modeļiem. Atkarībā no tā, kā ir realizēta atbilstība starp globālo un lokālajiem modeļiem – kā GAV vai kā LAV, – mainās pieprasījumu pārformulēšanas algoritmi.

Kad pieprasījums ir pārformulēts, notiek pieprasījumu optimizēšana, jo datu avoti var būt ar ierobežotām pieprasījumu izpildes iespējām – daži datu avoti var mācēt izpildīt SQL pieprasījumus, citi avoti var būt ar ļoti nabadzīgām pieprasījumu izpildes iespējām.

Pēc tam, kad ir notikusi optimizācija un ir sastādīts pieprasījumu izpildes plāns, pieprasījumu izpildes dzinis veic pieprasījumu izpildi. Pieprasījumi tiek nodoti iepakotājiem, kas tos pārformulē datu avota sintaksē un veic pieprasījumu konkrētam datu avotam. Pēc tam rezultāti tiek pārformulēti starpnieksistēmas sintaksē un semantikā, kā arī tiek integrēti.



Zīm. 6 Pieprasījumu apstrādes shēma

2.5.1 Pieprasījumu pārformulēšana

Pielietojot datu avotu aprakstus, pieprasījumi tiek pārformulēti no globālā modeļa jēdzieniem uz lokālo modeļu jēdzieniem (bet izmantojot globālo modeli). Šai pārformulēšanai ir jāatbilst diviem kritērijiem:

- Pārformulēšanas semantiskā pareizība: atbildes, kas tiks saņemtas no datu avotiem, būs korektas atbildes uz sākotnējo pieprasījumu;
- Minimizēt vēršanos pie datu avotiem: nav jāvēršas pie datu avotiem, kas nesatur atbildi uz pieprasījumu vai tā daļu. Papildus tam, pieprasījums ir jāpārformulē tā, lai pieprasījumi datu avotiem būtu pēc iespējas precīzāki, lai izvairītos no liekas informācijas pieprasīšanas.

Tālāk tiks apskatītas vairākas metodes, kas tiek pielietotas LAV pieejā, jo GAV pieejā pieprasījumu pārformulēšana aprobežojas ar mainīgo un formulu daļu substitūciju. LAV pieejā pieprasījumu pārformulēšana ir daudz sarežģītāka un pazīstama kā "Pieprasījumu atbildēšana, izmantojot skatus".

2.5.1.1. Valoda DATALOG

Datu pieprasījumus un skatus var izteikt kā DATALOG programmas [GUW02]. DATALOG programma sastāv no nosacījumu kopas. Katrs nosacījums ir formā

```
Q(X) :- R1(X1), ..., Rn(Xn)
```

`Q` un `R1, .., Rn` ir predikātu vārdi un `X, X1,..., Xn` ir mainīgie vai konstantes. Atomu `Q(X)` sauc par nosacījuma galvu, bet atomus `R1(X1), ..., Rn(Xn)` par nosacījuma apakšmērķiem nosacījuma ķermenī. Tiek pieņemts, ka katrs mainīgais, kas parādās nosacījuma galvā, parādās arī nosacījuma ķermenī. Tas nozīmē, ka nosacījumi ir droši, t.i. nosacījuma galvā neparādās nedefinēti mainīgie. Mainīgie iekš `X` ir ar universālo kvantoru priekšā, bet pārējie mainīgie ir ar eksistences kvantoru priekšā.

Pieprasījumi var saturēt apakšmērķus, kuru predikāti ir salīdzināšanas operācijas. Mainīgajam, kas parādās šādā apakšmērķī, ir jāparādās arī parastā apakšmērķī, lai tas būtu sasiets (*binding*).

Predikāti, kas apzīmē datu bāzē esošas relācijas, tiek saukti par EDB *(Extensional DataBase)* predikātiem, savukārt predikāti, kuru relācijas tiek konstruētas no nosacījumiem, tiek saukti par IDB *(Intensional DataBase)* predikātiem. Iepriekš definētais nosacījums `Q` ir IDB predikāts. Ja visi `Ri` ir EDB predikāti, tādā gadījumā nosacījums ir konjunktīvs pieprasījums. Konjunktīvam pieprasījumam ir šāda semantika – EDB relācijām pielieto nosacījumu, veicot mainīgo substitūcijas nosacījuma ķermenī. Ja visi apakšmērķi ir patiesi, tad tiek veikta mainīgo substitūcija nosacījuma galvā un šis kortežs tiek atgriezts kā pieprasījuma rezultāts.

2.5.1.2. Pieprasījumu atbildēšana izmantojot skatus

Neformāli šo problēmu var aprakstīt šādi: kaut kādā datu modelī dots pieprasījums Q un skati V1, ..., Vn. Nepieciešams pārformulēt sākotnējo pieprasījumu Q par Q0 tādā veidā, lai Q0 izmantotu tikai dotos skatus. Ja tas ir iespējams, tad, lai atbildētu Q, pietiek atbildēt Q0, izmantojot skatus [Lev00b].

Pārformulējot to LAV pieejas terminos, katra datu avotu entītija ir definēta kā skats no globālā modeļa. Ja ir dots pieprasījums globālajā modelī un to var pārformulēt, izmantojot definētos lokālo modeļu skatus, tad šo pārformulēto pieprasījumu var izpildīt lokālajos modeļos. Principā sākotnējais pieprasījums tiek pārformulēts vairākos, kuri katrs pieprasa datus vienam datu avotam.

Pieprasījumu atbildēšanu izmantojot skatus var pielietot arī citur. Piemēram, ja ir datu noliktava ar materializētiem skatiem uz sistēmu, tad šos skatus var izmantot, lai atbildētu uz pieprasījumu, un nav vēlreiz dati jāpieprasa no sistēmām.

Vislabāk būtu atrast pārformulējumu, kas būtu "ekvivalents" sākotnējam pieprasījumam, tomēr ne vienmēr tas ir iespējams. Integrējot sistēmas, dati tajās var būt nepilnīgi un arī sistēmu pieprasījumu izpildes iespējas var būt ierobežotas, kas noved pie sākotnējam pieprasījumam līdzīga pieprasījuma izveides. Starp daudziem līdzīgiem pieprasījumiem interesē "labākais", jeb tehniski tas tiek definēts kā "maksimāli saturošais" pieprasījums.

Formāli šie termini ir [Lev00b]:

- Pieprasījumu iekļaušana un ekvivalence. Pieprasījums Q0 ir iekļauts pieprasījumā Q, ja visām datu bāzēm D, Q0(D) ir apakškopa no Q(D). Pieprasījums Q ir ekvivalents pieprasījums Q0, ja Q0 un Q ir iekļauti viens otrā.

- Ekvivalenti pārveidojumi. Pieņemsim, ka Q ir pieprasījums un V = V1, .. , Vm ir skatu kopa. Pieprasījums Q0 ir ekvivalents pārveidojums pieprasījumam Q izmantojot V, ja:

  - Q0 izmanto tikai skatus no kopas V,
  - Q0 ir ekvivalents ar Q.

- Maksimāli iekļautais pārveidojums. Pieņemsim, ka Q ir pieprasījums un V = V1, .. , Vm ir skatu kopa pieprasījumu valodā L. Pieprasījums Q0 ir maksimāli iekļautais Q pārveidojums, izmantojot V valodā L, ja:

  - Q0 izmanto tikai skatus no kopas V,
  - Q0 ir iekļauts Q,
  - Neeksistē tāds pārveidojums Q1, kuram Q0 ir iekļauts Q1 un Q1 ir iekļauts Q, un Q1 nav ekvivalents ar Q0.

2.5.1.3. Pieprasījumu pārrakstīšanas pilnība un sarežģītība

Rakstā [Lev00b] tiek apskatīta gan algoritmu pilnība, gan to sarežģītība.

Pilnība tiek definēta šādi [Lev00b]: Dota skatu kopa V un pieprasījums Q. Vai pieprasījumu pārformulēšanas algoritms vienmēr atradīs Q pārformulēto pieprasījumu, izmantojot V, ja tāds eksistē? Atbilde ir atkarīga no izmantotās pieprasījumu valodas. Reizēm pieprasījumu valoda ir tik nabadzīga, ka tajā nav iespējams atrast pārformulēto pieprasījumu, pat ja tāds eksistē. Ja neeksistē ekvivalents pārformulētais pieprasījums, tad jāmēģina atrast maksimāli iekļauto pārveidojumu. Šajā rakstā arī ir piemērs, kurā

nepieciešams izmantot rekursīvas DATALOG definīcijas, lai izveidotu maksimāli iekļauto pieprasījumu.

Pieprasījumu pārformulēšanas sarežģītība ir atkarīga no dažādiem izmantotās valodas un modeļa ierobežojumiem un pieņēmumiem, tomēr parasti tā ir NP. Rakstā [Lev00b] ir plašāks apskats par pieprasījumu pārformulēšanas sarežģītību. Daži autori piedāvā algoritmus un modeļus ar dažādiem ierobežojumiem, kur pieprasījumu pārveidošana ir ar polinomiālu sarežģītību.

2.5.1.4. Pieprasījumu pārformulēšanas algoritmi

Ja ir dots pieprasījums Q un skatu kopa V1,...,Vn, tad, lai pārformulētu pieprasījumu Q skatu Vi jēdzienos, ir jāpārlasa visas iespējamās m vai mazāk elementu konjunkcijas, kur m ir pieprasījuma Q sastāvdaļu skaits. Tālāk būs aplūkoti daži pieprasījumu pārformulēšanas algoritmi, kas neprasa pilnu pārlasi.

***Bucket* algoritms**

Šī algoritma galvenā ideja [Lev00b] ir samazināt iespējamo pārrakstīto pieprasījumu skaitu, no sākuma apskatot katru pieprasījuma daļu atsevišķi, lai konstatētu, kuri skati ir svarīgi kurām sastāvdaļām. Ja ir dots pieprasījums Q, tad algoritms Q pārformulējumu atrod divos soļos:

1. Algoritms katrai Q sastāvdaļai izveido "buķeti" ar skatiem (t.i. datu avotus), kas ir svarīgi lai šos sastāvdaļu atbildētu;

2. Algoritms mēģina atrast pārveidojumus, kas ir konjunktīvi pieprasījumi, kuri sastāv no daļām, kur katra ir no vienas "buķetes". Katrai šādai kombinācijai algoritms pārbauda, vai pieprasījums Q satur izveidoto konjunkciju, vai arī pieprasījumā Q esošu konjunkciju var iegūt, pieliekot pārbaudāmajai kombinācijai papildus predikātus. Ja jā, tad pārveidojums tiek pievienots atbildei. Tādā veidā algoritma atbilde ir vairāku konjunktīvu pieprasījumu apvienojums.

Apgriezto nosacījumu algoritms

Galvenā ideja ir izveidot katram skatam nosacījumus, kā no skatu definīcijām izteikt globālā modeļa relācijas [Lev00b]. To var iztēloties kā GAV definīciju iegūšanu no LAV definīcijām. Patiesībā notiek nevis konkrētā pieprasījuma pārveidošana, bet gan skatu definīciju pārveidošana tā, lai varētu viegli atbildēt uz pieprasījumiem.

Katra skata katrai sastāvdaļai tiek izveidots apgrieztais nosacījums. Veicot skatu apgriezto nosacījumu izveidi, katram ārējam mainīgajam tiek pielietota speciāla funkcija, lai nepazustu ekvivalences nosacījums starp mainīgajiem.

Ir zināmi arī citi algoritmi, ka piemēram *MiniCon* algoritms, kopīgo mainīgo *bucket* algoritms (abi ir uzlaboti *Bucket* algoritmi) un *CoreCover* algoritms.

2.5.2 Pieprasījumu optimizācija

Veikt pieprasījumu optimizāciju nozīmē translēt pieprasījumu par efektīvu pieprasījumu izpildes plānu (koku), t.i. darbību secību, kas ir jāizpilda pieprasījumu izpildes dzinim, lai izpildītu pieprasījumu. Vienam pieprasījumam var būt vairāki izpildes plāni. Labāko no tiem var izvēlēties pēc dažādām stratēģijām, ņemot vērā izpildei nepieciešamos resursus (procesora laiku, pārraidāmo datu apjomu, nolasāmo datu apjomu u.c.)

Optimālā izpildes plāna izvēle integrētās sistēmās ir daudz sarežģītāks uzdevums nekā parastās datu bāzē, jo ir jāņem vērā papildus apsvērumi:

- Datu avoti ir autonomi. Pieprasījumu optimizētājam var nebūt statistikas par datiem datu avotā, vai arī tā var būt neprecīza;
- Datu avoti ir heterogēni un tiem var būt dažādas pieprasījumu apstrādes iespējas. Pieprasījumu optimizētājam ir jāņem vērā gan iespējas, gan pieprasījumu apstrādes jaudas un ātrdarbības izmaiņas sakarā ar tīkla un sistēmu noslodzi;
- Tradicionālajās datu bāzēs ir viegli izrēķināt datu pārraides laiku, jo tas notiek starp disku un atmiņu. Datu integrācijas sistēmās datu pārraides laiks ir grūti paredzams, jo var mainīties tīkla noslodze;
- No vienas puses informācija datu avotos var pārklāties un optimizētājam ir jāmāk minimizēt atkārtotas informācijas pieprasīšana no dažādiem datu avotiem, no otras puses jebkurš datu avots jebkurā brīdī var kļūt nepieejams. Pieprasījumu optimizētājam jāmāk elastīgi reaģēt šādās situācijās.

Ņemot vērā šīs problēmas, datu integrācijas aplikācijās ir problemātiski pielietot tos pašus pieprasījumu optimizēšanas algoritmus, kādi tiek pielietoti tradicionālajās DBPS. Tāpēc pētnieki piedāvā dažādus adaptīvus algoritmus [Ives02], kur paralēli notiek gan pieprasījumu izpilde, gan optimizācija.

2.5.3 Pieprasījumu izpilde

Pieprasījumu izpildes laikā pieprasījums no globālā datu modeļa sintakses tiek translēts uz datu avotu lokālo modeļu sintaksi, un pēc tam to izpilda lokālās datu bāzes pieprasījumu dzinis. Pieprasījumu translēšanu veic datu avotu specifiskie iepakotāji, pārveidojot pieprasījumu no kopējā datu modeļa uz lokālo modeli un pēc tam translējot rezultātu no datu avota formāta uz starpnieksistēmas formātu. Parasti iepakotāji ir

diezgan sarežģīti un bieži vien tie tiek veidoti, izmantojot mākslīgā intelekta metodes (piemēram, datormācīšanās).

Iepakotājus var veidot vai nu pilnībā manuāli, vai pusautomātiski, izmantojot kaut kādas esošas sagataves standartoperāciju veikšanai, vai arī tos var ģenerēt automātiski, izmantojot metadatus par datu avotiem.

Informācijas atlase no WWW lapām ir tipisks piemērs, kur var pielietot iepakotāju pusautomātisku ģenerēšanu [AK97]. Ja ir nepieciešamas atlasīt informāciju no daudzām līdzīgām WWW lapām par līdzīgu tēmu, piemēram, laika ziņas, tad ir liela varbūtība, ka šajās lapās būs teksti "Šodienas laika ziņas", "Gaisa temperatūra" utml., līdz ar to varēs precīzi atrast vietas, kur ir nepieciešamā informācija.

Bieži vien iepakotājam ir jāveic papildus operācijas, kuras nenodrošina datu avots. Piemēram, ir dots datu avots ar relāciju `Auto(Marka, Cena, Gads)`. No šī datu avota tiek paprasītas Opel markas automašīnas, kurām cena ir līdz 5000.

```
C :- C:<Auto
{<Marka "Opel"><Cena P>}>
AND LessThan(P, 5000)
```

Pieņemsim, ka datu avots nenodrošina atlasi pēc atribūta Cena. Viena no datu avota uzturamo pieprasījumu sagatavēm ir šāda:

```
C :- C:<Auto{<Marka $A >}>
```

Ievietojot interesējošā pieprasījuma nosacījumus šajā sagatavē un translējot to datu avota pieprasījumu valodā, iegūst šādu pieprasījumu:

```
SELECT *
FROM AUTO
WHERE Marka = 'Opel'
```

Jāatzīmē, ka nepieciešamais rezultāts ir šī pieprasījuma apakškopa, tāpēc iepakotājam ir jāpielieto papildus atlases kritērijs:

```
C :- C:<Auto{<AutoPrice P>}>
AND LessThan(P, 5000)
```

Rezultātā tiks iegūtas Opel markas automašīnas ar cenu zem 5000.

2.6. Datu izmaiņas sadalītās sistēmās

Datu modificēšana sadalītās sistēmās ir daudz sarežģītāka nekā datu izmaiņas parastās datu bāzēs. Lai veiktu datu izmaiņas, ir jāatrisina vairāki jautājumi – kā veikt datu bloķēšanu un kā veikt transakcijas sadalītā vidē.

2.6.1 Bloķēšanas protokoli

Eksistē vairāki bloķēšanas protokoli. Viena bloķēšanas menedžera pieeja nozīmē to, ka eksistē viens bloķēšanas menedžeris, kas izvietots kādā no sadalītās datu bāzes daļām. Ja kādai citai daļai nepieciešams nobloķēt kādu ierakstu, tad tā sūta pieprasījumu bloķēšanas menedžerim, kurš veic bloķēšanu un dod atbildi, ka resurss (ieraksts) ir nobloķēts. Ja resursu nevar nobloķēt, tad pieprasījums tiek ielikts rindā un tad, kad resursu var nobloķēt, bloķēšanas menedžeris nosūta atbildi. Pēc izmaiņu veikšanas transakcijas veicējs bloķēšanas menedžerim nosūta ziņojumu, ka resursu var atbrīvot. Ja dati tiek tikai lasīti, tad tos var lasīt no jebkuras daļas. Ja dati tiek rakstīti, tad tiek iesaistītas visas daļas, kur šie dati atrodas, un visur tiek veiktas izmaiņas. Problēma ar centralizēto bloķēšanas menedžeri ir tā, ka tas ir šaurā vieta sistēmā – ja bloķēšanas menedžeris nestrādā, tad sistēma nav lietojama.

Cita, līdzīga pieeja ir galvenās datu kopijas lietošana – tiek pasludināts, ka viena no sistēmām satur datu galveno kopiju un izmaiņas tiek veiktas tikai galvenajā kopijā. Pēc tam izmaiņas no galvenās kopijas tiek nosūtītas pārējām sistēmām.

Var izmantot arī decentralizētu datu bloķēšanas algoritmu, jeb tā saukto kvoruma protokolu [SKS02]. Ja ir kopā n sistēmas, kur atrodas dati, tad var veidot datu bloķēšanas protokolu pēc sekojošas shēmas – lai datus lasītu, tad tie ir jānobloķē lasīšanas režīmā k sistēmās, bet, lai datus rakstītu, tie ir jānobloķē r sistēmās, pie tam, ir jāizpildās šādām sakarībām:

$$k+r > n$$

$$2*r > n$$

Dažādi saliekot k un r vērtības, var iegūt dažādus svarus lasīšanas un rakstīšanas operācijām, tādā veidā atvieglojot vienu un apgrūtinot otru operāciju.

2.6.2 Transakciju izpilde

Ja ir nepieciešamība veikt transakciju uz vairākām IS, tad parasti tiek izveidots transakciju menedžeris, kas rūpējas par transakcijas veikšanu un atriti *(rollback)*. Sarežģītākais ir korekti veikt atriti. Ir 2 metodes – divfāzu apstiprināšana *(2 phases commit* jeb 2PC*)* un kompensējošās transakcijas.

Ja ir iespējams sistēmā veikt 2PC, tad transakciju monitors visām sistēmām veic labojumus un, no visām sistēmām saņemot pozitīvas atbildes, nosūta sistēmām pieprasījumu datus reāli arī saglabāt.

Biežāk ir situācija, kad nav iespējams 2PC. Šādā gadījumā, lai veiktu korektu datu apmaiņu, ir nepieciešams, lai katrai darbībai būtu iespējama kompensējošā transakcija. Tā ir transakcija, kuru veicot pēc parastās transakcijas, IS atgriežas sākuma stāvoklī.

Piemēram, ir tabula Darbinieks (Vards, Alga). Tiek veikta transakcija, kas palielina darbinieka Bērziņa algu par 10%:

```
UPDATE Darbinieks
SET Alga = Alga*1.1
WHERE Vards = 'Bērziņš'
```

Ja izrādās, ka šī transakcija ir jāatceļ, tad tiek veikta kompensējošā transakcija

```
UPDATE Darbinieks
SET Alga = Alga/1.1
WHERE Vards = 'Bērziņš'
```

Diemžēl, ne vienmēr ir iespējams izveidot korektu kompensējošo transakciju. Piemēram, rodas problēmas, ja starp parasto un kompensējošo transakciju pa vidu ir veikta vēl kāda darbība ar šiem pašiem datiem un datus vairs nevar atgriezt sākotnējā stāvoklī.

3. Izmantojamās tehnoloģijas

Datu apmaiņa salīdzinoši vislabāk ir atstrādāta un realizēta tieši no tehniskā aspekta. Parasti tehniskajā līmenī daudzas lietas ir standartizētas, līdz ar to viegli savietojamas starp dažādām sistēmām. Datu apmaiņa starp sistēmām notiek, izmantojot standartizētus protokolus. Šajā nodaļā īsumā apskatītas dažas šo protokolu priekšrocības un trūkumi.

3.1. Piekļuve datiem datu avotos

Tālāk apskatītas šādas iespējamās pieejas –

- Izstrādātāja specifiski DB piekļuves formāti *(SQL*Net, Microsoft DB_Library, ODBC, Ole DB)*,
- Starpprogrammatūra ar to izstrādātāja specifiskiem formātiem (RPC, DCOM, J2EE *Java Beans*),
- HTTP orientēti izsaukumi (SOAP, *Web Services*).

Tabula 1 Pieeju priekšrocības un trūkumi

Pieeja	Priekšrocības	Trūkumi
Izstrādātāja specifiskais DB piekļuves formāts	Vienkāršāka realizācija sistēmas servera galā (ja var realizēt identifikāciju, autorizāciju un darbību reģistrāciju). Ja klienta sistēma atbalsta izstrādātāja specifisku DB piekļuves formātu, tad vienkārša realizācija klienta galā.	Sistēmā, kura pieprasa datus, ir jāinstalē attiecīgās DB klients un ir jāmāk izmantot šo klientu. Servera sistēmas DB jānodarbojas ar lietotāju autentifikāciju, autorizāciju un darbību reģistrāciju. Tas varētu būt netriviāli, ja tiek izmantota 3-līmeņu arhitektūra, kur šos uzdevumus veic starpprogrammatūra. Var rasties problēmas ar ugunsmūri.

Pieeja	Priekšrocības	Trūkumi
Starpprogrammatūra ar to izstrādātāja specifiskiem formātiem	Bieži vien ir vieglāk nodrošināt autorizāciju, autentifikāciju un darbību reģistrāciju. Ja servera sistēma ir 3-līmeņu arhitektūrā, tad varētu būt ļoti viegla realizācija servera pusē.	Var rasties problēmas ar ugunsmūri. Piesaiste konkrētajai platformai. Sarežģītāk realizēt (var būt nepieciešama papildus kodēšana gan servera, gan klienta galā).
HTTP orientēti izsaukumi	Nav problēmu ar ugunsmūri. Var panākt labu savietojamību ar praktiski visām platformām.	Varētu būt ļoti liela kodēšana realizācijā, ja sistēmas nav orientētas uz WWW un *Web* servisu tehnoloģiju izmantošanu.

3.2. Datu apmaiņas formāti

3.2.1 XML pielietošana

Ļoti lielu popularitāti ir ieguvis XML *(eXtensible Markup Language)*. Tas ir standarts, ar kura palīdzību var aprakstīt datu apmaiņas formātus. XML standartā jebkuri dati tiek nodoti teksta formā, kas dod iespēju viegli veikt konversiju starp dažādām platformām. Dati tiek kodēti datoram viegli analizējamā formā, tajā pašā laikā šo formātu var saprast arī cilvēks. Plašu XML attīstību nodrošina arī tas, ka XML standartu atbalsta lielākie datorindustrijas giganti un ir izveidojuši rīkus XML formāta datu apstrādei visām populārākajām platformām un programmatūrai. Līdz ar to loģiski, ka šībrīža izvēle datu apmaiņas formātam starp IS vairumā gadījumu ir XML.

XML priekšrocības - pašaprakstošs, viegli modificējams formāts, saglabājot savietojamību, var aprakstīt ne tikai tabulārus, bet arī kokveida datus.

XML trūkumi – salīdzinoši jauna, vēl ne līdz galam atstrādāta tehnoloģija, liels lieki pārraidāmās papildinformācijas apjoms, atsevišķi apstrādes rīki pie lieliem apjomiem strādā neoptimāli.

Megasistēmu veidojot 1998. gadā, XML vēl nebija populārs, līdz ar to Megasistēmas sākotnējā dokumentācijā nav uzsvērta XML kā standarta pielietošanas nepieciešamība datu apmaiņā. Pašlaik XML lietošanu var minēt kā standartu datu apmaiņu veidošanai Megasistēmas ietvaros, kā arī reģistriem ieteicams publicēt iespējamo XML pieprasījumu

un atbilžu shēmu Komunikāciju serverī (KS), lai potenciālie ārējie lietotāji var veikt datu apmaiņas procedūru izveidi.

3.2.2 Apmaiņa ar failiem

Populāra ir datu apmaiņa, izmantojot specifisku failu formātu, piemēram, ar speciālu simbolu atdalīti lauki, fiksēta platuma lauki, .DBF tipa faili. Šādiem failiem ir iepriekš definēta precīza struktūra, kura ir stabila un reti kad mainās.

Datu apmaiņai ar failiem ir vairākas priekšrocības - ļoti mazs papildus pārraidāmās informācijas apjoms, plaši pielietota un labi apgūta tehnoloģija.

Datu apmaiņas ar failiem trūkumi – failu struktūru ir praktiski neiespējami saprast bez papildus paskaidrojumiem, grūtāk modificēt formātu (pielikt vai noņemt laukus), piemērota apmaiņai ar tabulāriem datiem (netabulāru datu struktūru apstrādei var būt nepieciešams veidot sarežģītu programmatūru).

4. Datu apmaiņa Megasistēmā

Šajā nodaļā tiks apskatīta datu apmaiņa Megasistēmā. Ja apskata visus valsts pārvaldei nepieciešamos datus kā vienu lielu datu kopumu, tad katra Megasistēmā iesaistītā sistēma uzkrāj daļu no kopējiem datiem, kopā veidojot sadalītu, augsti heterogēnu informācijas sistēmu ar augsti autonomām lokālajām sistēmām. Šajā sistēmā ir svarīgi visi heterogēnu informācijas sistēmu integrācijas aspekti. Šajā nodaļā aprakstītas Megasistēmai raksturīgās problēmas, kā arī doti Megasistēmas izveides laikā pielietotie risinājumi.

Pēc [Hil04] klasifikācijas eksistē vairākas sistēmu integrācijas pieejas:

- Datu bāzu replicēšana;
- Eksportēšana/ transformēšana/ ielāde. Datu bāzes saturu eksportē uz starpformātu, veic nepieciešamās transformācijas un ielādē citā datu bāzē;
- *Enterprise application integration* jeb procesu koordinācija. Tā tiek veikta ar papildus programmatūras palīdzību. Parasti tiek izmantoti ziņojumu rindu serveri un starpprogrammatūra, kas šos ziņojumus apstrādā un nogādā pareizajiem adresātiem;
- *Information logistics agents* jeb biznesa procesu sinhronizācija. Katrai sistēmai ir integrācijas aģents, kas sadarbojas ar citu sistēmu aģentiem, šādā veidā sinhronizējot savā starpā dažādus biznesa procesus;
- *Enterprise information integration* jeb pieejamības uzlabošana. Parasti tiek izmantotas datu noliktavas.

4.1. Iespējamie datu apmaiņas modeļi

Megasistēmas ietvaros ir nepieciešama un iespējama dažādu tipu apmaiņa starp reģistriem un IS atkarībā no iesaistītajām pusēm, atbildes steidzamības un datu apmaiņas scenārija. Datu apmaiņā izmantojamās pieejas un tehnoloģijas mainās atkarībā no scenārija parametriem – sinhrons vai asinhrons, datus tikai lasa vai arī raksta, pieprasījums ir individuāls vai masveida. Var identificēt sekojošus datu apmaiņas scenārijus:

- sinhrona datu lasīšana no viena reģistra
- sinhrona saistītu datu lasīšana no vairākiem reģistriem
- asinhrona datu lasīšana no viena vai vairākiem reģistriem
- sinhrona datu ierakstīšana no vienas IS citā,

- asinhrona datu ierakstīšana no vienas IS citā,
- informācijas nodošana daudziem interesentiem,
- darba plūsmas tipa datu apstrāde.

Atkarībā no scenārija ir iespējamas dažādas tehniskās realizācijas datu apmaiņai. Viena no populārākajām ir tieša sazināšanās IS ar IS. Šeit ir iespējami dažādi sazināšanās līmeņi – DB līmenī vai starpprogrammatūras līmenī, vai arī kombinēti – starpprogrammatūra ar DB.

Ja datu apmaiņā iesaistās vairākas DB, tad kļūst aktuāls jautājums par centrālā resurspunkta izveidi datu apmaiņas koordinēšanai. Megasistēmas ietvaros ir izveidots centrālais punkts datu atlasei – Komunikāciju serveris.

4.1.1 Sinhronā atlase no viena reģistra

Sinhronā datu atlase no viena vai vairākiem reģistriem tiek lietota gadījumos, kad lietotājs skata datus vai arī viena sistēma pieprasa datus no kādas citas sistēmas. Sinhrons datu pieprasījums ir aktuāls, kad lietotājs tiešsaistes režīmā gaida atbildi. Piemēram, CSDD Transportlīdzekļu reģistrā tiek reģistrēta automašīna un reģistra IS pēc īpašnieka personas koda (PK) ievades apskatās IR ziņas par šo personu. Tādā gadījumā ir nepieciešama tiešsaistes datu pieprasīšana un atbildes saņemšana.

Problēmas – risks, ka kāda no sistēmām nav pieejama, ka citas sistēmas dati ir pretrunīgi salīdzinājumā ar pieprasītājsistēmā esošajiem datiem.

4.1.2 Sinhronā atlase no vairākiem reģistriem

Sarežģītāks ir scenārijs, kad tiek pieprasīti saistīti dati no vairākām sistēmām. Pieprasījumi vairākiem reģistriem tiek lietoti gadījumos, kad lietotājs veic kādu datu apskati. Piemēram, policistam ir nepieciešama informācija no vairākām sistēmām par aizdomās turamo personu. Līdzīgi varētu būt žurnālistam, veicot žurnālistisku izmeklēšanu vai arī veidojot analītisku rakstu.

Problēmas - nepieciešama iespēja veidot pieprasījumu uz dažādām platformām, dažādiem datu modeļiem, dažādu datu kodējumu. Tāpat ir jārisina kļūdaino un novecojušo datu problēma un datu modeļu atšķirības dažādās sistēmās.

4.1.3 Asinhronā datu atlasīšana

Šajā scenārijā tiek aizsūtīts pieprasījums un pēc kāda laika tiek saņemta atbilde. Tas varētu būt aktuāli gadījumos, kad tehniski nav iespējams nodrošināt sinhronu datu atlasīšanu vai kad sinhrona datu atlasīšana nav tik aktuāla. Piemēram, privātpersonai ir tiesības uzzināt, kādi dati par viņu ir zināmi dažādās IS.

Parastais risinājums – tiek atsūtīts pieprasījums faila veidā vai arī aizpildot WWW lapā pieprasījuma formu. Atbilde tiek sniegta faila veidā, kas tiek nosūtīts klientam ar e-pastu, vai izdrukas veidā, ko klients var saņemt kādā pakalpojuma sniedzēja institūcijā, vai arī klients var lejupielādēt failu ar atbildes informāciju no kādas WWW lapas.

Problēmas līdzīgas kā sinhronai datu atlasei. Jānodrošina drošs protokols informācijas apmaiņai un parasti nav precīzi zināms reakcijas laiks.

4.1.4 Datu apmaiņa sinhroni – ierakstīšana no viena reģistra otrā

Sinhrona datu ierakstīšana ir aktuāla gadījumos, kad datus savāc viena institūcija, bet par datiem atbild cita IS. Piemēram, ziņas par dzīvesvietu tiek reģistrētas pašvaldībās, bet šīs ziņas vāc un uzglabā IR. Tad ir aktuāli pieslēgties IR laikā, kad notiek dzīvesvietas reģistrācija pašvaldību informācijas sistēmā laikā, un veikt šo reģistrāciju IR. Ja reģistrācija ir veiksmīga abās sistēmās, tad var uzskatīt, ka arī kopumā transakcija ir izdevusies.

Problēmas – jārisina iespējamā datu nesakritība dažādos reģistros, datu izmaiņas pēc laika, sadalīto transakciju korekta veikšana, tiesību sadale – vai piemēram, drīkst labot un dzēst datus, kurus pats nav ievadījis? Kā identificēt maināmos vai dzēšamos datus – sistēmas iekšienē parasti lieto iekšējos identifikatorus, kuri bieži vien nav pieejami ārējām sistēmām.

4.1.5 Datu apmaiņa asinhroni – dati no viena reģistra tiek izsniegti citiem

Šis scenārijs ir interesants gadījumos, kad datu izmaiņas vienā reģistrā ir interesantas arī citiem reģistriem. Piemēram, mainot personas reģistrētās dzīvesvietas adresi IR, šīm ziņām ir jānonāk citās ieinteresētajās sistēmas, kur glabājas informācija par personu – UR, TR, VID IS u.c.

Iespējamie varianti šādai datu apmaiņai:

- Ieinteresētais reģistrs regulāri apprasās par visām to interesējošām personām – "Vai šai personai pēdējā laikā ir mainījusies dzīvesvietas adrese?". Problēma – daudz "lieku" pieprasījumu;
- Servera reģistrs publicē "paciņu" ar izmaiņām, kas notikušas pēdējā laikā (dienā, nedēļā, mēnesī), katrs ieinteresētais reģistrs paņem visu paciņu un pats izšķiro, kuras izmaiņas tam ir nepieciešamas. Problēma – ārējais reģistrs redz informāciju, kas viņam nepienākas;
- Serverī var pierakstīties uz konkrēto objektu (personu) datiem, un pie datu maiņas notikuma servera reģistrs izsūta klientiem nepieciešamo informāciju.

Problēma – liela slodze monitoringa veikšanai, ja tas tiek darīts daudz ierakstiem un daudz klientiem;

4.1.6 Datu apmaiņa asinhroni – dati no viena reģistra tiek ierakstīti citos reģistros

Šis scenārijs ir interesants gadījumos, kad par datiem atbild viena sistēma, bet datu ievads notiek citā sistēmā un sinhronu datu apmaiņu nav iespējams nodrošināt tehniski, vai arī tas nav lietderīgi. Piemēram, dzīvesvietas dati tiek mainīti pašvaldību vienotajā IS konkrētajā pašvaldībā, taču šai pašvaldībai var nebūt tiešsaistes pieslēguma. Tādā gadījumā dati par dzīvesvietas maiņu tiek nodoti IR asinhroni. IR veic datu pārbaudi pie sevis un var gadīties, ka pēc IR datiem šāda datu maiņa nav pieļaujama.

Problēma – atšķirīgu datu kvalitātes prasību dēļ dati, kas ir apstiprināti vienā sistēmā, var netikt apstiprināti otrā sistēmā.

4.1.7 Datu apmaiņa asinhroni – darba plūsma starp vairākām sistēmām

Šāds scenārijs notiek visos gadījumos, kad ir jāveic kāda sarežģītāka pieprasījuma apstrāde. Piemēram, personas iesniegums vai kāds cits dokuments apstrādes laikā parasti nonāk pie vairākiem ierēdņiem, kuri to apstrādā. Dokumenta apstrādes ceļš arī veido darba plūsmu.

Darba plūsmas var būt divu veidu [MA00]:

- Autonomas darba plūsmu sistēmas, kas tiek izmantotas kopā ar specifiskām, biznesa vajadzības nodrošinošām sistēmām;
- Iegultās darba plūsmu sistēmas, kas ir biznesa vajadzības nodrošinošās sistēmas sastāvdaļa.

Galvenā problēma – pagaidām gan Latvijā, gan ārzemēs speciālistiem ir maz pieredzes šādu sarežģītu sadalītu sistēmu izveidē un ekspluatācijā.

4.2. Datu integritāte vairākos reģistros

Gadījumā, kad ir viena institūcija, kas ir atbildīga par datiem, bet otra institūcija datus ievada un apkopo, jānodrošina datu integritāte vairākos reģistros. Piemēram, Tiesu informatīvajā sistēmā (TIS) ievada elektroniskā formā datus par laulības izbeigšanu, bet Iedzīvotāju reģistrs ir atbildīgs par šo datu uzkrāšanu. Tādā gadījumā dati no TIS ir jāievada IR. Tā kā TIS ir jāstrādā neatkarīgi no IR, tad datus ievada TIS arī gadījumos, kad IR IS kaut kādu iemeslu dēļ nav pieejama. Kad atjaunojas TIS pieeja IR, tad jaunievadītā informācija tiek nodota IR. Pastāv iespēja, ka datus par laulības izbeigšanos nevar ievadīt IR, jo IR tiek pārkāpta iekšējā integritāte. Tam var būt vairāki iemesli:

- Dati IR ir nekorekti vai novecojuši. Piemēram, nav ievadītas ziņas par laulības notikumu, līdz ar to laulību nevar izbeigt, laulība jau ir izbeigta, laulātais ir miris u.c.;
- Operatora kļūda, ievadot datus TIS, piemēram, nekorekti norādīts kāds no laulātajiem;
- Dati TIS ir nekorekti, piemēram, tiesa ir izbeigusi laulību, jo nav zinājusi, ka laulība jau ir izbeigta.

Šajā brīdī starp abiem reģistriem rodas atšķirības, kas ir jānovērš. Eksistē vairāki iespējamie varianti, kā to darīt:

- Par datu apstrādi un nekorektību novēršanu rūpējas pati sistēma;
- Par datu nekorektību novēršanu rūpējas atbildīgais personāls.

Ja tiek uzskatīts, ka sistēma pati tiks galā, tad tas parasti izpaužas kā iebūvēta loģika, kas nosaka, kā apstrādāt šādas izņēmuma situācijas. Tomēr parasti tas nav iespējams un ar izņēmuma gadījumiem nodarbojas personāls. Lai to izdarītu, personālam ir jāpārbauda reģistru datu atbilstība reālajai dzīvei (dokumentiem) un jāveic korekcijas vienā vai otrā reģistrā.

4.3. Datu apmaiņas ieteicamais modelis

Reģistriem sadarbojoties un apmainoties ar datiem, var rasties dažādas datu integritātes problēmas starp tiem, kas nozīmē, ka dati par vienu un to pašu objektu dažādos reģistros atšķiras. Tas notiek tāpēc, ka reģistri ir ievadījuši atšķirīgus datus vai nu kļūdas rezultātā (gramatiskās vai loģiskās), vai arī ir izmantoti novecojuši dati. Lai no tā izvairītos, ir ieteicams sekojošs modelis:

- Ir viena institūcija, kas atbild par konkrēta objekta konkrētu datu pareizību (atbildīgā institūcija). Piemēram, IR atbild par personas vārda, adreses, valstiskās piederības aktuālu un pareizu reģistrāciju, Transportlīdzekļu reģistrs atbild par ziņām par transportlīdzekli, kā arī par transportlīdzekļa īpašnieku identificējošo informāciju (personas kods vai uzņēmuma reģistrācijas numurs UR);
- Ir viena vai vairākas institūcijas, kas atbild par konkrēto datu aktualizāciju (datus aktualizējošā institūcija). Parasti tā ir tā pati institūcija, kas atbild par datiem, bet reizēm ir izņēmumi. Piemēram, par personas deklarētās dzīvesvietas adresi atbild IR, bet šo informāciju aktualizē pašvaldības, IR, PMLP u.c institūcijas. Institūcija, kas datus aktualizē, tos nosūta atbildīgajai

institūcijai, un dati ir aktualizēti tikai tad, kad atbildīgā institūcija akceptē izmaiņas;

- Citas institūcijas (datus izmantojošās institūcijas), kurām ir nepieciešami šie dati, tos ņem no atbildīgās institūcijas elektroniskā veidā. Ja ir nepieciešamība, citas institūcijas var ņemt elektroniskā veidā kopijas no kopijām. Galvenais princips, kas ir jāievēro - dati netiek sakropļoti un ievadīti atkārtoti un ka visas nepieciešamās izmaiņas nonāk līdz izmantotājinstitūcijai. Piemēram, Transportlīdzekļu reģistrs sākotnējos datus un vēlākās izmaiņas par personas vārdu, uzvārdu, deklarētās dzīvesvietas adresi ņem kopiju veidā no Iedzīvotāju reģistra.

5. Komunikāciju serveris

BVVDPT izstrādes laikā izvirzījās nepieciešamība pēc vienota resurspunkta pieejai valsts datiem, kad vienas valsts dienestiem, lai iegūtu informāciju par citā valstī reģistrētiem objektiem (uzņēmumiem, personām, transportlīdzekļiem), ir lietderīgi iegūt nepieciešamās ziņas no viena informācijas avota, neiedziļinoties citas valsts datu bāzu struktūrās. Latvijā kā centrālais resurspunkts tika izveidots Komunikāciju serveris, kura izveidē autors piedalījās. KS pielietošana, kā to ir pierādījusi Megasistēmas izstrāde, ir aktuāla arī vienas valsts ietvaros kā universāls līdzeklis informācijas apmaiņai starp dažādām IS. Vēlāk līdzīgu ideoloģiju izvirzīja arī Eiropas valdību datu apmaiņas projektā IDA [IDA00].

Šīs pieejas pareizību apstiprina tas, ka līdzīgs kopējs centrālais resurspunkts tiek veidots sistēmu integrācijas projektos arī citās valstīs (Lielbritānija [UKGV], Somija [SOGV], Krievija [RUGV]).

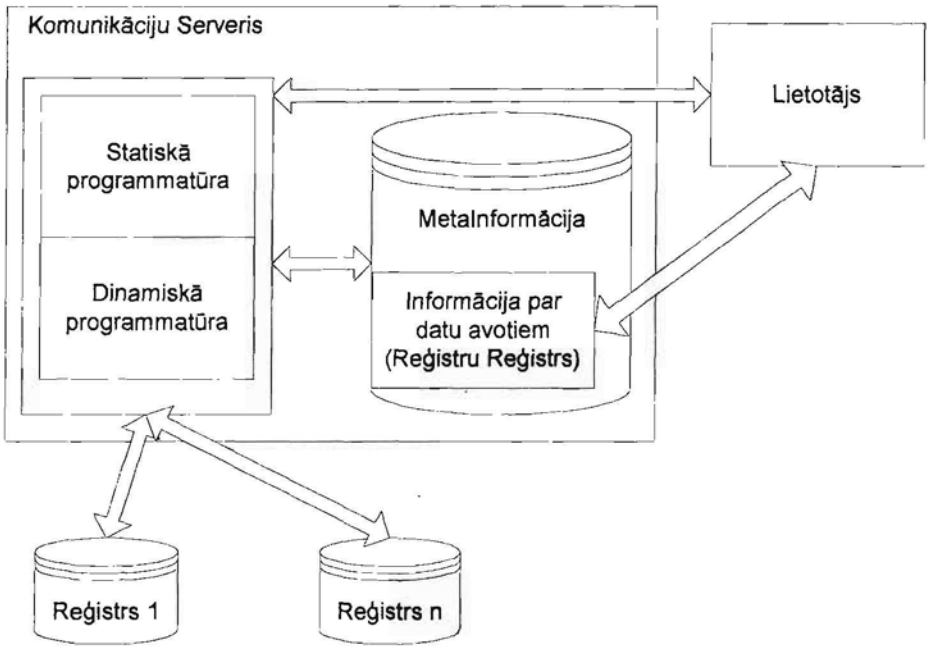
Vienota resurspunkta pieeju veiksmīgi var lietot arī vienas institūcijas ietvaros, lai veidotu vienotu datu atlasīšanas mehānismu no daudzām šīs institūcijas datu bāzēm, kā tas tika izdarīts Iedzīvotāju reģistrā.

5.1. Komunikāciju Servera struktūra

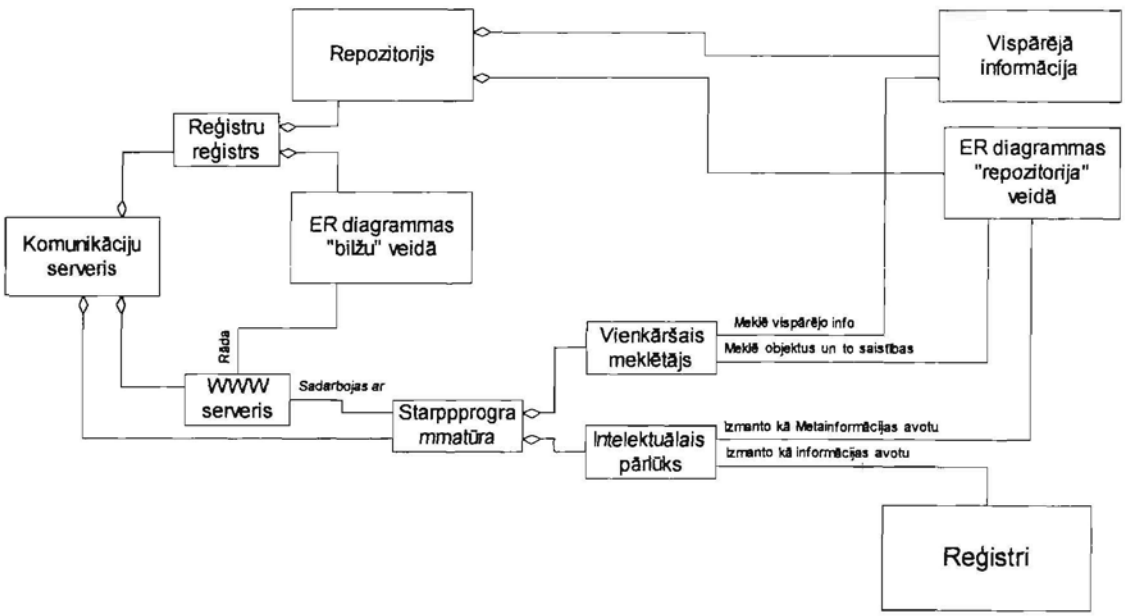
KS struktūra un galvenās sastāvdaļas – Repozitorijs (Reģistru reģistrs) un programmatūra darbam ar to, kā arī Universālais datu pārlūks ir attēlotas zīmējumos Zīm. 7 un Zīm. 8.

- Reģistru reģistrs ir reģistrs, kurā ir reģistrētas citas IS. Tas satur vispārēju informāciju par konkrēto reģistru, tajā reģistrētajiem datu objektiem, izstrādātāju un īpašnieku, tehnisko raksturojumu u.c. ziņas. Reģistru reģistram var piekļūt mājas lapā [MEGA98+];

- Universālais pārlūks ir uz metamodeli bāzēts datu pieprasīšanas un apskates rīks ar WWW saskarni. Veidojot Universālo pārlūku, tika izpētītas un izanalizētas vairākas starpnieksistēmas: TSIMMIS [HGIP95], GARLIC [HMNT99], DISCO[TRV96], diemžēl neviena no tām nenodrošināja nepieciešamo funkcionalitāti, tāpēc tika pieņemts lēmums izveidot Universālo pārlūku – sistēmu, kas nodrošinās visu nepieciešamo funkcionalitāti.



Zīm. 7 Komunikāciju servera darbības modelis



Zīm. 8 Komunikāciju servera objektu modelis

Komunikāciju serveris, balstoties uz 2.nodaļas klasifikāciju, ir tipiska starpnieksistēma. Pēc globālā modeļa tipa KS par globālo modeli izmanto semantisko modeli (paplašināts ER modelis). Pēc saites tipa starp globālajiem un lokālajiem modeļiem KS piemīt gan GAV, gan LAV īpašības. Sīkāk KS repozitorijs ir aprakstīts tālāk šajā nodaļā.

5.2. Komunikāciju servera funkcionalitāte

Komunikāciju serveris, kura darbību shematiski rāda Zīm. 7, realizēts kā interneta resurspunkts. Lietotāji piekļūst KS ar HTTP protokola palīdzību. KS lietotājiem nodrošina iespēju:

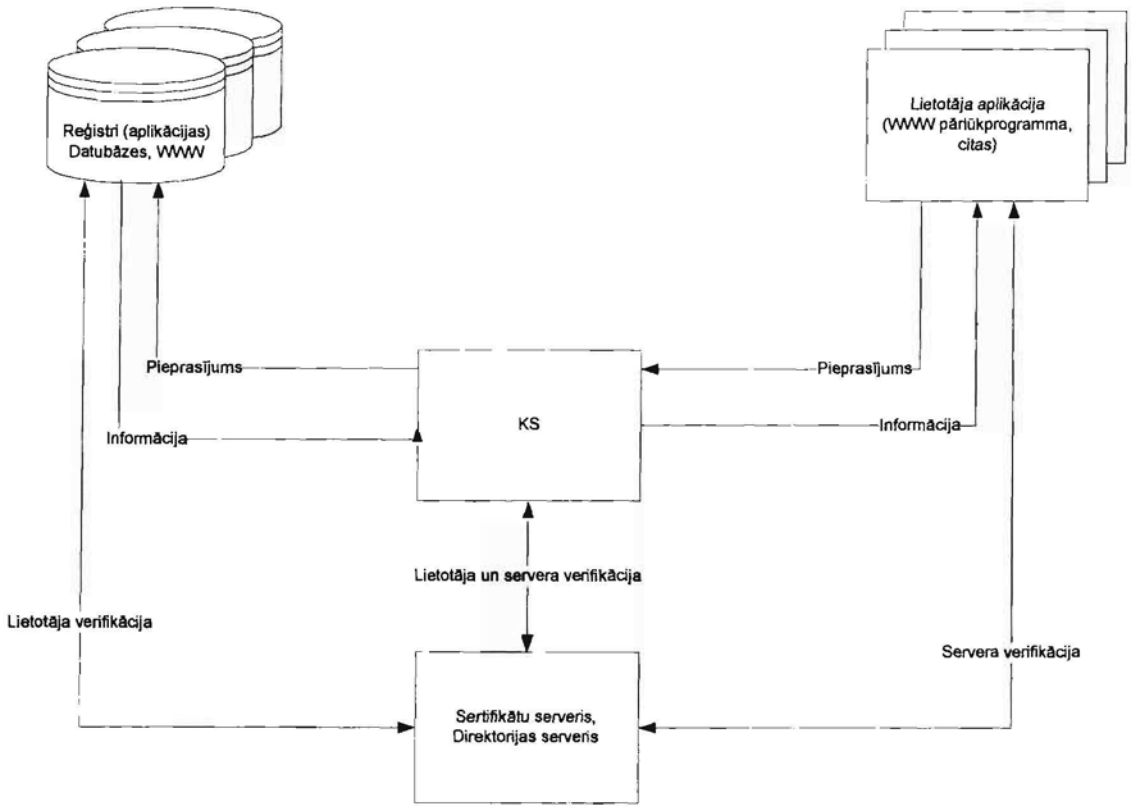
- iepazīties ar to, kur un kāda informācija glabājas (to veic Reģistru reģistrs) un, otrkārt,
- pieprasīt un saņemt informāciju no dažādiem reģistriem, neiedziļinoties to struktūrā (to nodrošina Universālais pārlūks).

Lietotājus, kuri grib piekļūt sensitīvai informācijai, pirms darba sākuma ar sistēmu identificē ar sertifikātu palīdzību. Šiem lietotājiem ir jāsaņem X.509 standartam atbilstošs sertifikāts. Sertifikāts satur sevī lietotāju identificējošu informāciju, to praktiski nevar noviltot un tiek izmantots gan datu šifrēšanai, gan lietotāja identificēšanai. KS tiek lietots standarta šifrēšanas protokols SSL.

Lietotājs piesūta KS informācijas pieprasījumus un saņem no tā atbildes. Informācijas piesūtīšana un atbildes saņemšana notiek tiešsaistes režīmā, izmantojot pārlūkprogrammu.

Lietotājs sadarbojas ar KS pēc šādas shēmas (Zīm. 9):

- Seansa sākumā tiek identificēts lietotājs. Identifikācija notiek vai nu ar sertifikāta palīdzību, vai arī lietotājam ievadot savu lietotāja vārdu un paroli;
- KS pārbauda sertifikātu vai lietotāja vārda un paroles atbilstību un nosaka lietotāja tiesības;
- Lietotājs pieprasa informāciju. KS, saskaņā ar lietotāja tiesībām, pieprasa informāciju no reģistriem un, saņēmis atbildi, sniedz to lietotājam;
- Reģistrs saņem no KS ne tikai informācijas pieprasījumu, bet arī lietotāja identifikatoru un var identificēt lietotāju un tā tiesības, kā rezultātā reģistrs sniedz informāciju KS atbilstoši konkrētā lietotāja tiesībām.



Zīm. 9 Komunikāciju servera sastāvdaļu sadarbība

KS var izdalīt šādas piecas galvenās funkcijas:

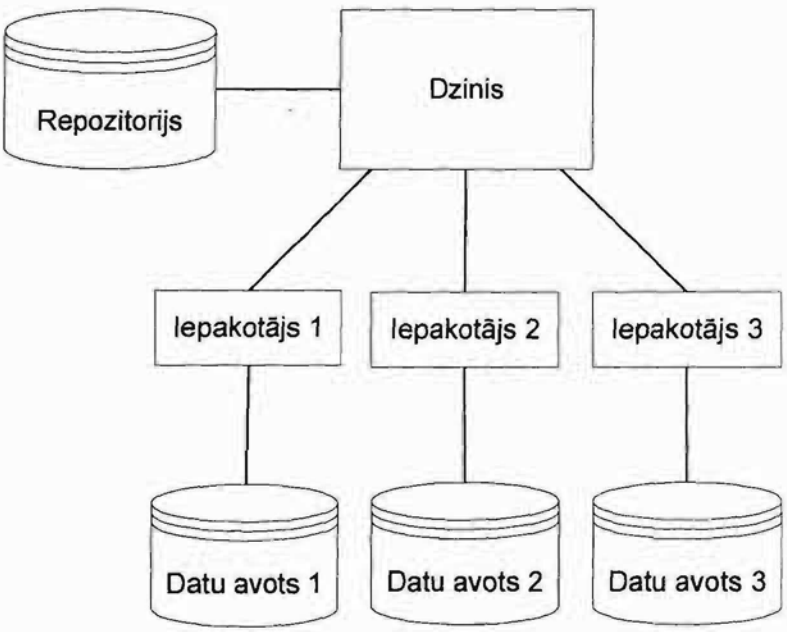
- lietotāja identifikācija un verifikācija,
- informācijas izmantošanas autorizācija,
- lietotāja tiesību vadība,
- pieprasījumu, kas prasa vēršanos pie vairākiem informācijas avotiem, izpilde,
- lietotāja pieprasījumu finanšu uzskaite.

5.3. Universālais pārlūks

Lai veiktu datu meklēšanu un attēlošanu no dažādiem reģistriem, ir izveidots UP, kurš, ņemot metainformāciju no reģistru datu modeļa, spēj strādāt ar datiem – veikt meklēšanu, parādīt saistītos datus utt. UP ir veidots kā tipiska starpnieksistēma, tāpēc UP arhitektūra atbilst starpnieksistēmas arhitektūrai.

UP galvenās daļas ir Dzinis un Repozitorijs (Zīm. 10):

- Dzinis ir programmatūra, kura ņem no repozitorija informāciju par reģistriem un to datu objektiem, informāciju par funkcijām, ar kuru palīdzību var piekļūt konkrētam reģistram, un veic meklēšanu un datu attēlošanu lietotājam;
- Repozitorijs jeb metadatu bāze ir datu bāze, kurā glabājas informācija par reģistriem, tajos glabājamajiem datu objektiem un saitēm starp tiem, kā arī informācija par funkcijām, kuras var izpildīt, lai saņemtu informāciju no konkrētā reģistra.



Zīm. 10 Universālā pārlūka sastāvdaļas

UP funkcionalitāte īsumā ir šāda - lietotājs, izmantojot interneta pārlūkprogrammu, ievada savu pieprasījumu, un UP atlasa atbilstošos datus un nosūta tos lietotājam kā HTML lapu. Lietotājs apskata šos datus un var veikt atkārtotu, neatkarīgu pieprasījumu vai pieprasīt ar atlasīto informāciju saistītus datus.

UP, izmantojot specifiskas, nelielas programmiņas - iepakotājus var piekļūt pie praktiski jebkādiem datu avotiem. Iepakotāji tiek katram datu avotam izstrādāti specifiski, līdz ar to datu avoti var būt pieejami ar visdažādākajām tehnoloģijām, piemēram, DCOM, SQL*Net, ODBC, WebServices. Iepakotāji saņem no UP pieprasījumu, konvertē to datu avotam saprotamā formā un tehnoloģijā, izpilda un pēc tam saņemto rezultātu konvertē UP saprotamā XML formātā. UP iekšpusē dati tiek apstrādāti XML formātā.

Dati par datu avotu funkcionalitāti un datu savstarpējo saistību glabājas UP metadatu bāzē. Šeit arī glabājas informācija par formatēšanu, kas ir jāveic datiem pirms rādīšanas

lietotājam. UP datu transformācijas veic ar XSL palīdzību. Metadatu bāzē esošo modeli vispirms uzzīmē ar GRADE palīdzību, pēc tam eksportē kā EIF failu un pēc tam šo eksportēto modeli ar speciāli izstrādāta rīka palīdzību importē metadatu bāzē.

Lai pievienotu jaunu datu avotu (datu bāzi), ir:

- jāizstrādā iepakotāji, kas atlasa datus no šī datu avota;
- jāpapildina esošais GRADE datu modelis ar jaunā datu avota datu modeli un pēc tam šis jaunais datu modelis jāpārnes uz metadatu bāzi;
- jāizveido XSL fails, ar kura palīdzību jaunie dati tiks transformēti no standarta izskata uz kādu citu izskatu - varbūt vizuāli labāku.

Līdz ar to, pievienojot jaunu datu avotu, nav jāveic modifikācijas paša UP biznesa loģikas slānī. Var gadīties, ka ir nepieciešamas modifikācijas datu avota pieprasījumu izpildes funkcijās, kas var būt saistītas ar papildus prasībām lietotāju identifikācijai un autorizācijai.

5.4. Repozitorijs (reģistru reģistrs)

Latvijā ir vairāk kā 100 dažādu IS un datu bāzu, kas tiek izmantotas dažādos valsts pārvaldes uzdevumos. Pieaugot informācijas avotu skaitam, ātri var nonākt pie informācijas pārbagātības, kurā neviens nespēs orientēties. Ir nepieciešams visus informācijas avotus un tajās ietverto informāciju klasificēt, ņemot vērā, ka informācijas avoti mainās. Līdz ar to ir loģiski informāciju par šīm datu bāzēm un IS apkopot kādā metadatu bāzē – reģistru reģistrā. Vienlaicīgi reģistru reģistrs satur arī repozitoriju, kas tiek veidots modeļu integrācijas procesa laikā (sk. Nodaļu 2.3.1.2, terminu *Datu vārdnīca*).

Reģistru reģistram eksistē vairāku veidu lietotāji un pastāv vairāki pielietojumi.

- Pirmkārt, tie ir parastie cilvēki, kurus interesē vispārējā informācija, kur kāda veida informācija tiek saglabāta. Lietotājam saprotamā veidā un terminos ir jāapraksta potenciāli iegūstamā informācija un veidi, kā viņš to var pieprasīt. Jābūt iespējai definēt lietotājam saprotamus skatus uz sistēmām ar lietotājam nepieciešamo informāciju un lietotājam saprotamos terminos;
- Otrkārt, tie ir sistēmu izstrādātāji un uzturētāji, kurus interesē citu sistēmu datu modelis un galvenokārt eksporta modeļi – t.i. to datu modelis, kurus konkrētā sistēma piedāvā citām sistēmām, kopā ar precīzu saskarnes specifikāciju, kā šos datus pieprasīt un saņemt. Tāpat šeit būtu jābūt aprakstītiem likumiem, kā atrisināt pastāvošās atšķirības starp datu avotiem (sk. Nodaļu 2.4 par semantiskajām heterogenitātēm starp datu avotiem)

- Trešais lietojums ir klasiskais *Datu vārdnīcas* lietojums Komunikāciju serverī. Tas nozīmē, ka informācijai jābūt formalizētai, lai to varētu viennozīmīgi izmantot komunikāciju servera programmatūra.

KS iekšienē eksistē datu avotu repozitorijs, kurā formālā veidā ir aprakstīti avoti, to īpašības, tajos ietvertie dati un to īpašības. Repozitorijam ir jābūt ļoti elastīgam, viegli un ātri maināmam, lai sekotu līdzi apkārtējās vides izmaiņām. Lai adekvāti reaģētu uz lietotāja pieprasījumiem, citām KS sastāvdaļām jāprot pašām dinamiski pieskaņoties izmaiņām šajā repozitorijā.

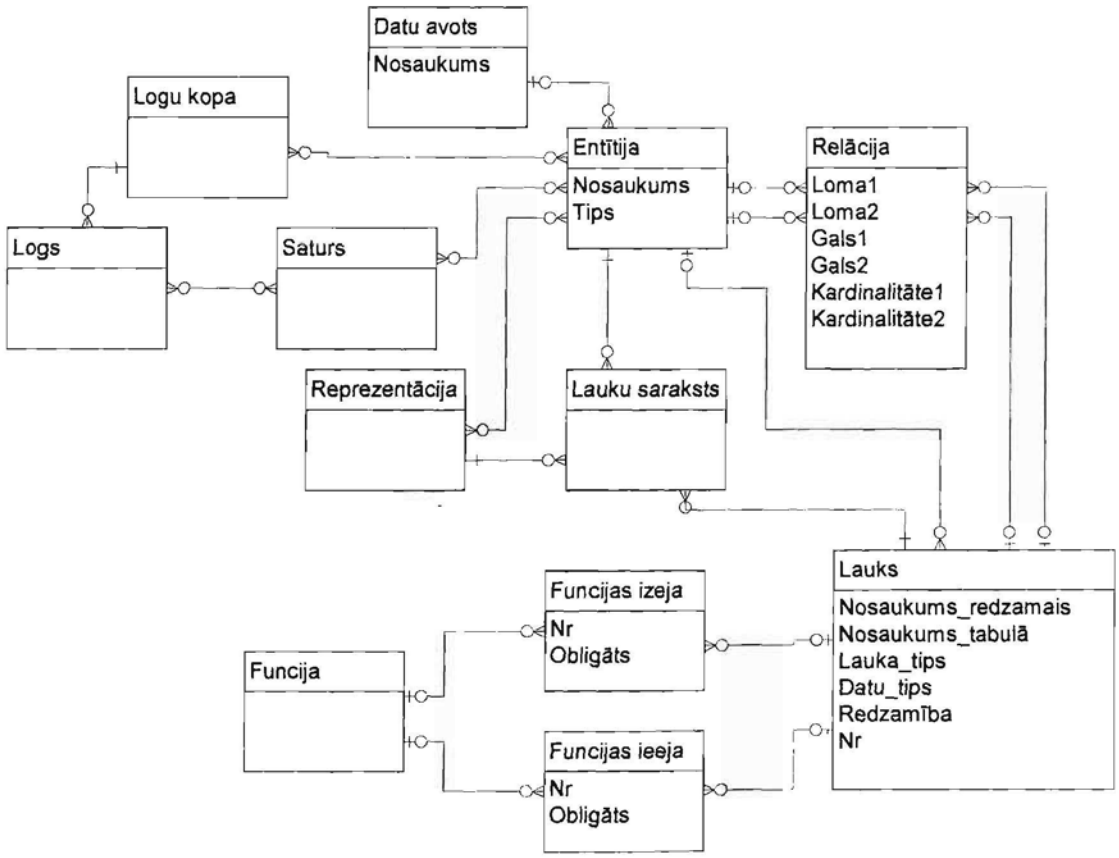
Taču šis repozitorijs nav domāts tikai KS iekšējai lietošanai. Arī lietotājam ir jāzina, kur un ko viņš var saņemt (protams, lietotāja tiesību ietvaros). Līdz ar to KS ir jāsniedz arī informācija par informāciju (metainformācija). Lietotājam saprotamā veidā un terminos ir jāapraksta potenciāli iegūstamā informācija un veidi, kā viņš to var pieprasīt. Bez tam, jācenšas daudzus pieprasījuma formulēšanas mehānismus cieši saistīt ar šo repozitoriju, tādējādi atvieglojot darbu lietotājam, kas reti lieto KS pakalpojumus.

Lietotājam bieži vien neinteresē, kur un kā vēlamā informācija glabājas. KS jāspēj apmierināt pieprasījumus, kas saistīti ar informāciju no daudziem avotiem. Līdz ar to repozitorijā ir jāapraksta arī avotu savstarpējā saistība un likumi, kā risināt dažādas nesaskaņas starp avotiem, kā veikt datu konvertēšanu utt. Repozitorijam jāsatur arī informācija par funkcijām, kuras izmantojot var piekļūt konkrētajiem datiem.

Veidojot KS, XML vēl nebija ieguvis plašu popularitāti. Pašlaik turpinot attīstīt KS, būtu lietderīgi noteikt datu apmaiņu XML formātā, kā arī datu avotus aprakstīt XML shēmu veidā, sasaistot šīs XML shēmas ar pārējo repozitoriju.

5.4.1 Repozitorija metamodelis

Zīmējumā Zīm. 11 ir attēlots repozitorija metamodelis.



Zīm. 11 Repozitorija metamodelis

Modelis sastāv no šādām daļām:

- Entītijas *Datu avots*, *Entītija*, *Relācija*, *Lauks*, *Lauku saraksts* satur datu avotu modeļus, t.i. informāciju par to entītijām un relācijām;
- Entītijas *Funkcija*, *Funkcijas ieeja*, *Funkcijas izeja* satur informāciju par funkcijām, kas atlasa informāciju no datu avotiem, par to ieejas un izejas laukiem;
- Entītijas *Reprezentācija* un *Lauku saraksts* satur informāciju par dažādiem entītiju attēlošanas veidiem un to, kādi lauki kādā secībā ir jāattēlo konkrētajā attēlošanas veidā. Piemēram, entītija *Iedzīvotājs* satur informāciju par personu. *Īsajā* attēlojumā ir jārāda lauki *PersKods, Vards, Uzvards*, bet *Garajā* attēlojumā ir jārāda lauki *PersKods, Vards, Uzvards, Adrese;*
- Entītijas *Logs*, *Logu kopa*, *Saturs* satur informāciju par vizuālo attēlojumu.

5.4.2 Datu avotu konceptuālais modelis

2. nodaļā tika aprakstīti vairāki datu bāzu integrācijas mehānismi. Viens no tiem ir datu bāzu modeļu integrācija un kopējā modeļa izveide. KS izmantota līdzīga ideja – tiek veidots kopējs datu avotu modelis ar vairākām specifiskām lietām, kas ir nepieciešamas UP darbībai:

- Meklēšanas funkcijas ar ieejas un izejas laukiem,
- Specifiski lauku atribūti (redzamība),
- Lai vienkāršotu saišu attēlošanu, ieviestas virsklases, kuras reālajā dzīvē nevienā reģistrā neeksistē.

Parasti dažādās sistēmās ir viens un tas pats objekts dažādās lomās, piemēram, persona vienā sistēmā ir *Skolotājs*, citā sistēmā - *Transportlīdzekļa īpašnieks*. Līdz ar to atšķirsies par šo personu vācamais informācijas apjoms, atribūtu nosaukumi utt. Parasti šādos gadījumos modeļu integrācijas metodoloģijas piedāvā veidot klasi *Persona*, kurai piemīt gan *Skolotāja*, gan *Transportlīdzekļa īpašnieka* visas īpašības. Tas savukārt nozīmē, ka ir nepieciešamas plašas zināšanas par integrējamo sistēmu semantiku, kā arī jāatrisina vairākas problēmas:

- Kā atpazīt un kartēt konkrētos laukus. Piemēram, ir divas entītijas *Skolotājs* un *Transportlīdzekļa īpašnieks*, kur vienai ir lauks *Adrese* un otrai ir lauks *Nosaukums*. Reālajā dzīvē ir iespējams, ka semantiski tie satur vienu un to pašu - adreses informāciju;
- Pretējs gadījums, ka lauki, kas divās entītijās saucas vienādi, satur semantiski dažādu informāciju. Piemēram, lauks *Adrese* var vienā gadījumā būt reģistrācijas adrese un otrā gadījumā kontaktadrese;
- Vēl problemātiskāks ir gadījums, ja ir entītijas *Filma* un *Transportlīdzeklis*, un abās ir lauks *Garums*. Līdz ar to lauki, kas saucas vienādi, var būt patiesībā dažādi pēc savas semantikas;
- Var būt arī problēma, ka informācija, kurai būtu jāsakrīt, kaut kādu iemeslu pēc ir atšķirīga. Piemēram, nosaukums vienam un tam pašam uzņēmumam dažādos datu avotos var būt dažāds;
- Var atšķirties mērvienības - Latvijā garuma mērvienība ir metri un kilometri, Anglijā - jardi un jūdzes.

Veidojot Universālā pārlūka metadatu bāzi, problēma tiek risināta daudz vienkāršāk. Tiek identificēts, kuru klašu objekti ir saistīti vai sakrītoši, un tiek izveidota norāde, kā, zinot

vienas klases objekta informāciju, var uzzināt otras klases objekta informāciju. Līdz ar to ir nepieciešams tikai identificēt saistītās klases un informācijas atlasei nepieciešamos atribūtus. Piemēram, ja ir entītijas Transportlīdzeklis un Transportlīdzekļa īpašnieks, kā arī funkcija, kurai kā parametru iedodot transportlīdzekļa īpašnieka Personas kodu, tiek atlasīti šīs personas transportlīdzekļi, tas nozīmē, ka pastāv saite no entītijas *Transportlīdzekļa īpašnieks* uz entītiju Transportlīdzeklis, pie tam šo saiti realizē ar laukiem *Personas Kods*, kuri ir abās šajās entītijās..

5.4.2.1. Objektu hierarhijas - virsklases

Bieži ir situācija, ka viens objekts dažādās sistēmās tiek pierakstīts ar dažādiem tipiem, piemēram, *Skolotājs* un *Transportlīdzekļa īpašnieks*. Ja integrējamo sistēmu ir daudz, tādā gadījumā parādās ļoti daudz saišu starp klasēm. Lai saišu skaitu mazinātu un līdz ar to samazinātu kopējā modeļa sarežģītību, repozitorijā ir ieviests speciāls entītijas tips - virsklase jeb bāzes klase. Šī tipa entītijas mazliet līdzinās objektorientētu programmēšanas valodu bāzes klasei vai klašu diagrammu virsklasei. Bāzes klases entītijas neeksistē nevienā datu avotā un kalpo, lai varētu vieglāk izveidot saites starp dažādu datu avotu viena tipa entītijām.

Atsevišķo sistēmu modeļos tiek identificētas klases, kuru objekti reālajā dzīvē sakrīt, piemēram, *Skolotājs* un *Transportlīdzekļa īpašnieks*. Tiek izveidota šo klašu virsklase, kas aptver visas sakrītošās klases - *Persona*. Lai virsklasi varētu izmantot meklēšanai un dažādu klašu sasaistīšanai, ir nepieciešams virsklasē ieviest meklēšanā izmantojamos atribūtus. Izmantojot to, ka šādiem saturīgiem objektiem parasti eksistē visās sistēmās vienots globāls unikālais identifikators, tad tas tiek izmantots kā meklēšanas lauks un tiek ieviests virsklasē kā atribūts (šinī gadījumā, piemēram, *PersonasKods*). Pēc tam tiek izveidotas saites starp virsklasi un apakšklasēm, norādot atribūtus, kas tiks izmantoti meklēšanā.

Virsklases var izmantot arī, lai saistītu ne tikai viena tipa entītijas, bet arī lai saistītu dažāda tipa entītijas, tādā veidā samazinot attēlojamo saišu skaitu un vienkāršojot modeli. Piemēram, Zīm. 17 attēlotajā repozitorija piemērā bāzes klase *Persona ar PK* saista kopā informāciju no entītijām *Auto*, *Auto Īpašnieks*, *Persona* un *Pase*.

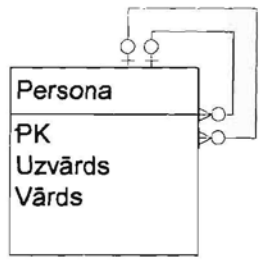
5.4.2.2. Viena entītija – n objekti (lomas)

Reizēm viena un tā pati entītija ir dažādās lomās. Piemēram, Iedzīvotāju reģistra datu bāzē persona var būt arī bērns un vecāks. Šādā situācijā varētu pastāvēt tāds fiziskais modelis kā attēlots Zīm. 12. Pastāv vairāki veidi, kā attēlot šo informāciju.

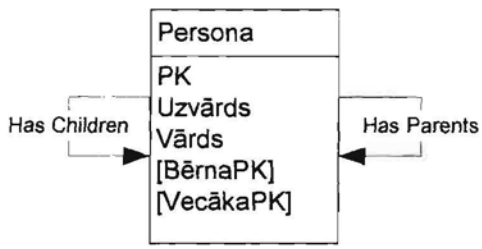
- Iespējams izveidot vienu entītiju persona, kurai pielikt papildus meklēšanas laukus *BērnaPK* un *VecākaPK*, izveidot saiti no PK uz *BērnaPK* un no PK

uz *VecākaPK*, šādā veidā implementējot saistību starp personu un tās bērniem un vecākiem (Zīm. 13);

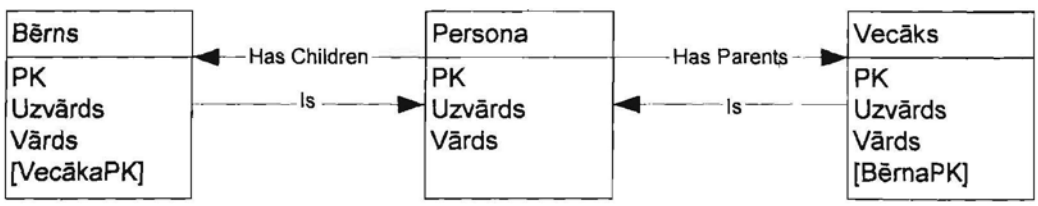
- Otrs variants ir izveidot entītijas *Bērns* un *Vecāks*, kuras ir saistītas ar entītiju *Persona*, šādā veidā ar saiti no entītijas *Persona* uz entītiju *Bērns* izsakot to, ka personai ir bērni, bet no entītijas *Bērns* uz entītiju *Persona* izsakot to, ka *Bērns* arī ir *Persona*. Līdzīgā veidā saista arī entītijas *Persona* un *Vecāks* (Zīm. 14).



Zīm. 12 Fiziskais modelis



Zīm. 13 Loģiskais modelis I

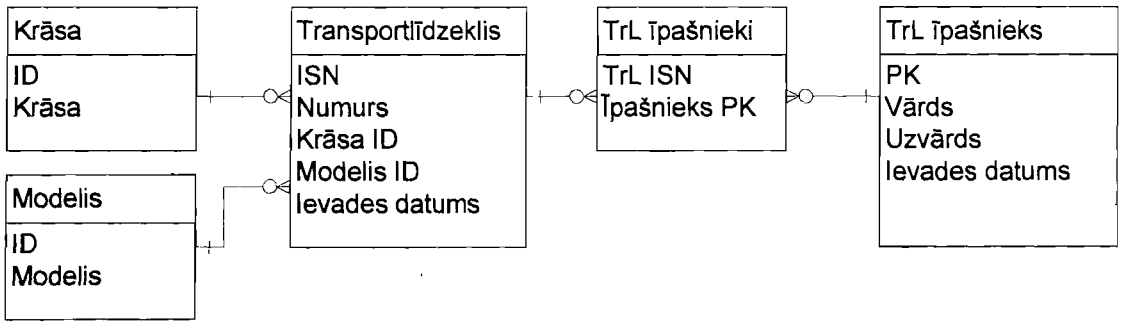


Zīm. 14 Loģiskais modelis II

5.4.2.3. N entītijas - 1 objekts

Parasti datu bāzes projektējot tiek normalizētas, tādā veidā sadalot vienas entītijas informāciju pa vairākām tabulām. Parasti tiek atdalīti atsevišķi klasifikatori, tiek atrastas *sastāv-no* tipa saites un objekts sadalīts sastāvdaļās, atdalīti semantiski neatkarīgi objekti,

darbošanās ar kuriem konkrētās sistēmas ietvaros notiek atsevišķi, bet kurus lietotājs bieži vien uztver kā vienu veselu. Piemēram, informācija par transportlīdzekļiem reālas sistēmas fiziskajā modelī var sastāvēt no vairākām tabulām (Zīm. 15). Tajā pašā laikā konceptuālo datu modeli vajadzētu veidot bez šādām tehniskām detaļām. Tas nozīmē, ka konceptuālajā modelī transportlīdzekli var attēlot kā vienu entītiju (Zīm. 17).



Zīm. 15 Transportlīdzekļu uzskaites sistēmas fiziskā modeļa fragments

5.4.2.4. Saites (meklēšanai)

Fiziskajā (relāciju) modelī starp tabulām nevar pastāvēt M:N tipa saites. Pastāv tikai 1:M tipa saites, kuras tiek realizētas kā primārās atslēgas lauki relācijā, kur kardinalitāte ir 1, un ārējās atslēgas lauki relācijā, kur kardinalitāte ir M. M:N tipa saišu problēma tiek atrisināta, ieviešot starptabulu un pārveidojot šo saiti par divām 1:M tipa saitēm. Loģiskajā datu modelī (piemēram, ER modelī vai objektu modelī) M:N tipa saites var pastāvēt.

Klasiskajā datu bāzu modelēšanā visas saites ir divvirzienu. Tas nozīmē, ka, ja ir zināms viens saites gals, iespējams atrast otru saites galu neatkarīgi no tā, kurā virzienā meklē. Relāciju modelī tas ir skaidrs, jo var jebkurai saitei izveidot atbilstošu pieprasījumu.

Veicot integrāciju un veidojot loģisko modeli, šī klasiskā lieta nav spēkā. No datu avotiem informāciju var pieprasīt, izmantojot tikai pieejamās funkcijas. Ja datu avotā eksistē funkcija, kas spēj, izmantojot par ievadu informāciju no vienas entītijas, atlasīt citas entītijas saturu, tad starp šīm entītijām eksistē saite. Piemēram, ja ir zināma informācija par personu, precīzāk, šīs personas Personas kods, tad, ja eksistē funkcija, kas pēc personas koda atlasa informāciju par personas pasi, ir iespējams pieprasīt un iegūt informāciju par personas pasi. Tas nozīmē, ka eksistē saite no entītijas *Persona* uz entītiju *Personas pase*. Tajā pašā laikā var būt situācija, ka nevar, zinot tikai informāciju par personas pasi, iegūt informāciju par personu, jo, atlasot informāciju par pasi, neiegūst nekādu informāciju par personu un neeksistē funkcija, kas pēc pases numura noskaidro pases īpašnieku.

Līdz ar to visas saites ir vienvirziena – virzienā uz relāciju, kurai ir atbilstošā datu atlases funkcija. Šīs saites ir 1:1 vai 1:M – vienam objektam, no kura iziet bulta, var būt neviens, viens vai vairāki saistīti objekti, kurus var atlasīt ar attiecīgās funkcijas palīdzību. M:N saites līdz ar to sadalās divās 1:M tipa saitēs – viena uz vienu pusi, otra uz otru pusi. Piemēram, attēlā Zīm. 15 *Transportlīdzeklis* un *TrL īpašnieks* ir saistīti ar saiti M:N (ja pieņem, ka iespējams iegūt visu transportlīdzekļa īpašnieku vēsturi). Pie tam, kāda no šīm saitēm var arī nebūt. Piemēram, var būt situācija, kad, zinot transportlīdzekļa īpašnieku, iespējams iegūt informāciju par viņa transportlīdzekļiem, bet, zinot transportlīdzekli, nav iespējams iegūt informāciju par tā īpašniekiem.

ER modeļa saites 1:M un 1:1 arī sadalās 2 saitēs – uz vienu pusi un uz otru pusi. Tāpat kā M:N tipa saites gadījumā var būt situācija, ka eksistē saite uz vienu pusi, bet neeksistē saite uz otru pusi.

5.4.2.5. Atribūtu tipi (atslēgas, parastie, tikai meklēšanas)

Datu avotu konceptuālajā modelī ir divu tipu lauki:

- Lauki, kuru saturs var tikt atlasīts ar kādas funkcijas palīdzību;
- Lauki, kuru saturu nevar atlasīt ne ar kādas funkcijas palīdzību. Tas nozīmē, ka šie lauki nav nevienas datu atlases funkcijas izejā. Parasti šie lauki tiek lietoti kā ieejas lauki kādai datu atlases funkcijai un šos laukus gala lietotājam nerāda. Šie lauki tiek lietoti, lai sasaistītu vairākas entītijas, t.i. šie lauki implementē saites starp entītijām.

Reālā datu avota datu bāzes tabulā var būt arī tehnoloģiskie lauki, kas nepieciešami sistēmas funkcionēšanai, bet nedod nekādu praktisku informāciju gala lietotājam, tāpēc šos laukus ne ar kādu funkciju neatlasa un konceptuālajā modelī neattēlo.

5.4.2.6. Meklēšanas funkcijas (uz funkcijām balstīta modeļa izveide)

Datu avots ir reāli eksistējoša sistēma, kas savus datus sniedz citām sistēmām. Kā jau tika minēts, katru datu avotu var veidot citādākā tehnoloģijā un savus datus sniegt dažādos veidos. Katram datu avotam ir funkciju kopums, kuras izmantojot dabū informāciju no datu avota. Sistēmu izstrādātājiem ir svarīgi zināt precīzu šo funkciju izsaukumu specifikāciju, savukārt gala lietotājam tas ir absolūti nesvarīgi. Gala lietotāju interesē vienkāršs un saprotams informācijas attēlojums, kas ir loģiski saistīts ar reālās pasaules objektiem, kurus šī informācija apraksta.

Autors piedāvā integrēto datu modeli veidot, par pamatu ņemot tās datu piekļuves funkcijas, kuras piedāvā konkrētie datu avoti. Piemēram, ir divi datu avoti ar šādām funkcijām:

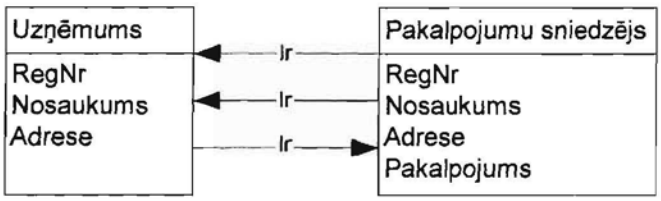
Uzņēmumu reģistrs:

```
URList(In: Nosaukums, Out: Nosaukums, RegNr)

URInfo(In: RegNr, Out: Nosaukums, RegNr, Adrese)
```

Pakalpojumu sniedzēju reģistrs:

```
PSList(In: Nosaukums, Out: RegNR, Nosaukums, Adrese,
PakalpList)
```



Zīm. 16 Meklēšanas saišu piemērs

Šādā gadījumā loģiski izveidojas divas entītijas - *Uzņēmums* un *Pakalpojumu Sniedzējs* (Zīm. 16). Pie tam, zinot pakalpojuma sniedzēja lauku *RegNr*, iespējams atlasīt uzņēmumus ar šādu *RegNr*. Arī zinot pakalpojuma sniedzēja nosaukumu, var atlasīt uzņēmumus ar šādu nosaukumu. Savukārt, ja zināms uzņēmuma nosaukums, var atlasīt pakalpojuma sniedzējus ar šādu nosaukumu. Ņemot vērā konkrēto sistēmu meklēšanas iespējas, modelī ir divas bultas no entītijas *Pakalpojumu sniedzējs* uz entītiju *Uzņēmums* un tikai viena bulta no entītijas *Uzņēmums* uz entītiju *Pakalpojumu sniedzējs*.

Funkcijas tiek glabātas UB metadatu bāzē. Katrai funkcijai ir definēti ieejas dati, t.i. kādu entītiju kādi lauki ir jādod ieejā, kā arī, vai tie ir obligāti vai nav. Katrai funkcijai ir definēti arī izejas lauki, t.i. kādas entītijas kādus laukus šī funkcija atdod kā rezultātu.

Skatoties no savas pieredzes, autors var sniegt vairākas rekomendācijas.

- Pirmkārt, vēlams, lai ieejas un izejas lauki būtu no vienas entītijas. Tas atvieglo iepakotāju un datu modeļa izveidi;
- Otrkārt, veidojot metamodeli, ir divas pieejas.
  - Viens variants ir veidot metamodeli izejot no tā, kādas funkcijas konkrētajam datu avotam ir pieejamas. Ņemot par pamatu šīs funkcijas, tiek veidots konceptuālais modelis.
  - Otrs variants - ja iespējams noteikt, kāda informācija ir nepieciešama UP lietotājam, tad, izejot no tā, būvē konceptuālo modeli un balstoties uz konceptuālo modeli, var noteikt, kādas funkcijas ir nepieciešamas, lai parādītu šo informāciju. Šādā gadījumā izvēlas divu veidu funkcijas pēc sekojoša principa:

- Dabūt identificējošo informāciju pēc kaut kādiem meklēšanas kritērijiem. Piemēram, dabūt personas PK pēc vārda un uzvārda (var būt nepilnīgi). Parasti šīs funkcijas atbilde ir saraksts ar kritērijiem atbilstošām personām.
- Dabūt vienas entītijas viena ieraksta informāciju pēc šī ieraksta identifikatora. Piemēram, pēc PK dabūt visu informāciju par personu.

Piemēram, entītijai *Persona* ir izveidotas 2 funkcijas:

```
Ieejas dati - Vārds, Uzvārds (nepilni),
izejas dati - PK, Vārds, Uzvārds
(pilni).

Ieejas dati - PK, izejas dati - PK,
Vārds, Uzvārds, Dzimšanas datums,
Adrese.
```

Vēl divas funkcijas ir izveidotas vecāku un bērnu meklēšanai:

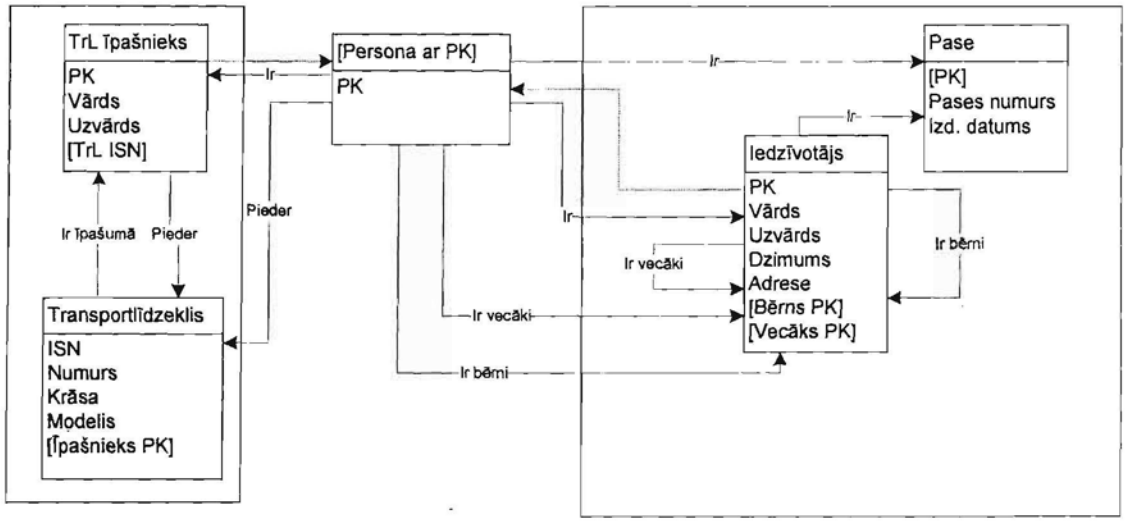
```
Ieejas dati - vecāka PK, izejas dati -
bērnu PK, Vārds, Uzvārds

Ieejas dati - bērna PK, izejas dati -
vecāku PK, Vārds, Uzvārds
```

5.4.2.7. Meklēšanas objekta - iepakotāja norāde

Lai UP varētu izpildīt meklēšanas funkcijas, tad katrā entītijā tiek norādīts iepakotājs - t.i. kāda COM komponente satur meklēšanas funkcijas, kas atlasa entītijas datus.

Attēlā Zīm. 17 ir attēloti divi datu avoti un viena bāzes klase, kas tos saista.



Zīm. 17 Repozitorija piemērs

Laukus, kas ir attēloti kvadrātiekavās, neatgriež neviena meklēšanas funkcija un tie tiek lietoti tikai meklēšanas vajadzībām (t.i. ar tiem tiek implementētas saites). Nepārtrauktās līnijas ar bultām nozīmē, ka ja ir zināma informāciju par entītiju, no kuras bulta sākas, tad var iegūt saistīto informāciju par entītiju, kur bulta beidzas. Pārtrauktā līnija parāda saiti starp *normālo entītiju* un *bāzes klases entītiju*. Bultu vērtības ir aprakstītas Tabula 2

Tabula 2 Relāciju apraksts

Gals1	**PK1**	**Gals2**	**PK2**	**Relācijas vārds**
Iedzīvotājs	PK	Iedzīvotājs	Bērns PK	Ir vecāki
Iedzīvotājs	PK	Iedzīvotājs	Vecāka PK	Ir bērni
Iedzīvotājs	PK	Pase	PK	Ir
Iedzīvotājs	PK	Persona ar PK	PK	
TrL īpašnieks	PK	Transportlīdzeklis	Īpašnieka PK	Pieder
TrL īpašnieks	PK	Persona ar PK	PK	
Transportlīdzeklis	ISN	TrL īpašnieks	TrL ISN	Ir īpašumā
Persona Ar PK	PK	TrL īpašnieks	PK	Ir

Gals1	PK1	Gals2	PK2	Relācijas vārds
Persona Ar PK	PK	Transportlīdzeklis	PK	Pieder
Persona Ar PK	PK	Pase	PK	Ir
Persona Ar PK	PK	Iedzīvotājs	PK	Ir
Persona Ar PK	PK	Iedzīvotājs	Bērna PK	Ir Vecāki
Persona Ar PK	PK	Iedzīvotājs	Vecāka PK	Ir Bērni

5.4.2.8. GRADE modelis

Datu avotu konceptuālo modeli veido ar rīka GRADE palīdzību. Pēc tam šo modeli eksportē kā EIF failu un šo failu ar speciālas programmas palīdzību importē UP repozitorijā.

Datu avotu konceptuālais modelis tiek veidots kā modificēta UML klašu diagramma. Entītijas tiek attēlotas kā klases, atribūti kā atribūti, piekļuves funkcijas kā metodes un savstarpējās saites tiek attēlotas kā relācijas.

Zīm. 18 skatāmajā piemērā var redzēt, kādas papildus lietas ir ieviestas modelī, lai varētu aprakstīt šo modeli. Klasei ir pielikti komentāri, kuros pierakstīti entītijas papildus atribūti, tajā skaitā XSL failu vārdi, kas tiks izmantoti attēlošanai nepieciešamo transformāciju veikšanai.

Katram atribūtam ir papildus atribūti - *Redzams* (vai atribūts ir vai nav redzams), *Lauka tips* (tikai meklēšanai vai arī tiek kādā rezultātā atgriezts), *NosaukumsRedz* - Lauka nosaukums, attēlojot informāciju, *IsaisSkats*, *GaraisSkats* – lauka kārtas numurs, attēlojot pamatinformāciju un izvērsto informāciju par objektu.

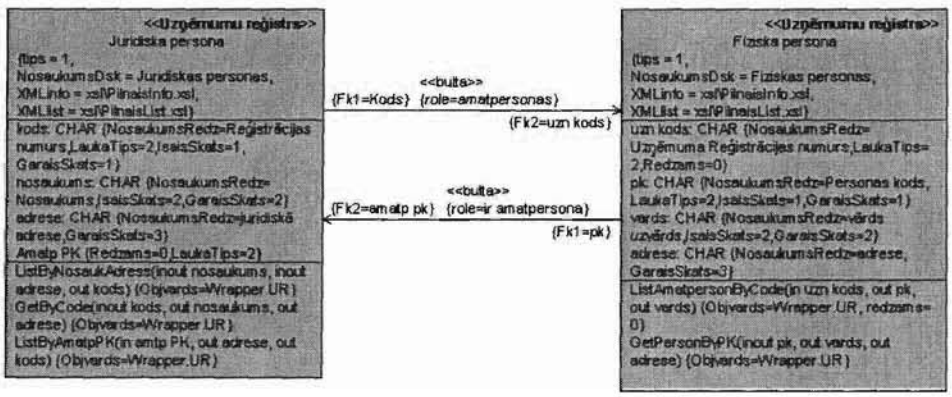
Meklēšanas funkcijām ir parametru saraksts. Katram parametram ir klāt pazīme, vai tas ir

- ieejas parametrs (*in*),
- izejas parametrs (*out*)
- gan ieejas, gan izejas parametrs (*inout*).

Katrai meklēšanas funkcijai ir COM komponentes vārds, kura satur konkrēto meklēšanas funkciju. Tas nozīmē, ka ir jābūt COM komponentei ar tādu vārdu, kāds ir norādīts

meklēšanas funkcijā pie parametra *Objvards*. Parasti komponentes vārds ir formātā "FailaVārds.ObjektaVārds". Šai komponentei ir jāsatur funkcija, kuras nosaukums sakrīt ar pieejas funkcijas nosaukumu, ieejošie parametri sakrīt ar *in* vai *inout* parametriem, kā arī atbildes XML jāsatur tagi, kuru nosaukumi atbilst out un inout parametru nosaukumiem.

Relācijas satur relācijas nosaukumu (atribūts *role*), kā arī, kurš atribūts entītijā, no kuras iziet meklēšanas saite (*Fk1*), ar kādu atribūtu ir saistīts entītijā, kurā ieiet meklēšanas saite (*Fk2*). Meklēšanas saitēm ir ierobežojums, ka var meklēt tikai pēc viena saistītā atribūta (t.i. saistība var būt starp vienu atribūtu no vienas entītijas un vienu atribūtu no otras entītijas). Šis ierobežojums līdz šim nav radījis problēmas, veidojot UP modeli.



Zīm. 18 Rīkā GRADE zīmēts repozitorija modeļa piemērs

5.5. Piekļuve citiem reģistriem

Lai KS piekļūtu citiem reģistriem, var lietot visdažādākos fiziskos piekļuves mehānismus. Katram reģistram tiek izveidots iepakotājs, kas noteiktā formātā no KS saņem pieprasījumus, pārveido tos attiecīgajam reģistram saprotamā formā un nosūta reģistram. Pēc tam reģistrs izpilda šos pieprasījumus, un atbildi iepakotājs pārveido KS saprotamā formā.

Informācijas avotu atšķirības problemātikas nopietnības ilustrācijai - daži aspekti:

- Par datu avotu jeb resurspunktu var kalpot jebkuras organizācijas jebkura informatīvā sistēma vai datu bāze. KS pakalpojumu tīklā iekļaujamās organizācijas, to IS un datu bāzes nosaka administratīvi;
- Datu avotu skaits aug. Bez tam, KS būs jānodrošina pieeja pie ārzemju informācijas avotiem, kā arī pie Latvijas dažādu organizāciju datu bāzēm, kuras arī varētu iekļauties KS pakalpojumu klāstā;

- Datu avots nav KS sastāvdaļa. Katrs datu avots primāri ir izstrādāts, lai nodrošinātu konkrētas specifiskas funkcijas, kas jāveic savā organizācijā. Datu bāzes, kas tiek izmantotas organizācijā, ir izvēlētas, izstrādātas un optimizētas tieši šīs konkrētās organizācijas vajadzībām. Tās var nebūt orientētas uz iespēju kādam citam saņemt datus, bet, ja tāda iespēja ir paredzēta, tad dati var būt ļoti specifiski un ar daudziem ierobežojumiem, kas nozīmē, ka KS ir jāpielāgojas datu avotiem, nevis otrādi. Protams, ka atsevišķi datu avoti var pilnveidot savas IS un pielāgoties KS prasībām, lai optimizētu datu apmaiņu;

- Datu avoti, kas iekļaujas KS tīklā, var atšķirties pēc savas nozīmības un lieluma. Jo nozīmīgāka būs datu bāze, jo kvalitatīvākai ir jābūt sadarbībai ar to. Arī datu bāzu lielums jāņem vērā, jo no tā ir atkarīga datu apstrādes mehānismu izvēle;

- Būtiska ir arī datu avotu kvalitāte un stabilitāte. IS var būt izstrādātas ar visdažādākajām tehnoloģijām un dažādos laikos. Atkarībā no ieguldītajiem resursiem, tās var būt vairāk vai mazāk kvalitatīvas. Protams, veikt pieprasījumus un saņemt atbildes ar modernām tehnoloģijām izstrādātu augstas kvalitātes IS un datu bāzi būs vienkāršāk un stabilāk, nekā ar vecu vai mazāk kvalitatīvu sistēmu. Tātad KS jābūt gatavam uz to, ka informācijas avoti var būt nestabili, kļūdaini strādājoši vai reizēm pat vispār nebūt pieejami;

- IS izstrādei varētu būt izmantotas dažādas izstrādes vides, dažādas datu bāzes un to darbināšanai izmantotas dažādas operāciju sistēmas un datortehnika. KS jāspēj tehniski tikt galā ar šīm problēmām, kas šobrīd gan vairs nav tā smagākā problēma, jo ir daudz dažādu risinājumu;

- Dati avotos var glabāties visdažādākajos formātos. Populārākais datu glabāšanas veids vēl arvien ir relāciju datu bāzes, strauji savu ietekmi paplašina XML datu bāzes, statiskās WEB lapas, dinamiskās WEB lapas, kas ģenerējas no kāda iekšēja formāta. Nedrīkst aizmirst arī citus informācijas glabāšanas veidus, piemēram, faili ar visdažādāko struktūru;

- Konkrēta informācijas vienība vai dati (loģiska informācijas vienību grupa) var dublēties, glabāties dažādos formātos, var būt dažādi kodēta, daļa no informācijas var tikt fiksēta kā noklusētā vērtība. Informācija var būt pretrunīga gan vienas IS ietvaros, gan starp dažādiem avotiem. Tas nozīmē, ka KS perspektīvā ir jāfiksē dažādi likumi un datu apstrādes algoritmi, kas pamatos balstās uz mākslīgā intelekta tehnoloģijām

- Nedrīkst aizmirst arī, ka katrs informācijas avots dzīvo savu samērā neatkarīgu dzīvi. Tas pilnveidojas, mainās, var likvidēties, var dzimt no jauna, var uz laiku pārtraukt savu darbību vai sadarbību ar KS. Tas nozīmē, ka KS jāeksistē ne tikai dažādā, bet arī ļoti mainīgā apkārtējā vidē.

Lai augstākminēto aspektu radītās problēmas varētu pārvarēt, UP sadarbojas ar datu avotiem, izmantojot iepakotājus. Šādam sadarbības veidam ir trīs priekšrocības:

- Iespējams piekļūt datu avotam, fiziskajā līmenī izmantojot dažādus protokolus un pieejas veidus: ODBC, OLE DB, SQL*Net, HTTP;
- Datu avotus parasti veido kaut kādu uzdevumu veikšanai, līdz ar ko tie nav piemēroti specifiskajiem UP uzdevumiem. Parasti arī piekļuve datu avotiem ir ierobežota, t.i. parasti var izpildīt noteiktas funkcijas lai piekļūtu datiem. Iepakotājs atļauj pāriet uz pieprasījumiem, kas ir veidoti saskaņā ar pieejamajām funkcijām;
- Funkciju izmantošana pieejai datu avotam atļauj viegli pāriet no datu avota fiziskā datu modeļa uz loģisko modeli, kas ir labāk saprotams ierindas lietotājam.

5.5.1.1. Pieprasījumu izpilde

UP izsauc iepakotāja funkciju, nododot meklēšanas kritērijus. Ja nepieciešams, tad iepakotāja funkcija meklēšanas kritērijus translē datu avotam saprotamā formātā. Piemēram, personas kods var būt ar vai bez svītriņas pa vidu. Ja UP ir pieņemts, ka personas kodam ir jābūt ar strīpiņu pa vidu, bet datu avotam ir nepieciešams nodot personas kodu bez strīpiņas, tad iepakotāja funkcijai ir jāveic šīs strīpiņas izgriešana.

Kad dati ir sagatavoti datu avotam saprotamā formātā, tad iepakotāja funkcija izsauc datu avota funkciju un saņem rezultātus. Pēc tam iepakotāja funkcija, ja nepieciešams, pārveido rezultātu UP saprotamā formātā (XML) un atgriež UP. Ja ir notikusi kļūda, tad iepakotājs atgriež UP kļūdas ziņojumu XML formātā.

Pieprasījuma funkcijas rezultātu UP atgriež šādā XML formātā:

```
<REZULTATS>
    <KLUDA>
        <KLUDAS_KODS/>
        <KLUDAS_APRAKSTS/>
        <KLUDAS_LIMENIS/>
```

```
        <KLUDA/>
        <RINDA isn=1>
            <LAUKS_1/>
            <LAUKS_2/>
            ...
        <RINDA/>
        <RINDA isn=2>
            <LAUKS_1/>
            ...
        <RINDA/>
        ...
    <REZULTATS/>
```

Tagā <KLUDAS> tiek aprakstītas radušās kļūdas. Ja ir rezultāts, tas tiek aprakstīts tagos <RINDA>. Katrai rindai ir savs unikāls rindas numurs. Tālāk seko tagi <LAUKS>, kas satur katras rindas visus laukus. UP tagu <LAUKS> neanalizē dziļāk, līdz ar to šajā tagā var ielikt sarežģītāku struktūru. To var izmantot pie datu attēlošanas, ja ir vēlēšanās izvairīties no metamodeļa sarežģīšanas, bet ir nepieciešams no datu avota saņemt detalizētāku informāciju. Piemēram, lauku *Adrese* metamodelī var attēlot kā vienu lauku, bet iepakotājs UP nodod XML, kur tags <ADRESE> ir šāds:

```
<ADRESE>
  <VALSTS>Latvija<VALSTS/>
  <PILSĒTA>Rīga<PILSĒTA/>
  <IELA>Raiņa 29<IELA/>
  <DZĪVOKLIS>413<DZĪVOKLIS/>
<ADRESE/>
```

Tas dod iespēju arī atkarībā no adreses valsts (Latvijas, ārzemju), mainīt lauku daudzumu un nosaukumus, nesarežģījot metamodeli.

5.5.1.2. Attēlošana

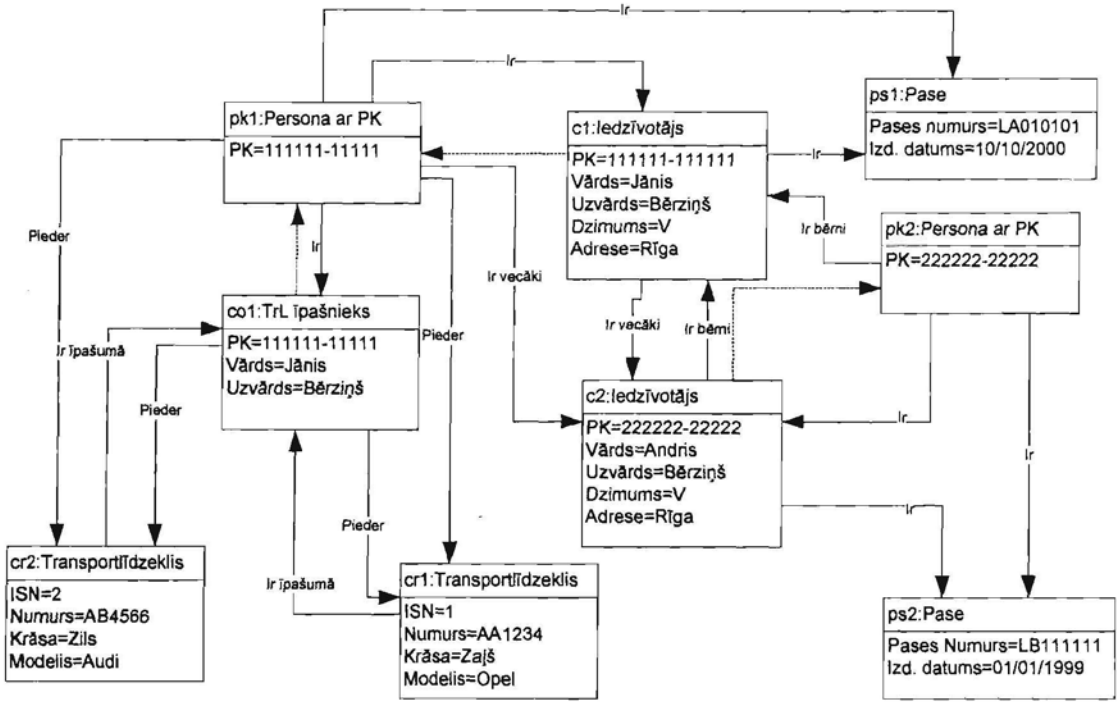
UP datus attēlot var dažādos veidos, atkarībā no lietotāja vajadzībām. UP Versijā, kas tika izveidota komunikāciju serverim, tika paredzēti dažādi attēlojumi, kas atšķiras ar attēlojamo datu daudzumu. Katrai entītijai var piesaistīt vairākus attēlojumus. Katrai entītijai eksistē īsais attēlojums, ko izmanto, ja entītija tiek rādīta kā saistīta ar kādu citu entītiju. Ja entītija ir izvēlēta par galveno entītiju, tad izmanto pilno attēlojumu. Īsajā attēlojumā tiek rādīta entītiju identificējošā informācija, piemēram, personai tas parasti ir personas kods, vārds un uzvārds. Pilnajā attēlojumā tiek rādīta pilna informācija par šo entītiju.

Attēlojuma veidošanā izmanto XSL transformāciju. Metamodelī tiek definēti XSL faili, kas nepieciešami transformācijām, lai iegūtu HTML lapas īsajam un pilnajam attēlojumam. Izpildot pieprasījumu, UP izpilda XSL transformāciju XML datiem, šādā veidā ģenerējot atbilstošu HTML lapu.

5.5.1.3. Navigācijas principi

Šajā sadaļā aprakstīti vispārīgie UP navigācijas principi, apskatot gadījumu ar vairākiem datu avotiem, kuru konceptuālais datu modelis attēlots Zīm. 17. Tam atbilstoša instanču diagramma ir attēlota Zīm. 19. Objektu saites attēlotas ar bultām - ja no pirmā objekta uz otro iet bulta, tad tas nozīmē, ka zinot informāciju par pirmo objektu, var uzzināt informāciju par otro objektu. Tas nozīmē, ka, ja ir informācija par kādu objektu, tad iespējams uzzināt informāciju par visiem tiem objektiem, uz kuriem var nokļūt, "ejot" pa bultām.

Datu aplūkošanu var iztēloties šādi - kādu no objektiem (grafa virsotnēm) izvēlas par galveno objektu un tādā gadījumā līdzi "velkas" visi ar šo objektu saistītie objekti - t.i. var uzzināt informāciju par šiem saistītajiem objektiem. Lietotājs izvēlas, pa kuru no bultām iet, un pēc noklikšķināšanas uz attiecīgās saites par aktīvo objektu kļūst objekts, kas atrodas otrā saites galā.



Zīm. 19 Instanču diagrammas piemērs

Piemēram, lietotājs izvēlas par galveno objektu c2 - personu Andri Bērziņu. Šādā gadījumā lietotājam ir iespēja veikt navigāciju pa bultu *Ir Bērni*, kas iet uz c1 un Ir, kas iet uz ps2. Pieņemsim, ka lietotājs iet pa bultu *Ir Bērni* un par aktīvo objektu izvēlas c1 – personu Jāni Bērziņu. Tādā gadījumā lietotājam ir iespējams pārvietoties no objekta c1 pa šādām bultām:

```
Has Parents -> c2,

Has -> ps1.
```

Kā arī caur pk1 var pārvietoties pa šādām bultām:

```
Is -> co1,

Owns -> cr1,

Owns ->cr2.
```

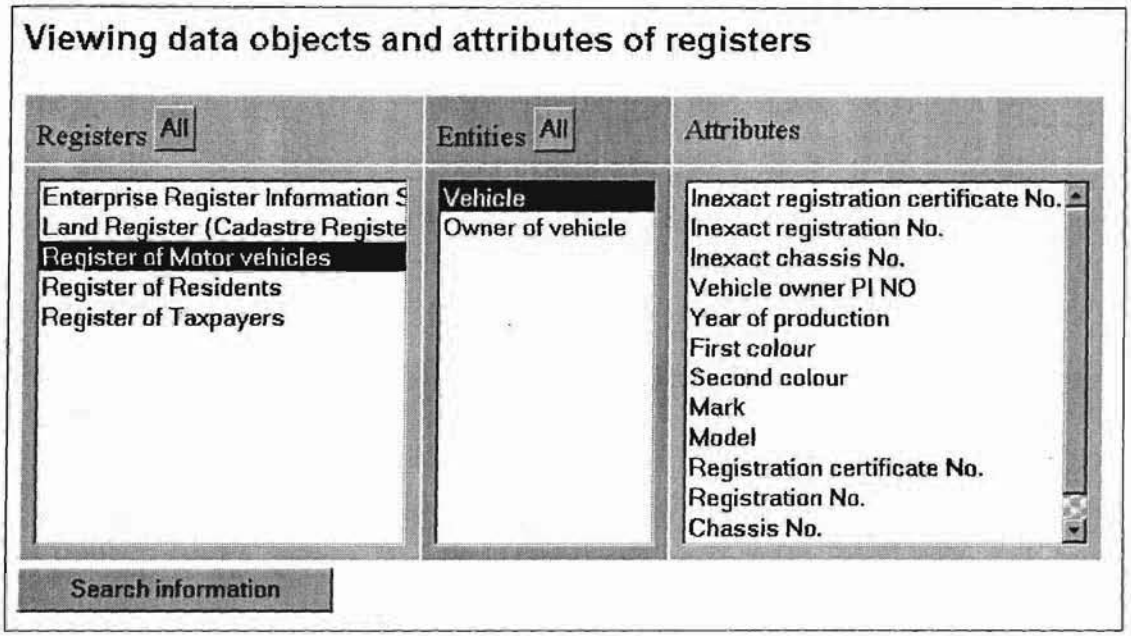
5.5.1.4. Darbs ar universālo pārlūku

Eksistē vairākas UP versijas: viena KS un otra speciāli IR. Šīs versijas atšķiras ar savu lietotāja saskarni, kā arī ar darbību.

Strādājot ar UP, kas darbojas KS, lietotājs no sākuma izvēlas, kāda tipa objektu meklēs (Zīm. 20). Katram objektam ir redzams,

- kurā reģistrā tas atrodas,
- kādi dati par šo objektu ir zināmi.

Ja izvēlas kādu reģistru, tad var redzēt, kāda tipa objekti ir pieejami konkrētajā reģistrā.

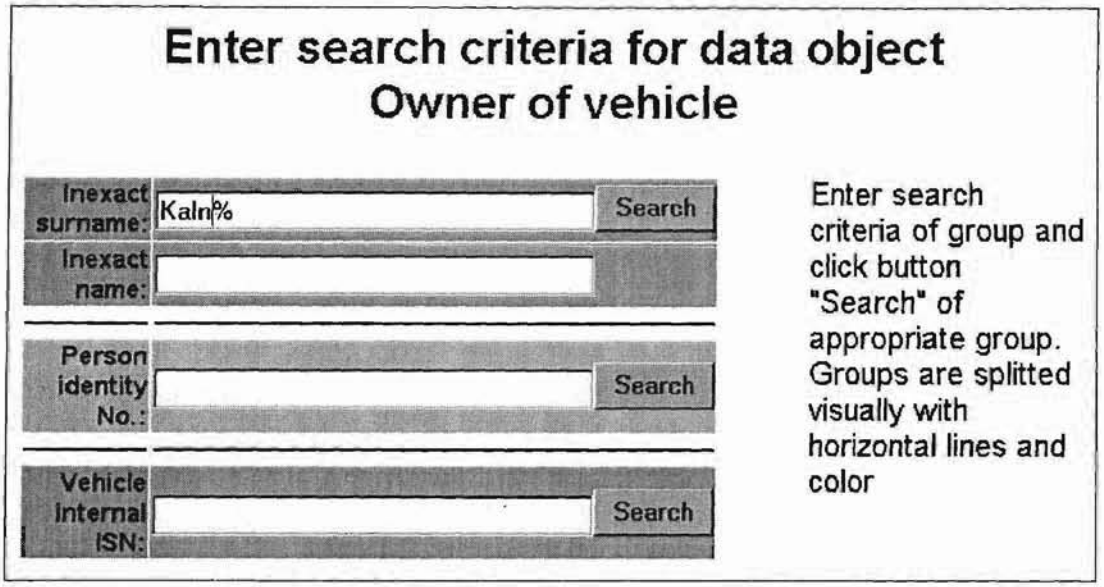


Zīm. 20 Meklējamā objekta izvēle

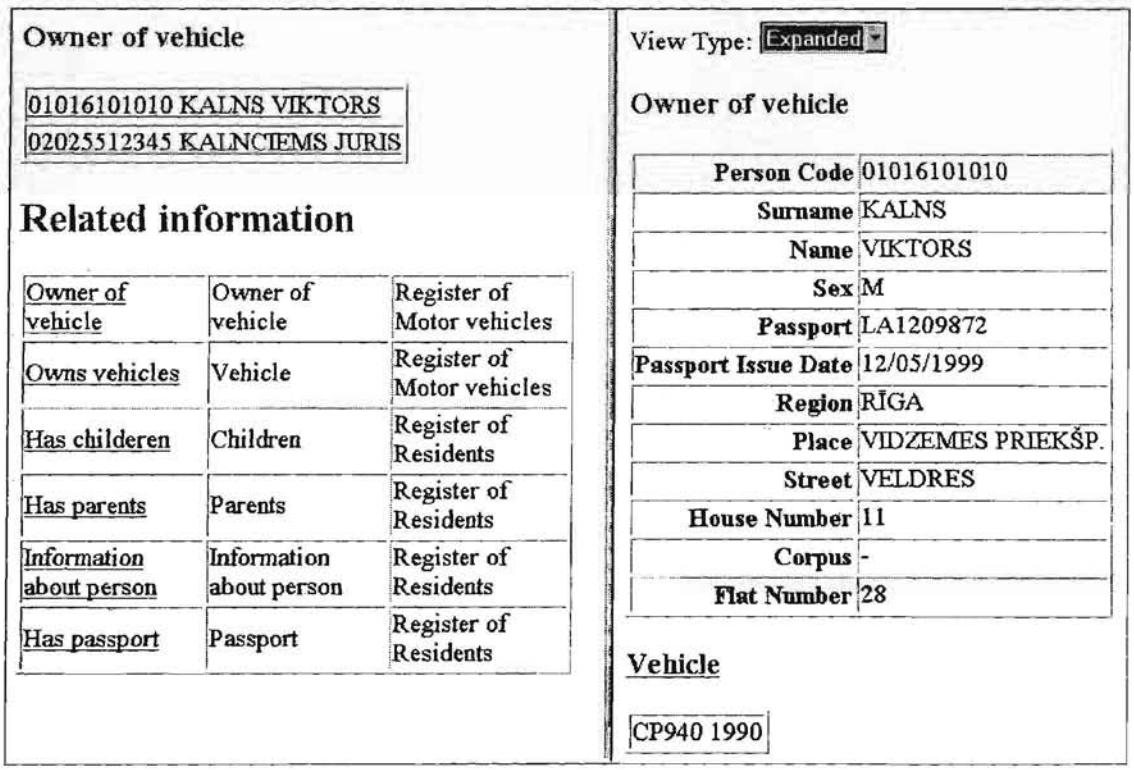
Pēc objekta izvēles parādās iespējamie pieprasījumi ar iespēju ievadīt meklēšanas kritērijus (Zīm. 21). Jāaizpilda meklēšanas kritēriji un jānospiež meklēšanas poga. Piemērā attēlota meklēšanas kritēriju ievadīšana transportlīdzekļu īpašnieku atlasei. Redzams, ka transportlīdzekļu atlasei ir pieejamas divas meklēšanas funkcijas, katra ar saviem meklēšanas kritērijiem.

UP apstrādā pieprasījumu un parāda rezultātu, kurš skatāms Zīm. 22:

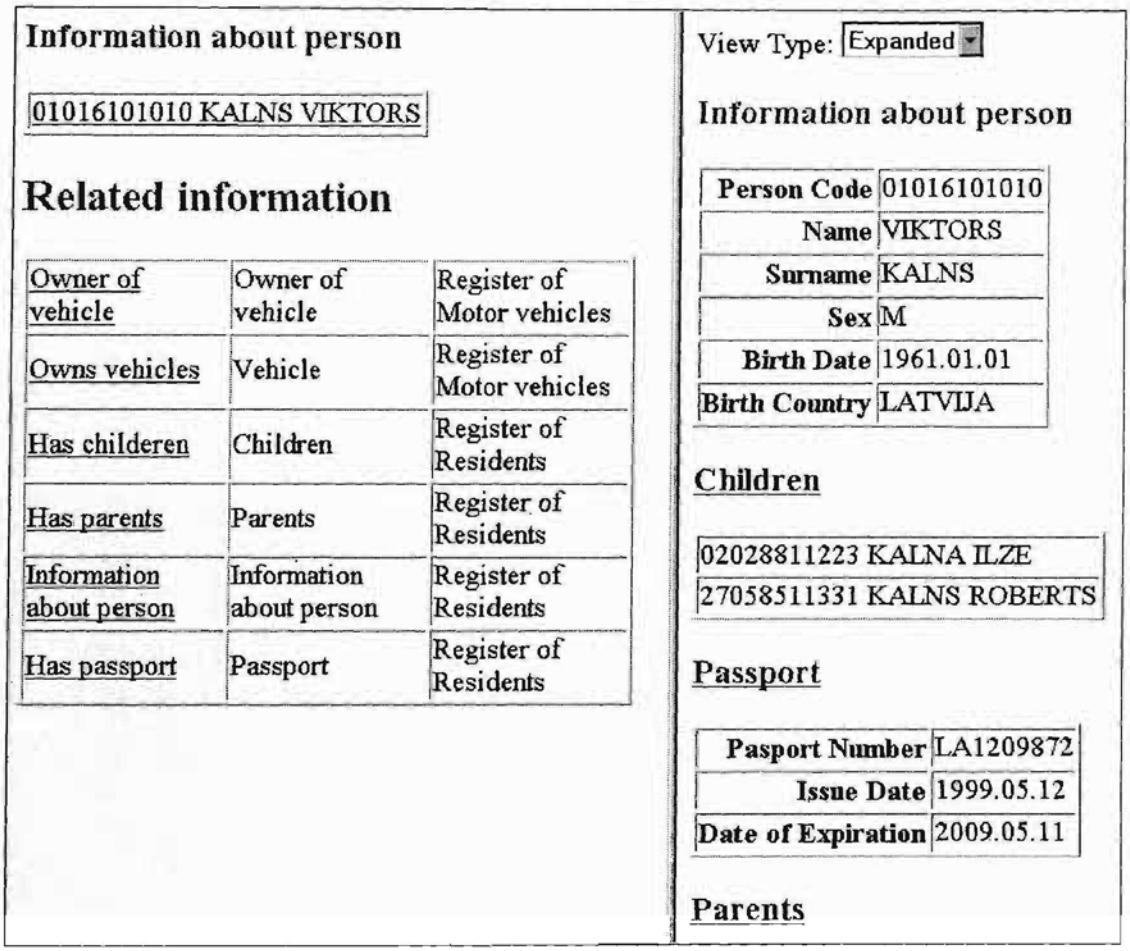
- Kreisajā pusē augšā ir saraksts ar visiem atrastajiem objektiem, kas atbilst meklēšanas kritērijiem. Uzklikšķinot uz konkrētā objekta, šis objekts kļūst par galveno objektu un labajā pusē skatāma izvērsta informācija par to;
- Kreisajā pusē apakšā ir saraksts ar visām saitēm, pa kurām var iet no konkrētā objekta - objekts, kas ir saites otrā pusē, saites nosaukums, reģistrs, kurā šis objekts atrodas;
- Uzklikšķinot uz konkrētās saites, saistītie objekti kļūst par galvenajiem objektiem un tiek parādīti augšējā kreisajā pusē. Piemēram, nospiežot uz saites *Information about person*, parādās skats, kāds redzams Zīm. 23. Labajā pusē ir izvērsta informācija par atlasīto objektu.



Zīm. 21 Meklēšanas kritēriju izvēle



Zīm. 22 Informācija par transportlīdzekļu īpašniekiem



Zīm. 23 Informācija par personu

Informāciju var skatīt vai nu saīsinātā variantā, vai arī izvērsti, kā arī var parādīt informāciju par visiem saistītajiem objektiem no tā paša datu avota.

6. Datu apmaiņas mehānismi Iedzīvotāju reģistrā

Šajā nodaļā aprakstīts Iedzīvotāju reģistrs un tā jaunā IS, kuras izstrādi autors vadīja laikā no 2000. līdz 2003.gadam. Šī sistēma tikai veidota kā pilotprojekts atbilstoši visām Megasistēmas nostādnēm.

6.1. Iedzīvotāju reģistra vēsture

Līdz ar tehnoloģiju attīstību radās iespējas veidot datorizētu iedzīvotāju uzskaites sistēmu - Iedzīvotāju reģistru. 1991. gada 11. decembrī tika pieņemts likums "Par iedzīvotāju reģistru" [Lat92], kas noteic šīs sistēmas darbības tiesisko pamatu. Šis likums stājās spēkā 1992. gada 1. janvārī un nosaka,

- kādas ziņas par personu iekļaujamas Iedzīvotāju reģistrā;
- ka ikvienai reģistrētajai personai jāpiešķir individuāls kods;
- ka Iekšlietu ministrijas Pilsonības un migrācijas lietu pārvalde (PMLP) vada Iedzīvotāju reģistra izveidošanu un izmantošanu.

1992. gada 30. martā PMLP reģistrēja pirmās personas un līdz 1993. gada marta beigām tika pabeigta iedzīvotāju pirmreģistrācija. 1996. gadā tika uzsākta pārvaldes teritoriālo nodaļu pārēja uz tiešsaistes datu pārraides režīmu.

1997. gadā sekmīgi tika nodota ekspluatācijā Nepilsoņu pasu drukāšanas programmatūra. Pirmo reizi IR vēsturē dokumenti tika drukāti saskaņā ar IR centrālajā DB iekļautajām ziņām. Tika uzsākts darbs pie IR iekļauto datu kvalitātes uzlabošanas programmas.

1997. aprīlī IR IS pārgāja uz programmatūru, kura tiek lietota līdz pat šim laikam un kura balstās uz DBPS Oracle.

1997. gada septembrī tika uzsākta programmatūras "WWW IRIS meklētājs" ekspluatācija, kas ļāva piekļūt IR datiem internetā. Daudzas valsts un pašvaldību institūcijas sāka apzināties IR datu nozīmīgumu savu funkciju veikšanā. 1998. gadā tika iegādāts jauns IR CDB serveru klasteris, izstrādāta jauna datu importa programmatūra, izveidota datu kvalitātes novērtēšanas programmatūra, kā arī likti pirmsākumi jaunai IR IS.

LR valsts pārvaldē līdz 1998.gadam bija raksturīga nesakārtotība, reģistru nekomunicēšanās savā starpā un datu apmaiņas neesamība praktiski visos LR valsts nozīmes reģistros, kā rezultātā dati tika nevajadzīgi dublēti, tie bija pretrunīgi dažādās sistēmās, kā arī nebija definētas kvalitātes prasības. Informācijas sakārtošanas nepieciešamība noteica nepieciešamību veidot Megasistēmu, un pēc Megasistēmas projekta uzsākšanas notika uzlabojumi arī Iedzīvotāju reģistrā.

1998. gadā tika izveidota IR vēsturē pirmā tiešsaistes datu izsniegšanas sistēma ar VID. Pieauga pieprasījums pēc IR informācijas - aizvien vairāk valsts un pašvaldību institūciju vēlējās saņemt IR informāciju, veidot savas IS uz IR datu pamata. Vairākām valsts institūcijām tika izveidoti tiešsaistes pieslēgumi IR CDB. IR dati tika izmantoti par pamatu 2000. gada tautas skaitīšanā.

1998. gadā pieņēma jaunu IR likumu [Lat98], saskaņā ar kuru valdība pirmo reizi noteica, kurām iestādēm kādas ziņas un kādos termiņos ir jāsniedz Iedzīvotāju reģistram.

2000. gadā tika uzsākts jaunas IR IS izstrādes projekts, kuru vadīja šī darba autors. Šī sistēma ir mērķtiecīgi veidota kā pilotprojekts atbilstoši visiem Megasistēmas principiem un prasībām. Jaunajā IR IS implementēti automatizēti datu apmaiņas mehānismi ar citām sistēmām, tajā skaitā informācijas saņemšana no citiem reģistriem un dokumentu, t.sk. personu apliecinošu dokumentu (pasu) drukāšana no sistēmas. Jaunā IR IS ir pabeigta un nodota pasūtītājam.

Autors kopā ar citiem vadošajiem IR projekta speciālistiem rekomendēja un palīdzēja IR izstrādāt kompleksu, visaptverošu un pilnīgu datu kvalitātes kontroles un uzlabošanas sistēmu, ietverot gan kvalitātes nodrošināšanas organizatoriskos darbus, gan tehnisko sistēmas risinājumu.

6.2. Glabājamo datu apjoms

IR darbība ir noteikta ar vairākiem normatīvajiem aktiem

- Pilsonības likums [Lat94],
- Iedzīvotāju reģistra likums[Lat98],
- Iedzīvotāju reģistrā iekļauto ziņu aktualizēšanas kārtība [MK02],
- Noteikumi par Iedzīvotāju reģistrā iekļauto ziņu izsniegšanu [MK03],
- Noteikumi par valsts nodevu par informācijas saņemšanu no Iedzīvotāju reģistra [MK99],
- u.c.

Saskaņā ar Iedzīvotāju reģistra likumu reģistrā par personu iekļaujamas šādas ziņas:

- Personas identificējošā informācija - personas kods,
- Personas vārds (vārdi), uzvārds,
- Ziņas par dzimšanu (datums, vieta) un dzimšanas reģistrāciju,

- Ziņas par dzimumu, tautību, valstisko piederību,
- Dzīvesvietas adrese,
- Ziņas par pasi vai citu personas apliecinošo dokumentu,
- Ziņas par ģimenes stāvokli, laulībām un laulāto,
- Ziņas par bērniem,
- Ziņas par vecākiem,
- Ziņas par rīcībspēju,
- Ziņas par emigrāciju,
- Ziņas par izraidīšanu no Latvijas,
- Ziņas par bezvēsts prombūtni,
- Ziņas par nāvi,
- Ārvalstniekiem – ziņas par uzturēšanās atļauju un ierašanos Latvijā.

Iedzīvotāju reģistrā ir aizliegts iekļaut šādas ziņas:

- Rase vai ādas krāsa,
- Reliģiskā pārliecība vai piederība pie kādas konfesijas,
- Politiskā pārliecība, piederība pie kādas politiskās partijas vai kustības, kā arī ziņas par politiskajiem uzskatiem,
- Seksuālā orientācija vai saslimšana.

6.3. Datu apmaiņa

Iedzīvotāju reģistrs, viena no Megasistēmas sastāvdaļām, sadarbojas ar daudzām iestādēm un sistēmām un 2003.gada beigās bija tiešsaistes režīmā ar 15 sistēmām. Atbilstoši Megasistēmas principiem tam būtu jābūt saistītam ar visām tām IS, kurās parādās personas dati. Tā kā šādu sistēmu Latvijā ir vairāk nekā 100, tad ir nepieciešams universāls mehānisms vieglai un ātrai datu apmaiņu izveidei. Lai to nodrošinātu, tika izveidots jauns datu apmaiņas mehānisms, kas balstās uz HTTP un XML. Šis mehānisms tika veidots viegli un ērti paplašināms un viegli konfigurējams.

Visus IR ārējos sadarbības subjektus var nosacīti iedalīt divās grupās - informācijas devēji un informācijas ņēmēji.

6.3.1 Informācijas devēji

Saskaņā ar MK noteikumiem "Iedzīvotāju reģistrā iekļauto ziņu aktualizēšanas kārtība", ziņas par personu Iedzīvotāju reģistrā iekļauto ziņu aktualizēšanai sniedz:

- Saeimas Kanceleja (par uzņemšanu Latvijas pilsonībā par īpašiem nopelniem Latvijas labā);
- Tiesa (par personas atzīšanu par rīcībnespējīgu un aizgādnības nodibināšanu, adopciju, laulības šķiršanu un laulības neesamību u.tml.);
- Tieslietu ministrijas Dzimtsarakstu departaments (par atkārtoti izsniegtu laulības apliecību, dzimšanas apliecību vai miršanas apliecību);
- Naturalizācijas pārvalde (par pilsonības piešķiršanu, atņemšanu un atjaunošanu);
- Pašvaldības dzimtsarakstu nodaļa (par laulības reģistrāciju, miršanas reģistrāciju, tautības ieraksta maiņu un papildinājumiem un labojumiem civilstāvokļa aktu reģistru ierakstos);
- Bāriņtiesas un pagasttiesas (par vecāku varas pārtraukšanu un vecāku varas atjaunošanu, aizbildnības un aizgādnības nodibināšanu un izbeigšanu);
- Pašvaldības un to iestādes, kuras atbild par personas deklarētās dzīvesvietas reģistrāciju, (par personu reģistrāciju dzīvesvietā un to noņemšanu no uzskaites dzīvesvietā, grozījumiem pašvaldību teritoriālajā dalījumā, nosaukumu piešķiršanu ielām, parkiem, laukumiem, namīpašumiem, kā arī par to pārdēvēšanu);
- LR diplomātiskās un konsulārās pārstāvniecības ārvalstīs (par laulības reģistrāciju, miršanas reģistrāciju, tautības ieraksta maiņu, ja civilstāvokļa akts reģistrēts LR diplomātiskajās un konsulārajās pārstāvniecībās ārvalstīs);
- Pašas personas (par izmaiņām reģistrā iekļautajās ziņās par sevi, saviem bērniem, kas ir jaunāki par 16 gadiem, un par personām, kuras ir viņu aizbildnībā vai aizgādnībā, ja ziņas mainījušās un attiecīgais fakts reģistrēts ārvalsts institūcijās).

6.3.2 Informācijas ņēmēji

Saskaņā ar MK noteikumiem "Iedzīvotāju reģistrā iekļauto ziņu aktualizēšanas kārtība", ziņas par personu IR ir tiesīgi saņemt:

- Latvijas iedzīvotāji, kuriem ir tiesības saņemt izziņu par to informāciju, kas par viņiem ir ievadīta IR;
- Pašvaldības, kuras ir ieinteresētas saņemt ziņas par savas teritorijas iedzīvotājiem;
- UR, kas ieinteresēts saņemt ziņas par uzņēmumu dibinātājiem, dalībniekiem un amatpersonām;
- Zemes kadastrs un Zemesgrāmata, kas ieinteresēti saņemt ziņas par nekustamā īpašuma īpašniekiem, kā arī Adrešu reģistrs.
- Nodokļu maksātāju reģistrs, kas ieinteresēts saņemt ziņas par nodokļu maksātājiem;
- Transportlīdzekļu reģistrs, kas ieinteresēts saņemt ziņas par transportlīdzekļu īpašniekiem;
- Iekšlietu ministrijas struktūrvienības, kas ieinteresētas saņemt visus IR datus, kurus izmanto personu identificēšanai un citiem mērķiem;
- Citas informācijas sistēmas un reģistri, kas darbojas Latvijā un ir ieinteresēti saņemt ziņas par personām un to dzīvesvietām.

Informācijas apmaiņa notiek ļoti dažādos veidos – gan papīra formā (izdrukas un veidlapas), gan elektroniski tiešsaistē un ar failiem.

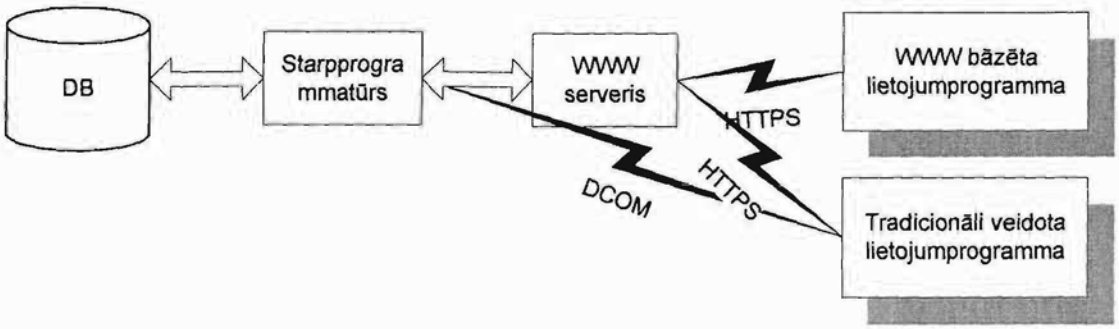
6.4. IR jaunās sistēmas koncepcija

Sakarā ar to, ka 1996. gadā izstrādātā programmatūra neapmierināja IR prasības ne pēc glabājamo datu apjoma, ne drošības un elastīguma ziņā, tika pieņemts lēmums veidot jaunu IR programmatūru.

IR jaunā programmatūra ir veidota daudzslāņu arhitektūras veidā (Zīm. 24). Šādā veidā izstrādājot programmatūru, tiek izdalīti 3 līmeņi:

- datu bāze,
- datu apstrādes loģika,
- lietotāja saskarne.

Šāda veida tehnoloģija ļauj mainīt kādu slāni, atstājot nemainīgus pārējos. Piemēram, vienai datu bāzei un datu apstrādes loģikai iespējams izveidot vairākas lietotāja saskarnes.



Zīm. 24 Daudzslāņu IR IS arhitektūra

Darbs ar programmatūru notiek pēc darba vietu principa. Darba vieta ir tāds programmatūras, datu un tiesību kopums, kas atļauj darbiniekam veikt visas ikdienas darbā nepieciešamās darbības. Piemēram, pasu inspektora darba vieta, sistēmas administratora darba vieta. Lai realizētu sadalījumu pa darba vietām, tiek izmantots SIA "Datorikas Institūts" izstrādātais CASE rīks ISTehnoloģija, kurš pēc autora iniciatīvas un autoram aktīvi līdzdarbojoties tika IR projekta ietvaros pielāgots IR vajadzībām.

6.4.1 Datu bāze

Datu bāzes galvenās funkcijas ir:

- Datu saglabāšana,
- Pamatintegritātes nodrošināšana, piemēram, starp kodifikatoru un tabulu, kur izmanto kodētās vērtības,
- Atomāro operāciju nodrošināšana darbam ar datiem, piemēram, vienā tabulā esošo datu ievads, labošana, dzēšana, kā arī visas atlasīšanas funkcijas.

DB ir veidota DBPS Oracle vidē. Primāro/ārējo atslēgu mehānisms tiek izmantots, lai nodrošinātu pamatintegritāti. Darbs ar datiem tiek veikts, izmantojot glabātās procedūras. Visi SQL teikumi, kas tiek izmantoti sistēmā, realizēti glabātajās procedūrās, un darbs ar datiem notiek, tikai un vienīgi izmantojot šīs procedūras. Tas nodrošina to, ka viss SQL kods darbam ar datiem glabājas datu bāzē, līdz ar to tiek atvieglota datu bāzes uzturēšana, kā arī datu bāzes un pieprasījumu optimizēšana, piemēram, indeksu izveide utt.

6.4.2 Starpprogrammatūra

Starpprogrammatūras slāņa galvenās funkcijas ir šādas:

- biznesa loģikas nodrošināšana,

- datu bāzes transakciju nodrošināšana,
- iespēja pieslēgties dažādiem datu devējiem un ņēmējiem caur vienotu autorizācijas un piekļuves mehānismu.

Starpprogrammatūras slāni var sadalīt vairākās loģiskās daļās:

- datu bāzes saskarne;
- biznesa loģika.

Starpprogrammatūras slānī ir izvietota datu apstrādes loģika jeb tā sauktā biznesa loģika. Šādi veidojot sistēmu, ir iespējams veidot dažādas lietotāja saskarnes (WWW bāzēta, tradicionālā u.c.), koncentrējot datu apstrādes loģiku vienā vietā un tādējādi paplašinot sistēmas iespējas, samazinot izmaksas un iespējamo kļūdu daudzumu.

Starpprogrammatūras slānim var pieslēgties arī lietotāji no citām aplikācijām, kas izmanto IR datus, tādējādi uzlabojot savu specifisko aplikāciju darbu (piemēram, no Rīgas domes). Starpprogrammatūras slāņa funkcijas ir izveidotas pietiekoši universālas, lai ar šīm funkcijām varētu strādāt gan CDB operatori, gan citas IS.

Izmantojot starpprogrammatūras slāni, datu bāzē tiek ievadīti gan CDB operatoru savadītie dati, gan importa moduļa importētie dati, gan ārējo sistēmu veidotie dati, kas tiešsaistes režīmā slēdzas pie IR IS. Tas dod iespēju vienā vietā (starpprogrammatūrā) koncentrēt gan datu kvalitātes procedūras, gan izveidot vienotu mehānismu transakciju apstrādei.

Starpprogrammatūra veidota ar *Microsoft Visual Basic*, par transakciju monitoru izmantojot *Microsoft Transaction Server*. Starpprogrammatūras slānis veidots tā, ka to var darbināt uz vairākiem datoriem paralēli, viena datora sabojāšanās gadījumā automātiski novirzot pieprasījumus uz citiem datoriem. Tā tiek iegūtas divas priekšrocības:

- tiek paaugstināts sistēmas drošums,
- tiek nodrošināta datu ielādes sabalansēšana, dinamiski sadalot slodzi starp serveriem.

6.4.3 Lietotāja saskarne

Lietotāja saskarnei ir šādi galvenie uzdevumi:

- nodrošināt ērtu darbu ar datiem - ievadi, meklēšanu, labošanu, dzēšanu, apskati,

- nodrošināt ērtu vidi citu sistēmā nepieciešamo darbību izpildei – administrēšanu, importu/ eksportu.

Lietotāja saskarne veidota *Microsoft Visual FoxPro* vidē. Datu attēlošanai plaši tiek izmantotas WWW tehnoloģijas – XML, HTML. Lietotāja saskarne sadarbojas ar starpprogrammatūras slāni, nevis tiešsaistē ar datu bāzi. Tā kā visa datu apstrādes loģika ir koncentrēta starpprogrammatūras slānī, tad ir viegli veidot dažādas lietotāja saskarnes datu ievadei un attēlošanai, līdz ar to ar CDB var strādāt gan izmantojot tradicionāli veidotu programmatūru, gan WWW bāzētu, un abas šīs saskarnes izmanto vienu un to pašu starpprogrammatūras funkcionalitāti.

Datu apskatei ir izveidota WWW bāzēta, specifiska UP versija. UP darbojas, izmantojot starpprogrammatūras slāni un kopīgo autorizācijas mehānismu. UP ir paredzēts, lai IR datus varētu apskatīt ārējie lietotāji, piemēram, policija, robežsardze. Tā kā UP ir integrēts ar divām citām IeM sistēmām – nederīgo dokumentu bāzi un meklēšanā esošo personu bāzi, pārskatot datus par personu, automātiski tiek parādīts arī, ja persona ir meklēšanā vai personas dokuments ir izsludināts par nederīgu. Plānots UP integrēt arī ar citām IeM sistēmām, tādā veidā paplašinot UP funkcionalitāti.

6.5. Tehniskie risinājumi

Jaunajā IR ir vairāki inovatīvi tehnoloģiskie risinājumi, kuru izstrādē autors līdzdarbojās un kas padara šo reģistru par vienu no modernākajiem reģistriem Latvijā:

- Pieeja datiem iespējama tikai caur IR IS programmatūru, izmantojot vienotu lietotāju un tiesību vadību un vienotu datu apstrādes loģiku visām apakšsistēmām. Līdz ar to ir garantija, ka neatkarīgi no tā, kādā veidā lietotājs datus saņem vai ievada IR, tiek nodrošinātas vienādas tiesības, vienots audita mehānisms un vienādas datu kvalitātes pārbaudes;

- Katram lietotājam var definēt tiesības ar precizitāti līdz konkrētai funkcijai. Atsevišķām funkcijām var noteikt līmeni, kādā lietotājs šo funkciju var darbināt – no tā, piemēram, var mainīties lietotājam redzamo datu apjoms;

- Konfigurējamas datu kvalitātes pārbaudes – atkarībā no lietotāja tiesībām un pieredzes var nokonfigurēt dažādas datu kvalitātes prasības – dažiem lietotājiem tiek iedotas plašākas tiesības ievadīt nepilnīgus datus;

- 3 līmeņu arhitektūra. Tradicionālā vidē FoxPro veidotā ievadprogramma nodrošina ļoti bagātīgu lietotāja saskarnes iespēju klāstu, krietni plašāku nekā WWW lapām. Plānots datu apmaiņu no ievadprogrammas uz starpprogrammatūru pārnest pilnībā uz HTTPS protokolu – tas atvieglo

sistēmas uzturēšanu, jo IR nodaļas atrodas visā Latvijā un ir izveidotas tiešsaistes ar visām nodaļām. Ja izmanto DCOM vai citu līdzīgu protokolu, tad kļūst diezgan sarežģīta tīkla konfigurēšana. HTTPS izmantošana to daudzkārt atvieglo;

- Jaunajā IR IS datu apmaiņām plaši tiek lietots XML, kas atļauj izmaiņu gadījumā viegli pielāgot arī datu apmaiņu;
- Datu izsniegšanai ārējām sistēmām tiek izmantota modificēta UP versija IR ziņu aplūkošanai. Šajā gadījumā caur UP var skatīt saistītos datus no IR, nederīgo dokumentu DB un meklēšanā esošo personu DB;
- Konfigurējams, uz metamodeli bāzēts Universālais datu apmaiņas rīks, kas atvieglo datu apmaiņu ar ārējām sistēmām;
- Paredzēts izveidot autonomu datu kvalitātes kontroles rīku, kas kontrolēs tās kvalitātes, kuras ir ļoti sarežģīti izkontrolēt datu ievades laikā vai arī kuras var uz laiku tikt atļautas sistēmā. Piemēram, ja jāievada pirmuzskaites informācija par ģimeni, kas iebrauc Latvijā, tad vecāku ievadīšanas laikā bērni tiek ievadīti kā radinieki un pēc tam bērniem tiek ievadīta lielā pirmuzskaite;
- Sistēmā ir iebūvēts mehānisms lielu un sarežģītu atskaišu izveidei laikā, kad sistēmā ir mazāka noslodze, un saglabāšana datu bāzē vēlākai izmantošanai. Paredzēts ar laiku izveidot datu noliktavu informācijas analīzei.

6.6. Megasistēmas principi Iedzīvotāju reģistra sistēmā

IR ir viens no svarīgākajiem Megasistēmas reģistriem, un Megasistēmas kontekstā ļoti svarīga ir datu apmaiņa starp IR un citām sistēmām.

6.6.1 Datu pareizības un aktualitātes nodrošināšana

Lai panāktu visu Megasistēmas sistēmu datu pareizību un aktualitāti. Lai to nodrošinātu, nepieciešams izpildīt vairākus priekšnoteikumus:

- Dati tiek ņemti no vienas sistēmas, nevis atkārtoti ievadīti;
- Dati netiek atkārtoti prasīti personai, bet ņemta kopija, līdz ar to tiek ietaupīts laiks un personai nav nepieciešams nēsāt izziņas no viena ierēdņa pie otra – nepieciešamās ziņas tiek saņemtas elektroniski;

- Dati tiek ievadīti to rašanās vietā un elektroniski pārsūtīti sistēmai, kas ir atbildīga par šiem datiem. Atbildīgā sistēma pēc tam datus elektroniskā formā izplata citām ieinteresētajām sistēmām.

Jaunā sistēma tiek veidota ar iespēju veikt dažādas datu apmaiņas ar citām sistēmām, tādā veidā izpildot Megasistēmas principus.

6.6.2 Sadarbība ar citām IS

Sadarbībai ar citām sistēmām IR jaunajā sistēmā ir paredzēti vairāki mehānismi:

- datu sniegšana no IR,
- interesējošo personu datu monitorings,
- datu saņemšana tiešsaistē un ar failiem,
- pasu izdruka, izmantojot IR.

Datu apmaiņa ar ārējām sistēmām notiek, izmantojot interneta tehnoloģijas – HTTPS protokolu un XML formātus.

6.6.3 Dokumentu izsniegšana

Viena no svarīgākajām Megasistēmas prasībām ir dokumentu izsniegšana no sistēmas, tādā veidā novēršot nesakritības starp ziņām dokumentos un ziņām sistēmā. Kopš 2002. gada visas pases tiek drukātas, izmantojot IR ziņas, līdz ar to ir novērsta iespēja, ka ziņas pasē un IR nesaskanēs.

Saskaņā ar Personu apliecinošo dokumentu likumu [Lat02b] un tā grozījumiem [Lat04], no 2005. gada 1. janvāra Latvijā paredzēts sākt izdot personas apliecības, bieži vien sauktas par identifikācijas kartēm (ID kartes), ar pamatinformāciju par iedzīvotāju. ID kartes būs mašīnlasāmas, līdz ar to šo informāciju varēs izmantot arī sistēmās, kas nav pieslēgtas tiešsaistē IR. Tāpat ir paredzēts, ka daļa no informācijas būs pārrakstāma (piemēram, ziņas par personas dzīvesvietu). Šīs ziņas varēs pārrakstīt arī pašvaldībās, līdz ar to iedzīvotājam nebūs jāapmeklē PMLP dzīvesvietas maiņas gadījumā.

Latvijā ir pieņemts elektronisko dokumentu likums [Lat02a]. Paredzēts, ka ID kartes kalpos par personas identifikatoru saziņai ar dažādām institūcijām. ID kartēs paredzēts glabāt iedzīvotāja elektronisko parakstu, līdz ar to persona varēs izmantot ID karti, lai parakstītu elektroniskos dokumentus, ko gribēs nosūtīt dažādām iestādēm.

Nākotnē paredzēts paplašināt dažādus servisus iedzīvotājiem, piemēram, dodot iespēju elektroniskā veidā veikt dzīvesvietas deklarēšanu, kā arī pasūtīt izziņas un jaunus dokumentus no Iedzīvotāju reģistra.

6.7. Iedzīvotāju reģistra datu struktūra

IR konceptuālais datu modelis ir attēlots attēlā Zīm. 25. Centrālais objekts ir *Persona.* Personas ir divu veidu –

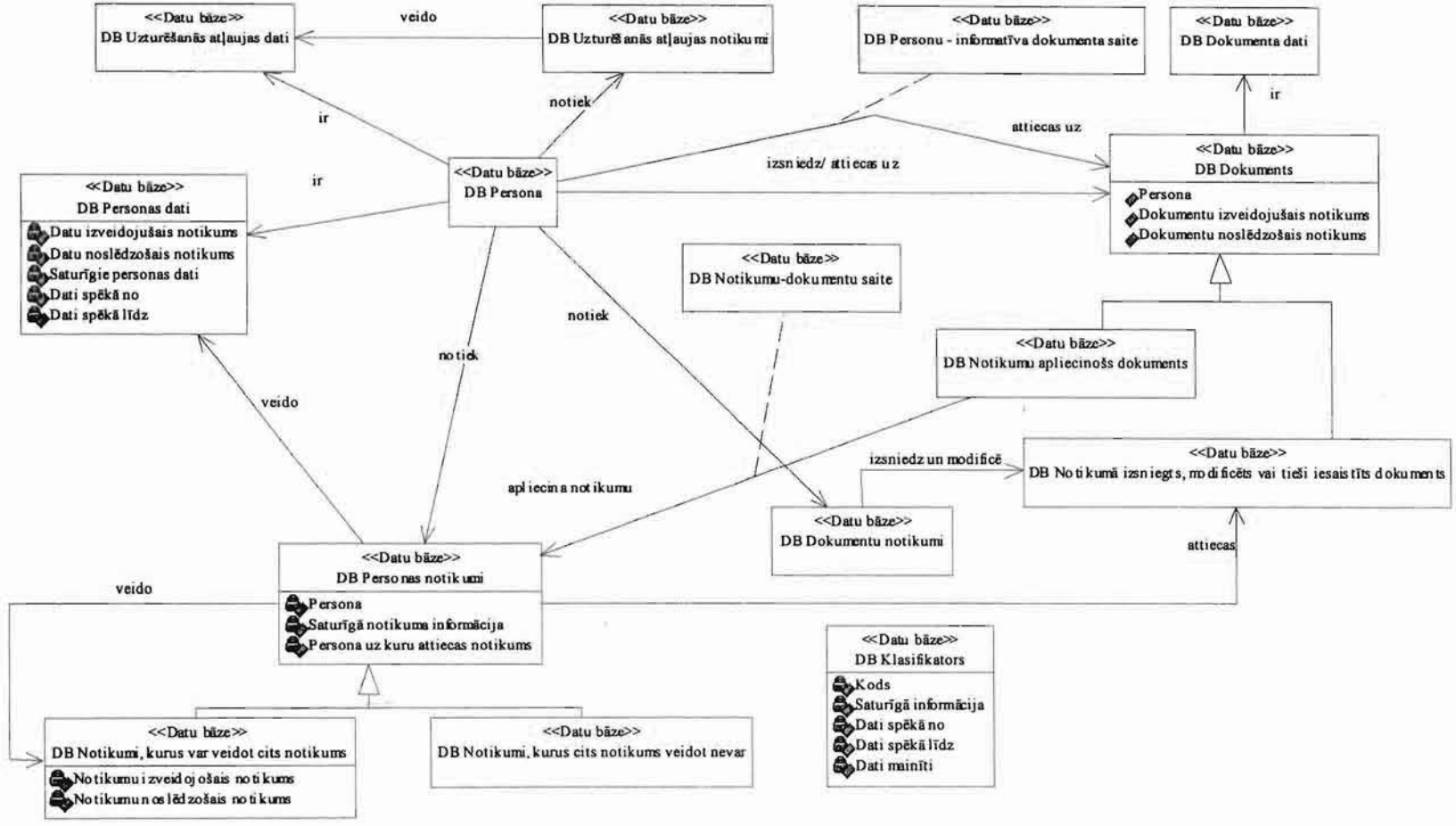
- IR reģistrētās personas (t.i. tās, kurām ir piešķirts personas kods);
- IR nereģistrētās personas (IR reģistrēto personu vecāki, bērni un laulātie bez personas koda).

Personai ir dati 17 datu grupās – *Vārda dati, Dzimšanas dati, Laulības dati* utt. Katrā datu grupā personai var būt vairāki dati, kas var būt vai nu aktuāli, vai arī vēsturiski. Dati ir divu tipu –

- tādi, kas personai vienmēr ir un vienmēr ir tieši vieni (piemēram, uzvārds),
- dati, kas var būt patvaļīgā skaitā (piemēram, deklarētās dzīvesvietas).

Notikums ir vēl viens pamatobjekts. Katri dati sākas un beidzas ar kādu notikumu. Notikums var iesākt un/vai izbeigt vairākus datus, pie tam, viens notikums var vienus datus iesākt, bet citus datus izbeigt. IR ir aptuveni 70 dažādi notikumi. Notikumi var būt salikti, piemēram, laulības reģistrācijas notikuma ietvaros var tikt mainīts personas uzvārds, t.i. notikt vārda maiņas notikums. Katram notikumam ir viens vai vairāki dokumenti, kas apstiprina šo notikumu, kā arī var būt dokuments, kas radies šī notikuma rezultātā.

Personai piesaistīti ir *dokumenti.* Dokumenti attiecas uz personu vai ir piesaistīti notikumam, kas attiecas uz personu. Dokumenti ir vairāku tipu – personu apliecinošie, reģistri, apliecības un informatīvie dokumenti.



Zīm. 25 IR datu bāzes konceptuālais modelis

Tabula 3 IR datu bāzes konceptuālā modeļa apraksts

Datu bāzes objekts vai objektu abstrakcija	**Apraksts**
DB Dokuments	Datu bāzes objektu abstrakcija, kura pārstāv datu bāzē uzkrājamos dokumentus
DB Dokumenta dati	Datu bāzes objektu abstrakcija, kura pārstāv dokumentu datus
DB Dokumenta notikums	Datu bāzes objektu abstrakcija, kura pārstāv notikumus, kas attiecas uz dokumentu
DB Notikumu apliecinošs dokuments	Datu bāzes objektu abstrakcija, kura pārstāv dokumentus, kuri kalpo par notikumus (datus) apliecinošiem dokumentiem
DB Notikumā izsniegts, modificēts vai tieši iesaistīts dokuments	Datu bāzes objektu abstrakcija, kura pārstāv dokumentus, kuri ir notikumā izsniegti, modificēti vai tieši iesaistīti
DB Notikumu – dokumentu saite	Atbilstošā datu bāzes tabula "SAITE_DOKUMENTS_NOTIKUMS"
DB Persona	Atbilstošā datu bāzes tabula "PERSONA"
DB Personas dati	Datu bāzes objektu abstrakcija, kura pārstāv personas datus
DB Personas notikumi	Datu bāzes objektu abstrakcija, kura pārstāv uz personu attiecošos notikumus. Katram personas notikumam atbilst savs datu bāzes objekts
DB Notikumi, kurus cits notikums veidot nevar	Datu bāzes objektu abstrakcija, kura pārstāv uz personu notikumus, kurus cits notikums veidot nevar

Datu bāzes objekts vai objektu abstrakcija	**Apraksts**
DB Notikumi, kurus var veidot cits notikums	Datu bāzes objektu abstrakcija, kura pārstāv uz personu notikumus, kurus var veidot cits notikums
DB Uzturēšanās atļaujas dati	Datu bāzes objektu abstrakcija, kura pārstāv personas uzturēšanās atļaujas datus
DB Uzturēšanās atļaujas notikumi	Datu bāzes objektu abstrakcija, kura pārstāv personas uzturēšanās atļaujas notikumus
DB Personu – dokumentu saite	Datu bāzes objektu abstrakcija, kas pārstāv saišu tabulas starp personām un tām izsniegtiem dokumentiem. Šādas tabulas ir definētas tikai tiem dokumentiem, kuri var tikt izsniegti vairāk nekā vienai personai. Konkrēti, datu bāzes objektu abstrakcija "DB Personu – informatīva dokumenta saite", kuru pārstāv tabula "SAITE_DOK_INF_DOK_PERS". Pārējiem dokumentiem saites funkciju pilda atbilstošs lauks konkrētā dokumenta tabulā.
DB Klasifikators	Datu bāzes objektu abstrakcija, kas pārstāv datu bāzē lietotos klasifikatorus

6.8. Datu apmaiņas vispārējs apraksts

Tā kā IR ir nepieciešama datu apmaiņa ar daudzām sistēmām un dažādos veidos, šī nepieciešamība noteica arī datu apmaiņas mehānismu dažādību. Ar dažādām sistēmām pēc 4.nodaļas klasifikācijas ir implementēti šādi datu apmaiņas modeļi:

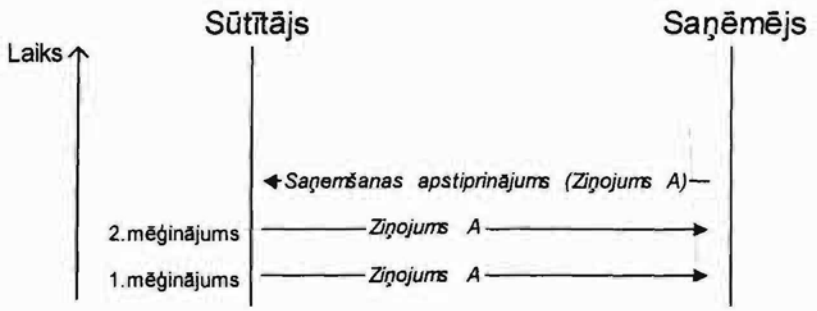
- Sinhrona atlase no viena reģistra. Datus no IR tiešsaistē ņem vairākas lielas sistēmas – UR, TR, PVIS u.c.;

- Sinhrona atlase no vairākiem reģistriem. Lieto vairāki tūkstoši ārējo lietotāju, kas skatās datus no IR, nederīgo dokumentu datu bāzes un meklēšanā esošo personu datu bāzes;
- Datu apmaiņa sinhroni. IR saņem datus no PVIS un Pasu IS un raksta datus Pasu IS;
- Datu sniegšana asinhroni. Šādā režīmā IR datu daļu kopijas regulāri saņem lielākā daļa pašvaldību. Regulāri datu kopijas tiek sniegtas arī Centrālajai statistikas pārvaldei un citām iestādēm.

Datu apmaiņas formāts ir izveidots tā, lai tas būtu pēc iespējas neatkarīgs no vides, kas nodrošina datu apmaiņu, tādējādi panākot programmatūras un datu apmaiņas protokola unifikāciju gan tiešsaistes datu apmaiņai, gan datu apmaiņai ar failu palīdzību.

Datu apmaiņas procesā informācijas pamatvienība ir ziņojums. Ziņojums tiek noformēts XML formātā.

- Tiešsaistes režīmā tiek pieņemts, ka ziņojumi nonāks no sūtītāja līdz adresātam;
- Ziņojumiem, kas tiek nodoti ar failu palīdzību, tiek izmantots īpašs mehānisms (Zīm. 26). Lai nodrošinātu, ka nesaistes režīmā visi ar failu palīdzību nosūtītie ziņojumi tiek saņemti, ir ieviests īpašs ziņojums "Saņemšanas apstiprinājums", līdz kura saņemšanai sūtītājs turpina atkārtoti sūtīt ziņojumus, par kuriem nav saņemts apstiprinājums. Savukārt ziņojuma saņēmējam pēc ziņojuma saņemšanas ir obligāti jānosūta "Saņemšanas apstiprinājums".



Zīm. 26 Datu apmaiņas failu veidā shēma

Standarta tiešsaistes režīmā datu apmaiņas sesija notiek pēc Zīm. 27 attēlotās biznesa procesa shēmas.

Zīm. 27 Tiešsaistes datu apmaiņas sesijas biznesa procesa shēma

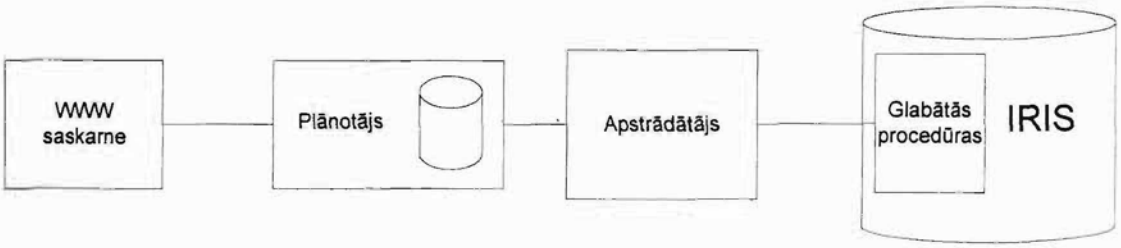
Jebkura tiešsaistes datu apmaiņas sesija sākas ar Klienta pieslēgšanos IR IS un beidzas ar Klienta atslēgšanos no IR IS. Pēc veiksmīgas pieslēgšanās Klients var veikt dažādas darbības – pieprasīt personas datus, nosūtīt datu aktualizācijas pieprasījumu un citus.

Visi ziņojumi, kurus Klients nosūta IR IS, satur transakcijas identifikatoru, kas sastāv no divām daļām – ārējās sistēmas (Klienta) identifikatora un unikāla kārtas numura. IR IS atbildes ziņojums uz Klienta ziņojumu satur šo pašu transakcijas identifikatoru. Šādā veidā tiek sasieti kopā Klienta ziņojumi ar IRIS atbildēm uz tiem.

Importa/ eksporta procedūras ir veidotas kā modulāra sistēma. Tiešsaistes un nesaistes procedūras izmanto dažus kopīgus moduļus, un daži moduļi ir atšķirīgi katrai konkrētai sistēmai.

Nesaistes (failu) importa/ eksporta shēma ir attēlota Zīm. 28. Sistēma sastāv no šādām daļām:

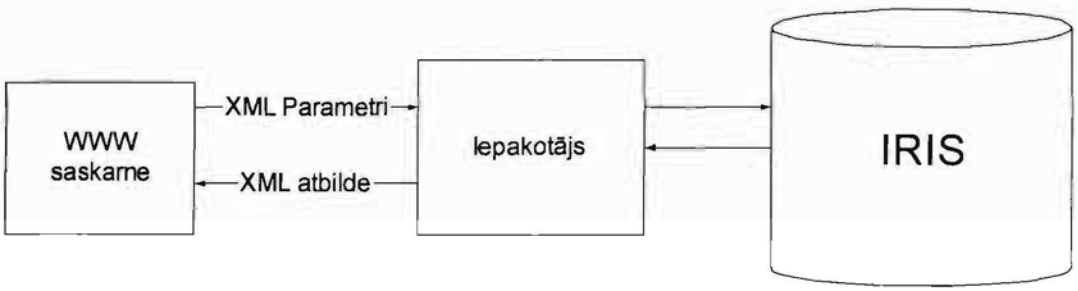
- WWW saskarnes, kur var norādīt, kuru datu apmaiņu veiks, kā arī papildus parametrus konkrētajai datu apmaiņai, piemēram, failu atrašanās vietas, datumus utt.,
- Plānotāja, kurā glabājas informācija par veicamajiem darbiem,
- VB moduļa, kurš māk darbināt Oracle procedūras,
- Oracle glabātās procedūras, kas pēc uzdotajiem parametriem veic datu importu/eksportu.



Zīm. 28 Failu importa/eksporta shēma

Tiešsaistes importa/ eksporta shēma ir attēlota Zīm. 29. Sistēma sastāv no šādām daļām:

- WWW saskarnes, caur kuru var veikt IR IS funkciju izsaukumus,
- VB iepakotāja, kas veic translāciju no XML izsaukumiem uz IR IS izsaukumiem un atpakaļ,
- IR IS moduļiem, kas veic nepieciešamās funkcijas.



Zīm. 29 Tiešsaistes importa/ eksporta shēma

6.9. Datu izsniegšana

Datu saņemšana no IR ir svarīga dažādām IS, kurās ir personas dati. Piemēram, uz UR atnāk fiziska persona, kas dibina uzņēmumu. Kad persona saskaras ar Uzņēmumu reģistru, tad UR informāciju par personu ņem no IR – tiešsaistē informācijas ievades laikā tiek veikts pieprasījums no UR uz IR un saņemta UR nepieciešamā aktuālā informācija par personu – vārds, uzvārds dzimšanas dati u.c. Līdz ar to šī informācija nav jāievada atkārtoti no personas dokumentiem – ietaupās laiks un samazinās kļūdu iespēja.

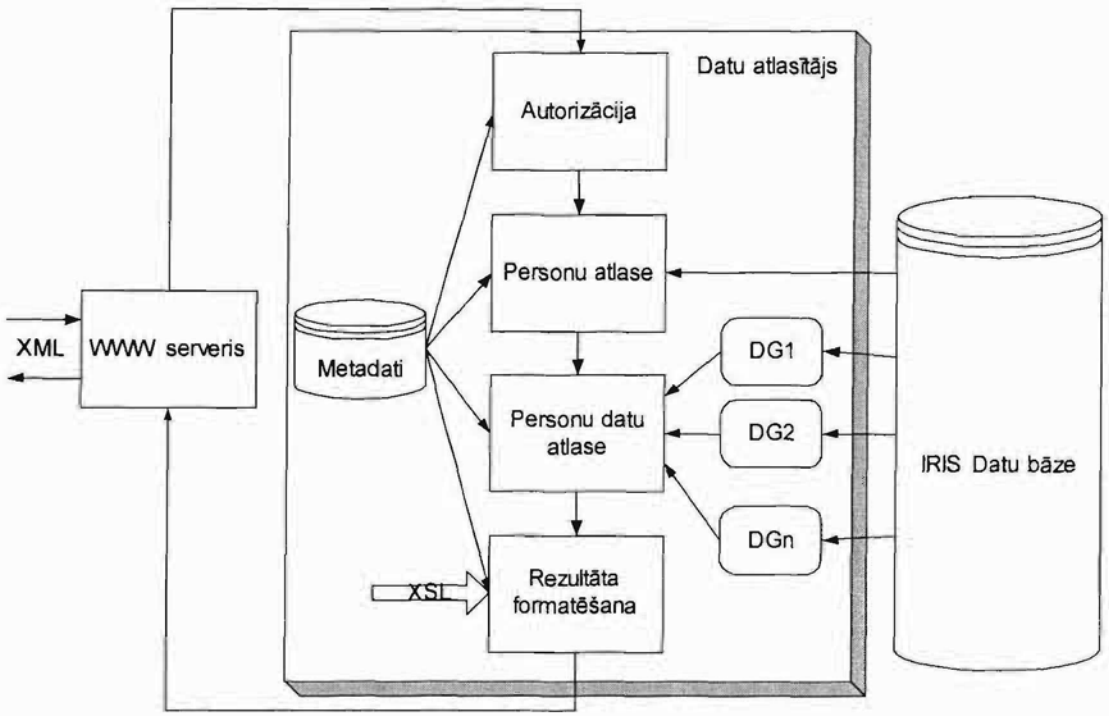
Tā kā ir svarīgi izveidot datu apmaiņas mehānismu, kurā būtu pēc iespējas vienkāršāk konfigurēt un modificēt datu apmaiņas formātus, IR jaunajā sistēmā ir iestrādāts universāls datu sniegšanas mehānisms (Zīm. 30), kas atļauj administratoram konfigurēt esošas sasaistes un definēt jaunas bez papildus programmēšanas, norādot tikai, kādi dati

un kādā formātā ir jādod lietotājam. Tas radikāli vienkāršo un paātrina esošo sasaistu modificēšanu un jaunu sasaistu izveidi.

Sistēmā metadatu bāzē glabājas informācija par ārējiem lietotājiem, tiem pieejamajām funkcijām, kā arī to interesējošām datu grupām. Tur ir arī informācija par XSL failiem, ar kuriem ir jāveic katra konkrētā pieprasījuma atbildes pēcapstrāde.

Kad ārējais lietotājs pieprasa informāciju no IR, šis pieprasījums tiek apstrādāts universālā veidā:

- WWW lapa, kas saņem pieprasījumu, izsauc pieprasījumu apstrādes moduli;
- Šis modulis no pieprasījuma noskaidro lietotāja identifikatoru un pieprasījuma tipu, un veic nepieciešamo autorizāciju;
- Pēc tam no metadatu bāzes tiek noskaidrots, kādas datu grupas šajā pieprasījumā ir nepieciešamas, un tiek atlasīta no IRIS informācija par attiecīgajām datu grupām;
- Iegūtie rezultāti tiek pievienoti atbildes XML failam, kuru nodot pēcapstrādei;
- Pēcapstrādes rezultāta formatēšanas modulis atkarībā no pieprasījuma tipa un ārējā lietotāja veic XML pārveidi ar atbilstošo XSL failu;
- Iegūtais rezultāts tiek nosūtīts pieprasītājam - ārējam lietotājam.



Zīm. 30 Datu izsniegšanas mehānisms

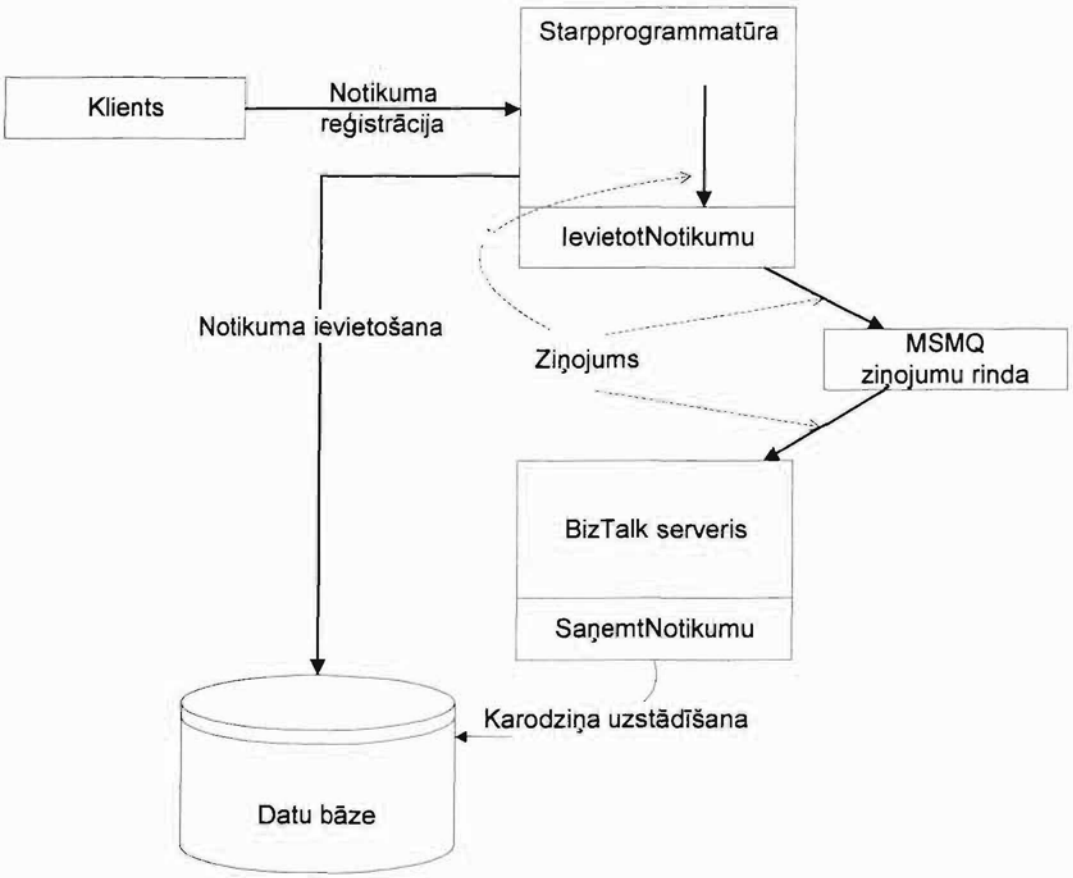
6.10. Datu monitorings

Sistēmām, kas satur datus par personu, ir svarīgi zināt, kas ir noticis ar personu. Piemēram, UR ir svarīgi zināt, vai uzņēmuma direktors ir vai nav mainījis valstisko piederību vai uzvārdu.

Lai nodrošinātu šādas informācijas piegādi, jaunajā IR sistēmā ir izveidots datu monitoringa mehānisms (Zīm. 31):

- Ārējās sistēmas var pieteikt IR, kuras personas un par kādiem datiem ir jāmonitorē, t.i kuras personas ietilpst šo sistēmu kompetences grupā;
- Ja personai mainās ārējo sistēmu interesējošie dati, tas tiek piereģistrēts IR monitoringa apakšsistēmā;
- Ārējā sistēma laiku pa laikam sazinās ar IR un noskaidro, kurām viņu interesējošām personām dati ir mainījušies.

Nākotnē ir paredzēts papildus serviss – līdzko dati par personu mainās, tā ziņas tiek automātiski nosūtīts ārējai sistēmai.



Zīm. 31 Monitoringa shēma

Datu monitorings tiek veikts pēc sekojošas shēmas:

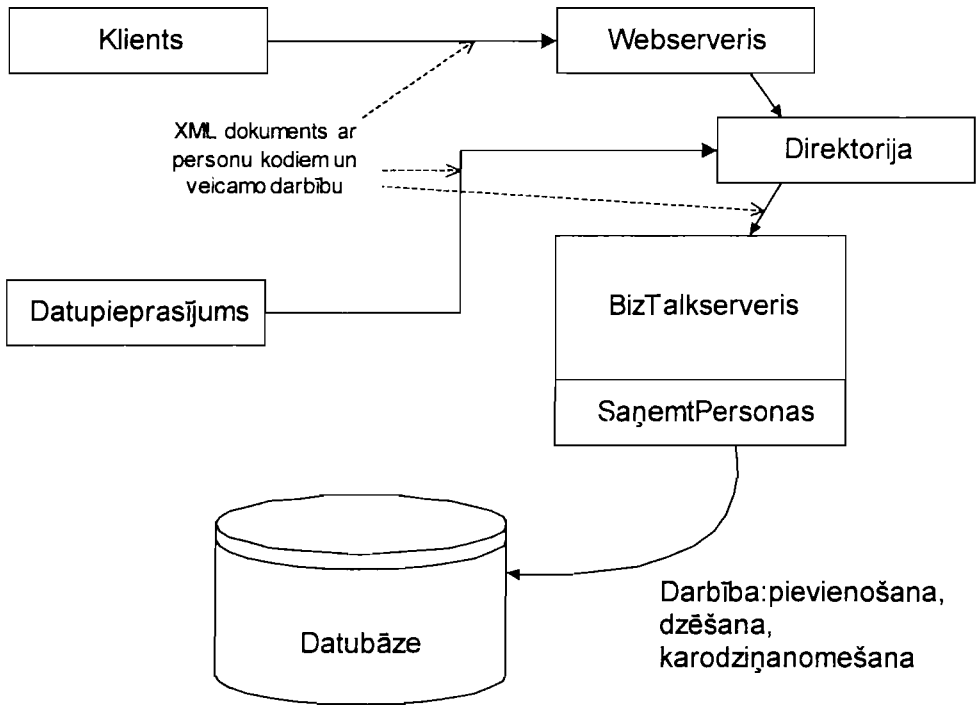
- Kad sistēmā tiek ievadīts kāds notikums, tas tiek ielikts gan IR IS datu bāzē, gan arī MSMQ rindā. No turienes BizTalk serveris šo notikumu paņem un konstatē, kuriem ārējiem klients šī persona un tās izmainījušais datu apjoms interesē, un "paceļ" attiecīgo karodziņu;
- Ārējais klients var saņemt sarakstu ar kompetences kopā ietilpstošo personu personas kodiem, kurām ir mainījusies interesējošā informācija;
- Kad ārējais klients saņem informāciju par interesējošo personu, tad karodziņš tiek "nolaists".

Ārējās IS var pievienot vai izmest interesējošās personas (Zīm. 32) no kompetences personu kopas divējādi:

- Klients sūta pieprasījumu par personas pievienošanu/ dzēšanu no kompetences personu kopas;

- Persona kompetences personu kopai tiek pievienota/ izdzēsta sakarā ar kāda notikuma notikšanu vai datu maiņu pēc uzdotiem likumiem.

Ja var uzdot nosacījumus, kuros gadījumos personai atrasties kompetences personu grupā, tad, izpildoties šiem nosacījumiem, persona automātiski tiek ievietota kompetences personu grupā. Piemēram, VOAVA kompetences personu grupā ietilpst visas tās personas, kuras ir LR pastāvīgie iedzīvotāji, t.i. Latvijas valsts piederīgie ar Latvijā deklarētu dzīvesvietu un ārzemnieki ar uzturēšanās atļaujām Latvijā ilgāk par pusgadu. Ja šādi precīzi nosacījumi nav iespējami, tad ārējais klients nosaka, kuras personas ietilpst kompetences personu grupā (CSDD, UR u.c.).



Zīm. 32 Intereses personu pievienošana/dzēšana

Datu pieprasīšana par interesējošajām personām tiek veikta, izmantojot datu pieprasīšanas moduli, kurš jau iepriekš tika aprakstīts 6.9. nodaļā.

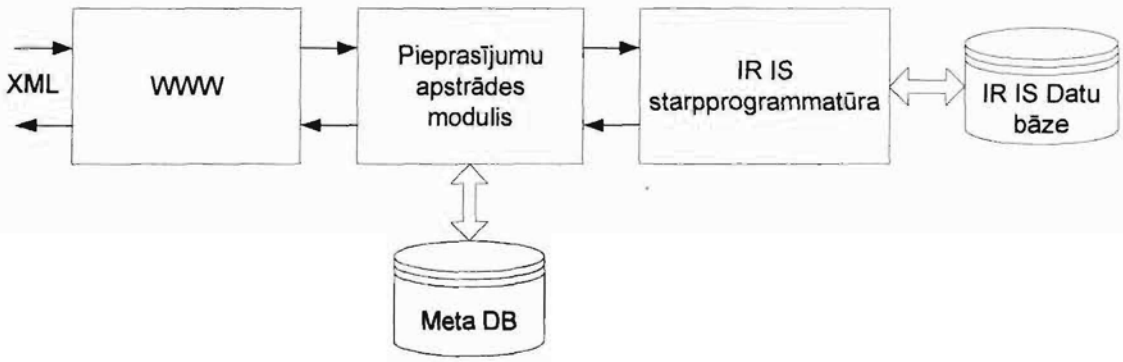
6.11. Datu ievade no ārējiem datu avotiem

IR glabājamā informācija tiek reģistrēta daudzās iestādēs, līdz ar to ir svarīgi šo informāciju savlaicīgi un bez kļūdām ievadīt IR. Vienīgais pieņemamais veids ir elektroniska datu nodošana no ārējās sistēmas uz IR. Piemēram, ziņas par deklarēto

dzīvesvietu tiek reģistrētas pašvaldībā un pēc tam šīs ziņas tiek nogādātas IR. Labākais veids no operativitātes viedokļa ir tiešsaiste starp pašvaldības informatīvo sistēmu un IR.

Jaunajā IR sistēmā ir izveidoti mehānismi (Zīm. 33), kas realizēti tiešsaistes režīmā:

- ārējām sistēmām datu ievadei IR sistēmā. Tiek izmantota vienota starpprogrammatūra, līdz ar to tiek garantēts, ka no ārējām sistēmām nākušie dati nenojauks sistēmas iekšējo integritāti;
- datu ievadei no ārējiem datu avotiem. Piemēram, no dzimtsarakstiem – dzimšanas, miršanas un laulības informācija, no pašvaldībām – dzīvesvietas informācija. Ir izveidotas atbilstošas WWW lapas un iepakotāji, kas piesūtītos XML failus ar notikumiem pārveido IR IS starpprogrammatūrai saprotamā formātā un izsauc atbilstošo IR IS starpprogrammatūras notikumu ievades, labošanas vai dzēšanas procedūru. Pie apstrādes ietilpst arī dažādu objektu (personu, konkrēto notikumu, adreses un tās elementu u.c.) identificēšana un sistēmas iekšējo kodu atrašana un piešķiršana.



Zīm. 33 Ārējo datu izmaiņu apstrādes shēma

6.12. Problēmas datu apmaiņā

Veidojot IR datu apmaiņu ar citām sistēmām, tika risinātas vairākas problēmas:

- Problēma, kas rodas tāpēc, ka IR reģistrē notikumus, kas notiek reālā dzīvē un kuru notikšana nav atkarīga no dažādiem dokumentiem utml. Piemēram, personas dzimšana un miršana. Persona piedzimst vai nomirst, un pēc tam ar kaut kādu laika nobīdi šīs ziņas tiek ievadītas konkrētā sistēmā, līdz ar to dati sistēmā objektīvu iemeslu pēc atpaliek no situācijas reālajā dzīvē.
- Problēma – ko darīt, ja dati no datu avota nesakrīt ar tiem datiem, kas ir sistēmā. Piemēram, tiesa piesūta ziņas par laulības šķiršanu personai, kas nav precējusies. Šajā gadījumā ir jāsaprot, kur ir problēma – vai IR nesatur datus

par personas laulību, kas ir notikusi, vai arī tiesa ir atsūtījusi nekorektus datus.

Šīs problēmas katru reizi ir jārisina individuāli un to dara IR personāls. Lai saņemtos datus tomēr varētu reģistrēt IR IS, sistēmā tika iestrādāta iespēja konkrētiem lietotājiem atslēgt lielāko daļu no datu kvalitātes pārbaudēm (izņemot kritiskās datu kvalitātes pārbaudes). Šādā veidā atbildība par konkrētajā gadījumā veiktajām izmaiņām ir jāuzņemas personām, kuras veic datu izmaiņas, kas neatbilst kvalitātes pārbaužu prasībām. Lai samazinātu risku, kvalitāšu kontroles tiek atslēgtas tikai atsevišķiem augstākās kvalifikācijas speciālistiem, kuri vislabāk pārzina Iedzīvotāju reģistra darbību gan no likumdošanas viedokļa, gan no reģistrācijas procesa viedokļa, gan arī IR IS darbību, un pilnībā apzinās, kādas sekas var būt katrā konkrētā gadījumā no nekvalitatīvu datu ievades. Lai varētu kontrolēt, cik un kādi dati neatbilst kvalitātes prasībām, ir paredzēts izveidot autonomu kvalitātes kontroles rīku, kas regulāri veiktu datu kvalitātes kontroles un signalizētu par visām atrastajām neatbilstībām datos.

Nobeigums

Strādājot Megasistēmas projektā un ar to saistītajos projektos, autors ir ieguvis plašas teorētiskās zināšanas un praktisko pieredzi sistēmu integrācijas jomā. Praktiski veiktā sistēmu izstrāde ir pierādījusi lietoto sistēmu integrācijas principu pareizību. Pārbaudītie sistēmu integrācijas principi būtu jāpielieto, veicot Megasistēmas tālāko attīstību un citu sistēmu integrāciju. Autors cer, ka Megasistēma tiks tālāk attīstīta, ka būs iespējas dalīties savā pieredzē ar citiem šīs jomas speciālistiem un ka arī pašam būs iespējas savu pieredzi pielietot citu sistēmu integrācijas projektos.

Lai varētu veiksmīgi Latvijā ieviest e-pārvaldi, nepieciešams turpināt Megasistēmas attīstīšanu, integrējot tajā citas sistēmas. Lai to varētu veiksmīgi izdarīt, būtu jāveic vairāki darbi:

- Jaunu sistēmu iesaistīšana datu apmaiņā, tostarp arī privāto sistēmu, piemēram, banku sistēmu. Plašāka datu izmantošana stimulēs datu kvalitātes uzlabošanos;
- Reģistru reģistra uzturēšana un paplašināšana. Lai atvieglotu jaunu sistēmu pievienošanu Megasistēmai, Reģistru reģistrā papildus būtu nepieciešams pielikt saskarņu aprakstus – kā piekļūt konkrēto sistēmu datiem. Šādas saskarnes varētu definēt kā XML modeļus vai arī WSDL valodā aprakstīti tīmekļa servisu *(web service)* izsaukumi, šādā veidā Reģistru reģistrs saturētu UDDI direktoriju;
- Jāpaplašina XML lietošana datu apmaiņā starp sistēmām. Lai izvairītos no viena piegādātāja slazdiem, būtu vēlams noteikt XML kā vienu no galvenajiem datu apmaiņas formātiem. Tāpat būtu vēlams noteikt, ka dažādu piegādātāja specifisku protokolu vietā (SQL*Net utml) jālieto vispārpieņemti protokoli (HTTPS utml.). Šādā veidā tiktu atvieglota datu apmaiņa starp sistēmām;
- Jāpaplašina Komunikāciju servera funkcionalitāte. Pašlaik KS un UP var izmantot tikai datus atlasīšanai un tos interpretē cilvēks. Būtu vēlams komunikāciju serverim pievienot tīmekļa servisu funkcionalitāti, kas atļautu izmantojot vienu resurspunktu, slēdzoties klāt dažādiem datu avotiem;
- Jāpaplašina UP funkcionalitāte, implementējot [AK02b] aprakstīto WWW lapu aprakstīšanas valodu. Šādā veidā UP kļūtu vēl elastīgāks un ar vēl lielākām iespējām konfigurēt dažādiem lietotājiem dažādu izskatu un uzvedību.

Pašlaik UP implementētajam globālajai modelim un tā izveidei piemīt daudzas labas GAV un LAV īpašības. Modeli ir viegli paplašināt vai mainīt saskaņā ar izmaiņām datu avotos. Tomēr tam ir ierobežotas iespējas izpildīt sarežģītus pieprasījumus ar datu atlasi no vairākiem datu avotiem vienlaicīgi. Būtu lietderīgi sakombinēt UP pieeju globālā modeļa izveidei ar pieprasījumu apstrādes iespējām, kādas ir DATALOG sistēmām, kā arī pārnest UP īpašības uz GAV un LAV modeļiem.

Darbā lietoto jēdzienu definīcijas un saīsinājumu skaidrojumi

Jēdzienu definīcijas

Datu integrācija. Modeļu integrācija plus globālā pieprasījumu apstrāde.

Eksporta modelis. Tā komponentes modeļa daļa, pie kuras konkrētā lokālā datu bāze ļauj piekļūt federālajai sistēmai.

Federālā datu bāzu sistēma. Vairāku autonomu, bet sadarbojošos lokālu datu bāzu kopums, kuras piedalās federācijā, lai varētu kopīgot datus. Lokālās datu bāzes var darboties neatkarīgi un to integrācijas pakāpe var būt dažāda.

Federālā shēma. Vairāku eksporta modeļu integrēts modelis.

Globālais modelis. Konceptuāli apvienots vairāku datu bāzu modelis sadalītā datu bāzu sistēmā.

Globāls kā skats. Modeļu kartēšanas veids, kurā globālais modelis tiek izteikts kā skats no lokālajiem modeļiem.

Heterogēna sadalīta datu bāzu sistēma. Sistēma, kurai ir heterogēnas sastāvdaļas, kas var būt aparatūrā, operētājsistēmā, datu modelī, BDPS, pieprasījumu valodā vai modelī.

Homogēna sadalīta datu bāzu sistēma. Sistēma, kurai ir viena loģiskā datu bāze, kas ir fiziski sadalīta, kopā ar sadalītu datu bāzu pārvaldības sistēmu, kas nodrošina pieprasījumu izpildi un datu izmaiņu veikšanu.

Informācijas sistēma (IS) Vairāku datu bāzu un reģistru apvienojums, kas nodrošina tā daudzfunkcionālu un multitematisku lietojumu.

Komponentes modelis. Lokālais modelis, kas izteikts kopīgajā federālajā datu bāzes modelēšanas valodā.

Komunikāciju serveris (KS). Pakalpojumu kopums, kas plašam lietotāju lokam (gan Latvijas, gan ārvalstu) nodrošina iespēju saņemt informāciju no vairākiem avotiem (valsts reģistriem, datu bāzēm, IS), izmantojot vienu resurspunktu. Šāda pieeja nodrošina vienotu politiku datu pieejas tiesībām, kā arī atļauj ar vienu identifikācijas reizi piekļūt vairākiem datu avotiem.

Lokālais modelis. Lokālais datu bāzes modelis, kas ir izteikta šīs datu bāzes modelēšanas valodā.

Lokāls kā skats. Modeļu kartēšanas veids, kurā lokālie modeļi tiek izteikti kā skati no globālajiem modeļiem.

Megasistēma. Integrētu valsts nozīmes informācijas sistēmu un to darbību regulējošu normatīvo aktu apvienojums, kas nodrošina to efektīvu darbību, augstu datu ticamību, funkcionēšanas drošību un lietošanas ērtību.

Nozares informācijas sistēma (NIS). Informācijas sistēma, kas risina vienas nozares vai problemātikas jautājumus.

Reģistrs. Datu krātuves un tās objektu juridiskas reģistrācijas procedūru apvienojums. Dotajā darbā, ja nav norādīts citādi, tiks aplūkoti tikai datorizēti reģistri, kur datu krātuve ir realizēta ar datu bāzes palīdzību.

Reģistru reģistrs (RR). IS, kura satur informāciju par Latvijā izstrādātajām IS un tajās glabājamajiem datu objektiem. Reģistru reģistrā glabājas gan vispārīga informācija par sistēmu (nosaukums, saturs, īpašnieks, likumdošanas bāze u.c.), gan arī sistēmas datu modelis, kā arī informācija par saistību starp dažādos reģistros esošajiem datiem.

Modeļu integrācija. Vairāku atsevišķu datu avotu modeļu apvienošana vienotā globālā modelī.

Valsts nozīmes informācijas sistēma (VNIS). Informācijas sistēma, kuras izveidošana un uzturēšana ir noteikta ar likumu vai Ministru Kabineta noteikumiem.

Darbā izmantoto saīsinājumu skaidrojumi

Saīsinājums	Skaidrojums
BAV	*Both as view*
BGLAV	*Both global and local as view*
BVVDPT	Baltijas Valstu valdību datu pārraides tīkls
CORBA	*Common request broker architecture*
CSDD	Ceļu satiksmes drošības direkcija
DB	Datu bāze
DBA	Datu bāzes administrators
DBPS	Datu bāzu pārvaldības sistēma
DCOM	*Distributed component object model*
DDL	*Data definition language*
DML	Datu manipulācijas valoda
EDB	*Extensional DataBase*

Saīsinājums	**Skaidrojums**
EIF	*Export Import format*
ER	*Entity-relationship*
FDBS	Federālo datu bāzu sistēma
GAV	Pieeja "globāls kā skats" (*Global as view*)
GDBPS	Globālā datu bāzes pārvaldības sistēma
GLAV	*Global and local as view*
IDA	*Interchange of data between administrations*
IDB	*Intensional DataBase*
IeM	Iekšlietu ministrija
IMS	*Information Management System*
IR	Iedzīvotāju reģistrs
IS	Informācijas sistēma (sinonīms informatīvā sistēma)
IT	Informācijas tehnoloģijas (sinonīms informatīvās tehnoloģijas)
J2EE	*Java 2 Enterprise Edition*
KS	Komunikāciju serveris
LAV	Pieeja "lokāls kā skats" (*Local as view*)
LDBPS	Lokālā datu bāzes pārvaldības sistēma
LR	Latvijas Republika
MDBS	Multidatu bāzu sistēma
MK	Ministru kabinets
MSMQ Server	*Microsoft message queue server*
NP	*Non polinomial*
P2P	*Peer-to-peer*
PK	Personas kods

Saīsinājums	Skaidrojums
PMLP	Pilsonības un migrācijas lietu pārvalde
RPC	*Remote procedure call*
SOAP	*Simple object access protocol*
SQL	Pieprasījumu valoda *Structured Query Language*
TIS	Tiesu informatīvā sistēma
TR	Transportlīdzekļu reģistrs
UDDI	*Universal Description, Discovery, and Integration*
UML	*Unified modelling language*
UP	Universālais pārlūks
UR	Uzņēmumu reģistrs
VID	Valsts ieņēmumu dienests
VOAVA	Veselības obligātās apdrošināšanas valsts aģentūra
WSDL	*Web services description language*
WWW	*World Wide Web*
XML	*Extensible markup language*
XSL	*Extensible Stylesheet Language*

Atsauces

Citu autoru darbi

[AK97] N. Ashish, C. Knoblock. **Semi-automatic Wrapper Generation for Internet Information Sources**. In Second IFCIS International Conference on Cooperative Information Systems (CoopIS), Charleston, SC, 1997

[Cali03] A. Calì. **Reasoning in data integration system: why LAV and GAV are siblings**. in Proc. of the 14th International Symposium on Methodologies for Intelligent Systems (ISMIS), 2003

[Che76] P. P. Chen. **The entity-relationship model: Toward a unified view of data**. ACM Transactions on Database systems 1(1):9-36, 1976

[CT92] B. Czejdo, M. Taylor. **Integration of information systems using an object-oriented approach**. The Computer Journal 35(5), 1992

[D02] К. Дж. Дейт. **Введение в системы баз данных**. Вильямс, Москва, 2002

[DAT87] S. Deen, R. Admin, M. C. Taylor. **Data integration in distributed databases**. IEEE TSE 13(7), 1987

[ERS99] A. Elmagarmid, M. Rusinkiewicz, A Sheth, editors. **Management of Heterogenous and Autonomous Database Systems**. Morgan Kaufmann Publishers Inc., 1999

[GRA00] **GRADE User guide**. Infologistik GmbH, 2000

[GUW02] H. Garcia-Molina, J. Ullman, J. Widom. **Database Systems: The Complete Book**, Prentice hall, 2002

[IDA00] **Interchange of Data between Administrations** http://europa.eu.int/ISPO/ida/

[Hal03] A. Halevy. **Data integration: A status report**. German Database Conference (BTW), Leipzig, Germany, 2003

[HBP94] A. R. Hurson, M. W. Bright, H. Pakzad. **Multidatabase systems: An advanced solution for global information sharing**. Los Alamitos, CA: IEEE Computer Society Press, 1994

[HGIP95] Hammer J, Garcia-Molina H, Ireland K, Papakonstantinou Y, Ullman J, Widom J. **Information translation, mediation, and Mosaic-based browsing in the TSIMMIS system.** in Proceedings of ACM SIGMOD

International Conference on Management of Data, Project Demonstration, 1995

[Hil04] J. Hill. **The future of data integration technologies**. A Meta group white paper, 2004

[HIST03] A. Halevy, Z. Ives, D. Suciu, I. Tatarinov. **Schema Mediation in Peer Data Management Systems**. International Conference on Data Engineering, March 2003

[HMNT99] L. Haas, R. Miller, B. Niswonger, M Tork Roth, P Schwarz, E. Wimmers. **Transforming Heterogeneous Data with Database Middleware: Beyond Integration.** Data Engineering Bulletin, 1999

[Ives02] Z. Ives. Efficient **Query Processing for Data Integration**. A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy University of Washington, 2002

[Kar01] E. Karnitis. **E-Government: An Innovative Model of Governance in the Information Society**. Baltic IT&T Review, 1, 2001

[KLK91] R. Krishnamurthy, W. Litwin, W. Kent. **Language features for interoperability of databases with schematic discrepancies**. Proceedings of the ACM SIGMOD, pages 40-49. New York: ACM, May 1991

[Lat92] Latvijas republikas likums "**Par iedzīvotāju reģistru**", http://www.nais.lv, 1992

[Lat94] Latvijas republikas likums "**Pilsonības likums**", Latvijas vēstnesis, 11.08.1994

[Lat98] Latvijas republikas likums "**Iedzīvotāju reģistra likums**", Latvijas vēstnesis, 10.09.1998

[Lat02a] Latvijas republikas likums "**Elektronisko dokumentu likums**", Latvijas vēstnesis, 20.11.2002

[Lat02b] Latvijas republikas likums "**Personu apliecinošu dokumentu likums**", Latvijas vēstnesis, 05.06.2002

[Lat04b] Latvijas republikas likums "**Grozījumi Personu apliecinošu dokumentu likumā**", Latvijas vēstnesis, 20.04.2004

[Len02] M. Lenzerini. **Data Integration: A theoretical perspective**, In proceedings of PODS 2002, pages 233-246. ACM, 2002

[Lev00a] A. Levy. **Logic-based techniques in data integration**. In J. Minker, editor, Logic Based Artificial Intelligence. Kluwer Academic Publishers, 2000

[Lev00b] A. Y. Levy. **Answering Queries Using Views**: A Survey. Submitted for publication, 2000

[LMR90] W. Litwin, L. Mark, N. **Roussopoulos. Interoperability of multiple autonomous databases**. ACM Computing surveys 22(3):267-293, September 1990

[LNEM89] J. Larson, S. B. Navathe, R. El-Masri. **A theory of attribute equivalence and its applications to schema integration**. IEEE Transactions on Software Engineering 15(4):449-463, April 1989

[LV03] P. Lyman, H. R. Varian. **How Much Information 2003** http://sims.berkeley.edu/research/projects/how-much-info-2003/execsum, UC Berkeley's School of Information Management and Systems, 2003

[MA00] M. Zur Muehlen, R. Allen. **Workflow Classification Embedded & Autonomous Workflow Management Systems**. Workflow Management Coalition, 2000

[McBP03] P. McBrien, A. Poulovassilis. **Data integration by bi-directional schema transformation rules**. In: 19th International Conference on Data Engineering, ICDE'03, March 5 - March 8, 2003

[MEGA98a] Integrētās valsts nozīmes informācijas sistēmas (Megasistēmas) projekts. **Prasības primārajiem reģistriem**. Latvijas Universitāte, 1998

[MEGA98b] http://www.mega.lv

[MK02] Ministru kabineta noteikumi Nr.454 "**Iedzīvotāju reģistrā iekļauto ziņu aktualizēšanas kārtība**", Latvijas vēstnesis, 11.10.2002

[MK03] Ministru kabineta noteikumi Nr.322 "**Iedzīvotāju reģistrā iekļauto ziņu izsniegšanas kārtība**", Latvijas vēstnesis, 21.06.2003

[MK99] Ministru kabineta noteikumi Nr.119 "**Noteikumi par valsts nodevu par informācijas saņemšanu no Iedzīvotāju reģistra**", Latvijas vēstnesis, 26.03.1999

[MoT98a] **The Latvian national program "Informatics"**, Ministry of Transportation, 211 pp., 1998

[MoT98b] **The Latvian national program "Informatics" (summary)**, Ministry of Transportation, 60 pp., 1998

[RR95] V. Ramesh, S. Ram. **A methodology for interschema relationship identification in heterogenous databases**. In proceedings of the Havaii International conference os Systems and Sciences, pages 263-272, 1995

[RUGV] http://www.it-gov.ru

[SKS02] A. Silberschatz, H. F. Korth, S. Sudarshan. **Database system concepts**. McGraw-Hill, 2002

[SL90] A. Sheth, J. Larson. **Federated database systems for managing distributed, heterogenous, and autonomous databases**. ACM Computing Surveys 22(3):183-236, September 1990

[SLCN88] A. Sheth, J. Larson, A. Cornelio, S. B. Navathe. **A tool for integrating conceptual schemata and user views**. Proceedings of the fourth international conference on data engineering, pages 176-183. Los Alamitos, CA: IEEE Computer Society Press, February 1988

[SKS02] A. Silberschatz, H. Korth, S. Sudarshan. **Database system concepts**. Fourth edition. McGrawHill, 2002

[SP94] S. Spaccapietra, C. Parent. View integration: **A step forward in solving structural conflicts**. IEEE Transactions on Knowledge and Data Engineering 6(2), April 1994.

[SOGV] http://e.finland.fi

[TRV96] A. Tomasic, L. Raschid, P. Valduriez. **Scaling heterogeneous databases and the design of disco**. In Proceedings of the International Conference on Distributed Computer Systems, 1996

[UKGV] http://www.govtalk.gov.uk

[XE02] L. Xu, D. Embley. **Combining the Best of Global-as-View and Local-as-View for Data Integration**. Department of Computer Science Brigham Young UniversityProvo, Utah 84602, U.S.A., 2002

Autora publikācijas starptautiskos izdevumos un zinātnisko konferenču rakstu krājumos

[ABK00a] G. Arnicans, J.Bicevskis, G. Karnitis. **The Unified Megasystem of Latvian Registers: Development of a Communication Server – the First Results and Conclusion**, in Abstracts of Papers of 4th International Conference "Information Technologies and Telecommunications in the Baltic States" , pages 163-168, Riga, 2000

[ABK99] G. Arnicans, J. Bicevskis, G. Karnitis. **The Concept of Setting Up a Communications Server**, in Abstracts of Papers of 3rd International Conference "Information Technologies and Telecommunications in the Baltic States" , pages 48-57, Riga, 1999

[ABKK02] G. Arnicans, J. Bicevskis, G. Karnitis, E. Karnitis. **Smart Integrated Mega-System as a Basis for E-governance**. Proceedings D of the 5th International Multi-Conference Information Society IS'2002, pages 197-202, Ljubljana, 2002

[AK00] G. Arnicans, G. Karnitis **Heterogeneous Database Browsing in WWW Based on Meta Model of Data Sources**, in Proceedings of the 4th IEEE International Baltic Workshop, edited by Albertas Čaplinskas, Vol. 1, pages 175-187Vilnius, Lithuania, 2000

[AK01] G. Arnicans, G. Karnitis. **Heterogeneous Database Browsing in WWW Based on Meta Model of Data Sources**, in Selected Papers of 4th IEEE Baltic Workshop on Databases and Information Systems, pages 167-178, Kluwer Academic Publishers, 2001

[AK02a] G. Arnicans, G.Karnitis. **Semantics for Managing Systems in Heterogeneous and Distibuted Environment**. H-M. Haav, A. Kalja (Eds), Databases and Information Systems II, Selected Papers from the Fifth International Baltic Conference, BalticDB&IS'2002, pages 149-160, Kluwer Academic Publishers, 2002

[AK02b] G. Arnicans, G.Karnitis. **Semantics for Managing Systems in Heterogeneous and Distibuted Environment**. In Hele-Mai Haav and Ahto Kalja, editors, Databases and Information Systems, Proceedings of the Fifth International Baltic Conference BalticDB&IS 2002, Vol1., pages 51-62, Tallinn, 2002

Citas autora publikācijas, referāti, raksti

[ABKK01] G. Arnicans, J. Bicevskis G. Karnitis, E. Karnitis. **The Mega-system: integration of National information systems**. Conceptual and Methodological Baselines, in Latvian Academic Library Gray Literature database, 2001

[ABK00b] G. Arnicans, J.Bicevskis, G.Karnitis. **Development of a Communication Server: First results and Conclusions**, Baltic IT Review, No.17(2), 2000, page 29-32

[BK98] J.Bicevskis, G.Karnitis. **Problems in the Integration of Registers of State Significance in Latvia**, Baltic IT Review, No.1, 1998, page75

[BK02] Я. Бичевскис, Г. Карнитис. **Принципы и опыт построения системы национальных регистров (Мегасистемы) в Латвии**, IX Byelorussian congress on telecommunications, information and bank technologies TIBO'2002, Minsk, April, 2-5, 2002

[Kar02] Ģ. Karnītis. **Problēmas un risinājumi sadalītās heterogēnās datu bāzu sistēmās**, LU 60. konference, Rīga, 2002

[Kar03a] Г. Карнитис. **Регистр Жителей - проблемы и решения**. X Byelorussian congress on telecommunications, information and bank technologies TIBO'2003, Minsk, April, 1-4, 2003

[Kar03] Ģ. Karnītis. **Datu apmaiņa sadalītajās datu bāzēs**. LU 61. konference, Rīga 2003

[Kar04] Ģ. Karnītis. **Datu apmaiņas problēmas informatīvajās sistēmās**. LU 62. konference, Rīga 2004