ISSN 1407-7493 COMPUTER SCIENCE 2008-7493
DATORZINĀTNE

TECHNOLOGIES OF COMPUTER CONTROL

DATORVADĪBAS TEHNOLOĢIJAS

APPLICATIONS OF MULTI-AGENT SYSTEMS IN THE MIDDLEWARE OF COMPUTER NETWORKS

DAUDZAĢENTU SISTĒMU PIELIETOJUMI DATORTĪKLU STARPPROGRAMMATŪRĀ

Andrejs Ermuiza, Dr. sc. comp., docents
Riga Technical University, Faculty of Computer Science and Computer Engineering,
Institute of Computer Control, Engineering and Technology,
Address: Meza 1, LV-1048, Riga, Latvia
Phone: +371 67558172
E-mail: ermuiza@edi.lv

Jevgenijs Ishchenko, student
Riga Technical University, Faculty of Computer Science and Computer Engineering,
Institute of Computer Control, Engineering and Technology,
Address: Meza 1, LV-1048, Riga, Latvia
Phone: +371 26495339
E-mail: verminous@gmail.com

Atslēgas vārdi: dalītas sistēmas, aģenti, daudzaģentu sistēmas, starpprogrammatūra, datu integrācija.

Ievads

Daudzaģentu sistēmu (DAS) teoriju [1, 2] pašlaik uzskata par vienu no perspektīvākajiem virzieniem datortīklu pastāvīgas klātbūtnes (ubiquity) attīstībai, jo uz šo teoriju balstās datortīklu lietojumu integrācija un tīmekļa (web) servisa jaunāko versiju (piem., semantiskais Web3.0) attīstība. DAS spēj nodrošināt zināšanu integrāciju un dažādu lietojumu sadarbspēju (interoperatibility), kas ļauj veikt efektīvu tīmekļa izmantošanu pakalpojumu saņemšanai un veiksmīgu sadarbību starp dažādu arhitektūru sistēmām uzņēmumu datortīklos. Uz DAS teorijas pašlaik bāzējas izkliedēto (dalīto – distributed) sistēmu decentralizētās pārvaldības teorijas attīstība un jaunāko datortīklu tehnoloģiju, kā arī ar tām saplūstošo plaši lietojamo mobilo tīklu protokolu izstrāde. Darba mērķis – izanalizēt daudz aģentu lietojumu nozīmi datortīklu starpprogrammatūrā un pārbaudīt to dažu lietojumu efektivitāti.

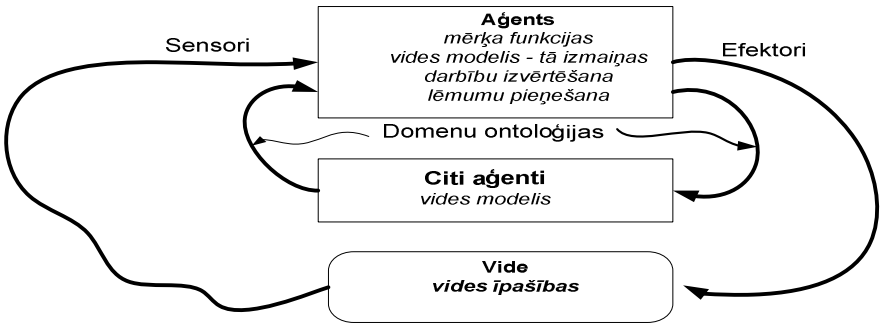
1. Aģenti, daudzaģentu sistēmas un to attīstība

Programmatūras aģents (PA) ir abstrakta, no objekt-orientētās programmēšanas (OOP) koncepcijas attīstījusies sistēma:

- kam ir mērķa funkcijas, zināšanas (priekšstats par vidi) sensori un efektori, ar kuriem sadarboties ar vidi,
- kas autonomi lemj kādas darbības veikt konkrētā situācijā.

Starpība starp OOP un aģentu programmatūras koncepcijām ir tāda, ka: OOP koncepcija paredz, ka katrs objekts strikti izpilda tam dotās pavēles, bet PA koncepcija paredz, ka:

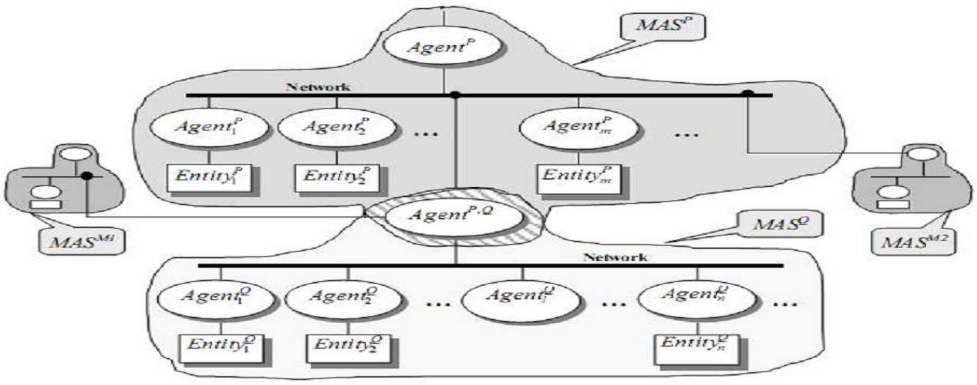
- katram aģentam ir vides modelis (priekšstats par vidi) un uzdotās mērķa funkcijas,
- aģents reaģē uz vides izmaiņām un uz saņemtām ziņām, ņemot vērā tā rīcībā esošo informāciju, vides un citu aģentu iedarbības (1. attēls).



1.att. Aģenta sadarbības modelis.
Fig.1. Interaction model of an agent.

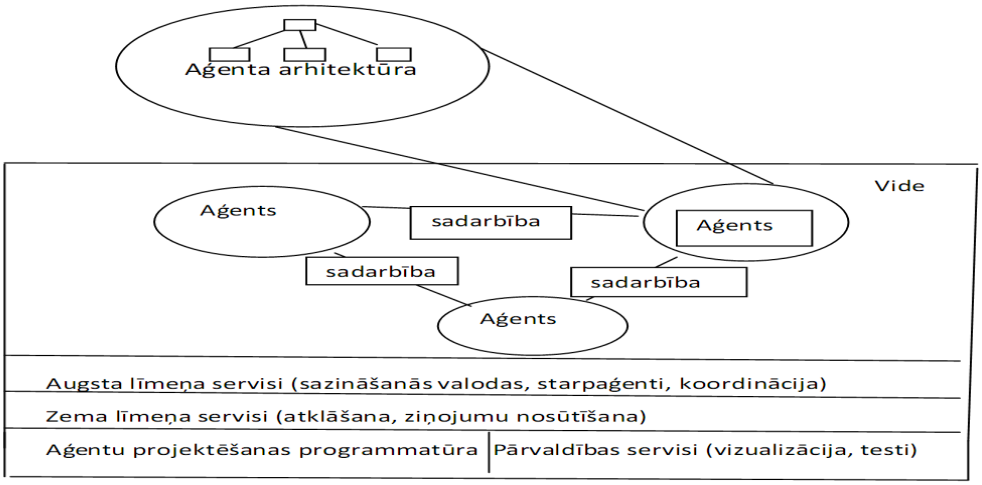
Programmatūras aģenti – autonomas programmatūras sistēmas, kas spēj patstāvīgi pieņemt lēmumus, vadoties no to rīcībā esošām zināšanām par vidi un uzdotajām mērķa funkcijām. Bet atsevišķam aģentam nav ar ko sazināties.

Daudzaģentu sistēmas sastāv no vairākiem aģentiem. Katrs no tiem sadarbojas ar citiem aģentiem un ar vidi, kas to aptver. Aģenti spēj apmainīties ar zināšanām, t. i., izmantojamo informāciju, un sadarboties savā starpā un ar cilvēkiem izmantojot kopējās sazināšanās ontoloģijas, kuru struktūras standartizētas. Aģentu sadarbībai, līdzīgi kā datortīklu arhitektūrā, standartizēti realizācijas un informācijas apmaiņas noteikumi un arhitektūra, balstoties uz FIPA (Foudation for Intelligent Physical Agents) standartiem [3]. Var konstatēt analoģiju: ja datortīklu standartizētā arhitektūra ļauj veikt tīklā iekļauto dažādo ***iekārtu*** sadarbības integrāciju, tad uz aģentu bāzes standartizēto FIPA ontoloģiju arhitektūra ļauj veikt to integrāciju ***datu struktūru*** līmenī, nodrošinot efektīvu sadarbojošos aģentu zināšanu semantisku savietojamību atsevišķa sadarbības domēna ietvaros. Ontoloģiju uzskata par vienu no pilnīgākajām datu konceptualizācijas un specifikācijas struktūrām, kas nodrošina savstarpēju datu interpretāciju. Vairuma aģentu arhitektūra atbilst BDI modelim (BDI – belief-desire-intention - priekšstati, vēlmes, motivācija) [2]. Šis modelis definē datu struktūras un to sadarbību, kas kopā ar vides modeli un lēmumu pieņemšanas procedūrām noteic aģentu uzvedību, kā arī datu apstrādes operācijas, ko jāveic aģentam. Sadalītas daudz aģentu sistēmas var aptvert vairākus datortīklus un atsevišķs to aģents vienlaikus var ietilpt vairāku DAS sastāvā, kas parādīts 2. attēlā [4]. Aģenti var būt mobili un pārvietoties no vienas daudz aģentu sistēmas uz citu, kā arī no viena tīkla uz citu.



2.att. Sadalītas daudz aģentu sistēmas arhitektūra [4]; MAS – daudzaģentu sistēma, entity – lietojums, ko aģents pārvalda.
Fig.2. The architecture of a distributed multi-agent system [4].

Daudzaģentu sistēmas praktiski nav iespējams sākt programmēt bez īpaša aprīkojuma. Šim nolūkam ir laika gaitā ir izstrādātas vairāki šo sistēmu projektēšanas aprīkojumi, piemēram, JADE, Jasson, JACK un citi (3. attēls) [5], kuru pielietošana var nodrošināt izstrādāto daudzaģentu sistēmu atbilstību standartu prasībām. Tie palīdz veikt izstrādājamo daudzaģentu sistēmu atkļūdošanu un verificēšanu.



3.att. Vispārēja aģentu projektēšanas aprīkojuma struktūras shēma [5].
Fig.3. Generic toolkit framework [5].

2. Daudzaģentu sistēmu pielietojumi modelēšanai

Nozīmīgs daudzaģentu sistēmas teorijas virziens, kas tieši nav saistīts ar datortīkliem, ir dažādas dabas objektu mijiedarbības modelēšana un izpēte, kas ir aktuāli, piemēram:

- objektu apdraudējumu novēršanai,
- ekoloģijas, lauksaimniecības, ekonomisko un sociālo procesu izpētei,
- satiksmes plūsmu optimizēšanai lielpilsētās,
- ekonomikas un tirgus pieprasījuma pētījumiem u. c.

Modelēšanas metodēm uz daudzaģentu sistēmu bāzes ir priekšrocības, salīdzinot ar tradicionālām modelēšanas metodēm. Tās ir iespējas izvērtēt mikrolīmeņa komponentu (aģentu) mijiedarbes iespaidu (atklāt, kas notiek sistēmā, ja dažu aģentu uzvedību izmaina) uz sistēmas uzvedību un parametriem makrolīmenī, ja aģentu mijiedarbības procesi norit paralēli. Viens no modelēšanas aprīkojumiem, ko var lietot šim mērķim, ir programmatūra Agentsheets2.6 [6], kas tika aprobēta [7] modelējot Lotka-Volterra konkurences sistēmu. Īpaša uzmanība paralēlu procesu modelēšanai tiek veltīta projektējot datortīklu protokolus, jo tur pat vienkāršu mehānismu paralēlas izpildes mehānismu verifikācija kļūst sarežģīta [8].

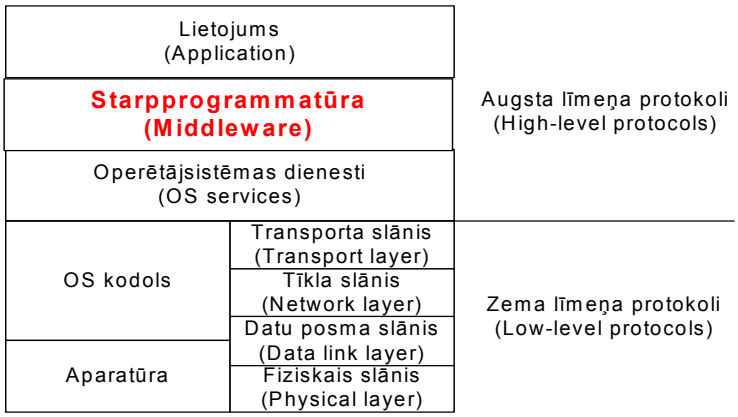
3. Datortīklu starpprogrammatūra un tās lietošana

Datortīklu starpprogrammatūras (middleware) uzdevums ir atšķirīgu programmatūru vai lietojumu sasaiste veiksmīgai kopdarbībai. Datortīklos to bieži noteic CORBA arhitektūras standarti. Starpprogrammatūras var iedalīt dažādos tipos, katrs no kuriem ir der tikai savā konkrētā situācijā. Datortīklu arhitektūrā DAS moduļus parasti uzskata par starpprogrammatūru, kas atrodas starp lietojumprocesiem un tīkla transporta servisa slāņiem (4. att.). Kā galvenie starpprogrammatūras veidi tiek uzskatīti [10]:

1. uz objektiem balstīta starpprogrammatūra (*Object based middleware*);
2. uz notikumiem balstīta starpprogrammatūra (*Event based middleware*);
3. uz ziņojumiem orientēta starpprogrammatūra (M*essage oriented middleware*).

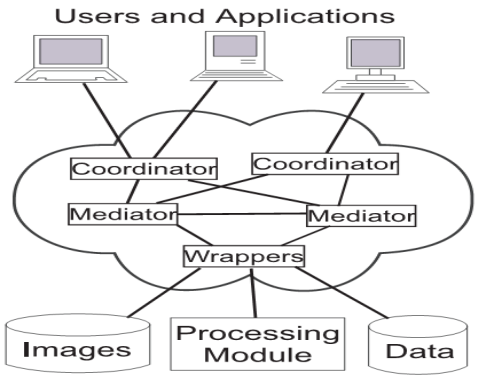
Par starpprogrammatūras funkcijām uzskata: 1) distribūciju, kas var sastāvēt no dažādās vietās esošām komponentēm, paslēpšana, 2) aparatūras, operētājsistēmas un protokolu heterogēnu (atšķirīgo) īpašību

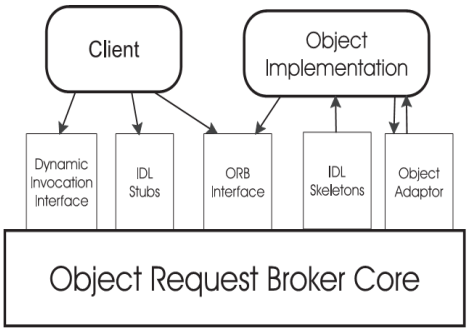
paslēpšana, 3) universālo, augsta līmeņa standartizētu saskarņu nodrošināšanas lietojumiem, lai izstrādātāji tos varētu viegli savienot, atkārtoti izmantot, pārvietot un padarīt sadarbību spējīgus, 4) spēja nodrošināt galveno pakalpojumu vispārējas funkcijas atvieglojot sadarbību starp lietojumiem [10].



4.att. Starpprogrammatūras vieta datortīkla arhitektūrā.
Fig.4. Placement of middleware in the architecture of computer network.

Pēdējo gadu laikā datortīklu apritē ir uzkrājusies visdažādāko veidu elektroniski glabāta informācija: teksti, audio ieraksti, attēli u. c. Tā tiek glabāta dažādos formātos – datnēs, relācijas vai objekt-orientētās datu bāzēs – un var būt organizēta pēc atšķirīgām semantikām. Lai lietotājs varētu bez pūlēm piekļūt šādai informācijai, to nepieciešams integrēt (5. att.) nodrošinot kopējas piekļuves metodes. Tādēļ šāda informācija jānodrošina ar attiecīgiem veidiem atbilstošiem draiveriem jeb iesaiņotājiem (wrapper), lai tā kļūtu pieejama visiem starpnieku moduļiem. Pēdējie izmanto kodētas zināšanas par datu saturu, lai padarītu to pieejamu augstākā līmeņa lietojumiem - starpnieku moduļiem un koordinatoriem. Viens no efektīviem līdzekļiem šādas standartizētas pieejas nodrošināšanai ir standartizētas starpprogrammatūras CORBA (Common Object Request Broker Architecture) [11] izmantošana. Tā, tēlaini runājot, nodrošina īpašu ORB (Object Request Broker) programmatūras ‘’kopni’’(6. att.), kas ļauj klientu lietojumiem caur standartizētas valodas saskarnēm (IDL – Interface Definition Language) piekļūt citiem objektiem pēc vārda, lai arī kurā datortīkla vietā tas neatrastos. Vienīgais, kas vajadzīgs, lai tiem būtu atbilstoši interfeisi un vārdu katalogs.
Taču viens no perspektīvākiem risinājumiem, acīmredzot, ir daudzaģentu sistēmu iekļaušana CORBA standarta arhitektūrā. Bet CORBA lietojama tikai viendabīgu komponentu zemākā līmeņa sadarbībai. Tā var kalpot par pamatu daudzaģentu sistēmu lietošanai.



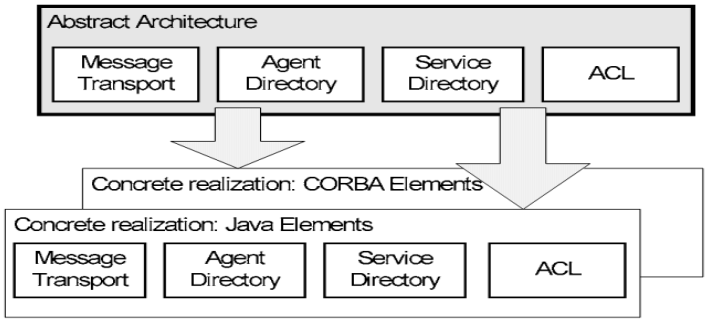


5.att. Dalītās informācijas integrācija [11]
Fig.5. Integration distributed information [11]

6.att. CORBA arhitektūra [11]
Fig.6. The CORBA architecture [11]

Jāņem vērā, ka neatkarīgi no daudzaģentu sistēmām datortīklu starpprogrammatūra attīstās pati par sevi. Tiek lietotas reālā laika un bezvadu CORBA versijas, kā arī tiek izmantota uz ontoloģijām

sakņota tīmekļa valoda (OWL – Web ontology language). Bieži konkrētajās realizācijās abstraktā aģentu arhitektūras galvenie komponenti (ziņojumu transports, aģentu un servisu direktorijas, aģentu sazināšanās valoda (ACL)) tiek realizētas, lietojot CORBA un Java elementus [3] (7. attēls).

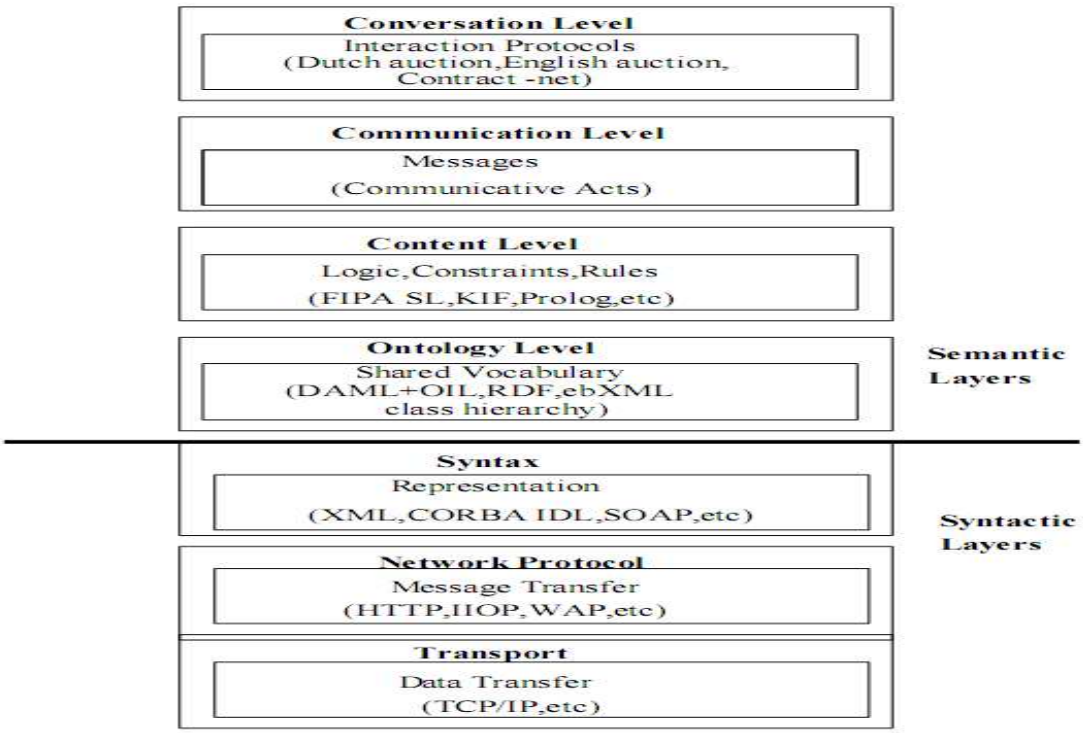


FIPA Abstract Architecture Mapped to Various Concrete Realizations

7.att. Aģenta FIPA abstraktās arhitektūras realizācija konkrētā starpprogrammatūrā [3]
Fig.7. FIPA abstract architecture mapped to concrete middleware [3]

Ņemot vērā augstāk teikto, aģenta programmatūras steku var sadalīt 2 slāņos (8. attēls):

- zemāko – sintaksisko, kas tieši saistīts ar datortīklu un starpprogrammatūru un
- augstāko – semantisko, kas ietver aģenta mērķim atbilstošus apakšslāņu komponentus.

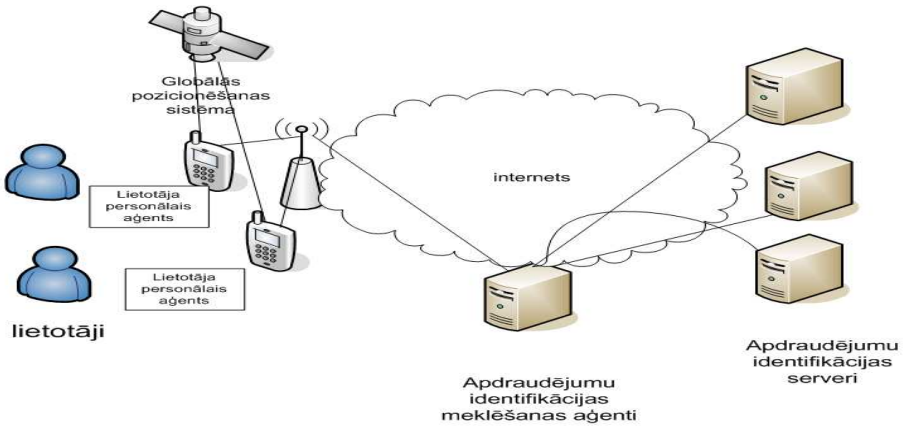


8.att. Aģenta sazināšanās tehnoloģijas steks [5]
Fig.8. Agent communication technology stack [5]

4. Perspektīva apdraudējuma novēršanas sistēmas arhitektūra

Ņemot vērā apdraudējuma novēršanas sistēmu savlaicīguma prasības [12], apdraudējuma novēršanas sistēmu projektēšanas procesā attiecīgā sadarbības arhitektūras izvēle jāizdara izejot no apdraudējumu

novēršanas savlaicīguma, kas svarīgi paaugstināta riska gadījumos. Tas nozīmē, ka tiem apdraudējuma domēniem, kur apdraudējuma novēršanas uzsākšanas savlaicīgums ir ļoti aktuāls un apdraudējuma riski ir būtiski, jau iepriekš jāparedz iespējas nodibināt kontaktus starp klientiem un serveriem un jāsaskaņo servisa parametri, ņemot vērā apdraudējumu objektu (lietotāju) parametru individuālās vērtības. Izejot no šādiem apsvērumiem un no iepriekšējo pētījumu rezultātiem par daudzaģentu sisēmu lietošanu apdraudējumu novēršanas sistēmās [12], var piedāvāt šādas sistēmas arhitektūru, kas parādīta 9. attēlā. Tajā apdraudējuma objekta - lietotāja - sazināšanās ar tīklu notiek caur mobilajās iekārtās esošiem personāliem aģentiem, kuros atrodas informācija par lietotāja iespējām, veselības stāvokli u. c. parametriem, kam var būt iespaids uz apdraudējumu. Iespējama arī šo iekārtu sadarbība ar GPS tīklu, lai saņemtu papildus informāciju par lietotāja atrašanās vietu un no tās izrietošām apdraudējuma iespējām. Šādā sistēmā lietotāju personālie aģenti atrodas tiešsaistes režīmā ar apdraudējumu identifikāciju meklēšanas aģentiem, kas, balstoties uz informāciju, saņemto no lietotāja, meklē tādas lietotāja profilam un individuālām prasībām atbilstošas apdraudējuma novēršanas metodes un resursus serveros, kas spēj dot rekomendācijas vai rosināt veikt pasākumus, lai novērstu apdraudējumu.

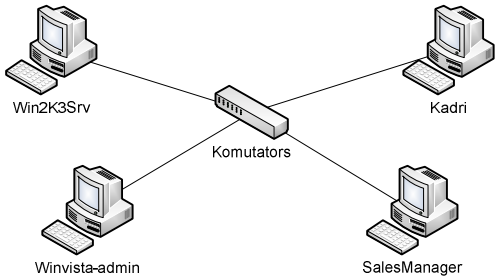


9.att. Perspektīvas draudu novēršanas sistēmas datortīkla arhitektūra
Fig.9. Perspective architecture of threat preventing computer network system

5. BitTorrent vienādranga tīkls datu izplatīšanai

BitTorrent ir viens no populārākiem P2P (peer-to-peer – vienādranga) tīkla protokoliem, kuru lieto datu apmaiņai. Tā izmantošana ļauj samazināt lielo datu izplatītāju izmaksas (piem., jaudīga aparatūra un platjoslas kanāla noma). Kad cilvēks sāk datnes lejupielādi, viņš uzreiz kļūst arī šo datu izplatītājs. Līdz ar to, jo vairāk ir datu lejupielādētāju, jo vairāk ir šo pašu datņu izplatītāju, un jo ātrāk tiek izplatīti dati. Lai lejupielādētu datni, nepieciešama speciāla **torrent** metadatu datne, kurā ir informācija par datiem kuri būs izplatīti - kāda tipa dati, no cik lielam daļām viņi sastāv, kādi ir datu daļas SHA1 kodi un kāda ir izplatītāja adrese, kurā tiek novietoti šie dati.
BitTorrent tīkls sastāv no datoriem, kuros ir uzinstalēta un palaista programma, kura veic datu apmaiņu izmantojot BitTorrent protokolu. Katrs dators ir klients (aģents), kurš angļu valodā saucās **peer** un var gan lejupielādēt, gan augšupielādēt datus. Ja tiek izplatīta datu pakete, tas klients kuram ir pilna (100%) datu pakete saucās izplatītājs (**seeder**).
Datu ielādes procesu vada viens no klientiem, kuru dēvē par trekeri (**tracker**). Trekeris reģistrē sekojošus klientu parametrus: datu paketes daļas, interneta ātrumu, IP adreses un citu informāciju. Trekeris var būt jebkurš klients. Parasti, trekeris ir klients, kurā ir ielādēta informācija no torrent metadatu datnes un kurš pirmais sāk datu izplatīšanu pārējiem. Katrs klients sākot datu lejupielādi, griežas pie trekera pēc klientu adresēm un reģistrē sevi. Datu pārraides procesā katra datnes daļa ir sadalīta vēl sīkākās porcijās (parasti 16 Kb) pirms sūtīšanas caur tīklu. Katras izplatītas datnes porcijas tiek ievietotas rindā. Porcijas no dažām rindām tiek ievietotas konveijerā. Konveijera garumu var

uzstādīt pašam, lai būtu iespējams utilizēt vairāku datu pārraides joslas platumu. Rezultātā iegūstam patstāvīgo savienojumu ar gandrīz nemainīgo ātrumu. Kad ir saņemtas visas porcijas daļas, tiek pārbaudīts daļas SHA1 kods un ja tas sakrīt, klients sāk saņemt vai sūtīt nākamo datnes daļu. Lejupielāde izmantojot BitTorrent protokolu notiek pēc noteiktiem nosacījumiem.
Pirmais galvenais datu pārraides nosacījums ir pilnībā lejupielādēt to datnes daļu, kuru sākām saņemt pirmo. Kā jau bija minēts, mēs sadalām katru daļu porcijās. Tikko mēs sākam saņemt no kāda izplatītāja šīs daļas porcijas, mēs neko citu neņemam – tikai porcijas no šīs daļas. Tas nodrošina visātrāko daļu saņemšanas laiku.
Otrais svarīgākais nosacījums ir daļu izvēles algoritms. Kad sākas lejupielāde, mums nav nevienas datu daļas. Šajā gadījumā saņēmējam nav nevienas datnes, ko viņš varētu augšupielādēt citam klientam. Tad galvenais uzdevums, ir vispirms saņemt kaut kādu datnes daļu. Priekšroka ir tām daļām, kuras ir lielākam klientu skaitam, kas nodrošina visātrāko lejupielādi (jo vairāk izplatītāju, jo ātrāks ātrums). Pēc tam, kad kaut kāda pirmā datnes daļa ir lejupielādēta, ielādes stratēģija mainās uz citu. Nākamās daļas tiek ņemtas pēc nosacījuma, ka lejupielādējamā daļa ir visretākā daļa no tām, ko piedāvā izplatītāji. Šo tehniku sauc par **„pirmais – retākais”**. Šī tehnika labi organizē lejupielādi, lai jebkurā laika momentā, nebūtu tā, ka izplatītājs ar vienīgu palikušu datnes daļiņu aiziet prom un lejupielāde apstājas pie 99,99%. Pēc šiem raksturojumiem, BitTorrent aģentu tīkls ir daudzaģentu sistēma, jo tajā ir liels skaits autonomu aģentu. Katrā datorā, kurā ir palaista programma, kura spēj ielādēt datnes no interneta izmantojot BitTorrent protokolu, ir autonoms aģents. Katrs klients pats pieņem lēmumu no kura klienta kādu datnes daļu ņemt – galvenais ir mērķa sasniegšana, proti, pilnu datu apjomu iegūšana. Aģents pats analizē savienojuma ātrumu, kanāla šaurumu un pats savāc statistiku par to, kuram klientam kādas daļas ir. Visi atslogošanas algoritmi u.c. BitTorrent funkcijas – viss attiecās uz klienta pusi. Operējot ar šiem datiem, klients var lejupielādēt vajadzīgo, retāko datnes daļu, kas arī, savukārt, ir ierakstīts BitTorrent protokola specifikācijā un no tā nevar izvairīties.
Katram BitTorrent protokola aģentam ir sava izpratne un savas zināšanas par vidi (tīklu, kurā viņi operē). Viņiem ir tikai daļēja vides uztvere. Viņi nezin visu klientu adreses, ko viņi lejupielādē utt. Viņi var uzzināt kopējo klientu skaitu, bet neko vairāk. Līdz ar to BitTorrent klients arī pieder pie aģentiem daudzaģentu sistēmā. Darba lielākā daļa tiek izdarīta tieši klientu pusē - BitTorrent klientu programmā. Trekeris ir nepieciešams: lai uzzinātu tekošo klientu skaitu un viņu IP adreses, lai varētu uzlikt šo .torrent datni un lai varētu sākt lejupielādi. Jebkurš klients var kalpot par trekeri.
BitTorrent protokola izmantošanas plusus un mīnusus tika izpētīta 4 veiktajos eksperimentos. Tika pieņemts, ka ir tīkls ar 4 datoriem un vienu komutatoru un tīkla topoloģija ir zvaigzne (10. attēls).



10.att. Eksperimentālā BitTorrent tīkla topoloģija
Fig.10. Topology of experimental BitTorrent network

Pirmajos 2 testos, datu ielāde notiek izmantojot BitTorrent protokolu un pēdējos 2 - izmantojot FTP protokolu. Katrā testā, izplatītāja loma bija datoram ar NetBIOS vārdu Win2K3Srv. Pirmajos divos testos izplatītāja datorā bija tāds pats BitTorrent aģents, pārējos testos FTP serveris. Pārējie datori bija datu saņēmēja lomās.
Pēc katra eksperimenta, datori un komutators tika pārstartēti, lai izvairītos no situācijas, kad aparatūras dēļ parādās kļūdas tīkla pārraidē, aizkaves vai tiek zaudētas tīkla paketes. Testa ietvaros datori kalpoja kā datu izplatītāji, vai kā datu saņēmēji. Aparatūras parametri, kā arī datoros uzinstalēta programmatūra, nav svarīga, jo visos testos tiek izmantotas tādas pašas darba stacijas un eksperimenta apstākļi ir vienādi.

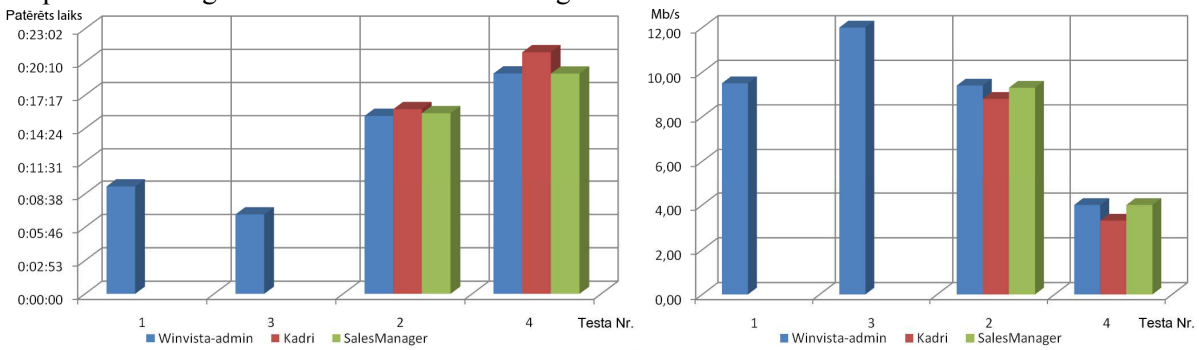
Pēc veiktiem eksperimentiem, tika iegūti sekojoši rezultāti:

1.tabula
Table 1.

Eksperimentu rezultāti
Experimental results

Testa Nr.	Saņēmēj dators	Pārraides protokols	Sākums	Beigas	Patērēts laiks	Maks. ātrums	Vid. ātrums
1	Winvista-admin	BitTorrent	20:05:27	20:14:44	00:09:17	9,5 Mb/s	8,1 Mb/s
2	Winvista-admin		20:24:47	20:40:16	00:15:29	9,4 Mb/s	4,9 Mb/s
	Kadri		20:24:49	20:40:53	00:16:04	8,8 Mb/s	4,6 Mb/s
	SalesManager		20:24:51	20:40:34	00:15:43	9,3 Mb/s	4,8 Mb/s
3	Winvista-admin	FTP	20:43:04	20:49:56	**00:06:52**	12 Mb/s	**11,3 Mb/s**
4	Winvista-admin		20:51:41	21:10:51	00:19:10	4,0 Mb/s	4,0 Mb/s
	Kadri		20:51:44	21:12:44	00:21:00	3,3 Mb/s	3,3 Mb/s
	SalesManager		20:51:49	21:10:48	00:19:09	4,0 Mb/s	4,0 Mb/s

Eksperimentos iegūtos rezultātus varam attēlot grafiski:



11.att. Eksperimentu rezultāti
Fig.11. Results of eksperiments

Kā ir redzams no testa rezultātiem, viena klienta gadījumā, visātrākais veids, kā var iegūt datni, no apskatītiem lejupielādes veidiem ir datu saņemšana caur FTP. Testā Nr. 3, kad ir tikai viens klients, kurš lejupielādēja datni caur FTP protokolu, tika iegūts maksimāli iespējamais pārraides ātrums. Šis ātrums bija tuvs lielākajam iespējamam ātrumam 100 Mbit/s eksperimenta tīklā, kas ir 12,5 Mb/s un kas ir praktiski nesasniedzams tīklā ar vairākiem datoriem.
Vairāku klientu gadījumā, lejupielāde izmantojot BitTorrent notiek ātrāk. Kaut gan daļa no resursiem tiek izmantota (centrāla procesora slodze ir lielāka, nekā FTP gadījumā), BitTorrent atļauj saņemt datus no dažiem avotiem vienlaicīgi, kas paātrina lejupielādi. Ja lokālajā tīklā starpība bija 3 minūtes, tad cik liela starpība varētu būt lejupielādējot caur internetu? Vienā lokāla tīkla ietvaros, laiku starpība lielāka par 30 sekundēm jau ir ļoti liela, jo ātrums ir ļoti liels un praktiski, dažreiz ir ierobežots vājas darba stacijas dēļ. Laiku starpība lielāka par 3 minūtēm viena lokāla tīkla ietvaros ir daudz - 3 minūtes ar ātrumu 12 Mb/s ir 2 Gb.

Secinājumi

- Daudzaģentu sistēmu pielietošanu datortīklu starpprogrammatūrā var uzskatīt par vienu no to perspektīvākajiem attīstības virzieniem, kas kalpos datortīklu pastāvīgas klātbūtnes (ubiquity) attīstībai;
- Visbiežāk standartizētas aģentu tehnoloģijas tiek iekļautas standartizētā starpprogrammatūras arhitektūrā, kas atvieglo to realizāciju;

- Daudzaģentu sistēmu pielietošanu var uzskatīt par vienu no perspektīvākajām paralēlu procesu modelēšanas metodēm, jo šāda pieeja ļauj izpētīt modelējamās sistēmas komponentu mikrolīmeņa iespaidu uz šīs sistēmas uzvedību makrolīmenī;
- Daudz aģentu sistēmu lietošana mobilajās iekārtās dod iespējas padarīt datortīklu pakalpojumus arvien plašāk pieejamus un ļauj integrēt mobilās iekārtas (piem., viedtelefonus) interneta struktūrā;
- Daudzaģentu sistēmas izmantošana dod vairāku objektu mijiedarbības priekšrocības (piem., BitTorrent gadījumā, vienlaicīga datu ielāde no dažādiem avotiem palielina lejupielādes ātrumu), palielina bojājumpiecietību (piem., ja viens aģents iziet no aprindas, viņu var aizvietots kāds cits aģents) un citas priekšrocības.

Literatūra

1. Ferber J. Multi-Agent Systems. – Harlow, England: Addison-Wesley Longman. 1999. - 508 p.
2. Woodridge M. An Introduction to Multiagent Systems.–John Wiley & Sons, Ltd – 2006 – 348 p.
3. Standard Status Specification [Elektroniskais resurss] – FIPA Specifications, 2005. http://www.fipa.org/repository/standardspecs.html - Resurss aprakstīts 2008. g. 18.okt.
4. A CORBA-Based Multi-Agent System Integration Framework/ Cheng Tao, Guan Zailin, Liu Liming, Wu Bo, Yang Shuzi. - Proceedings. Ninth IEEE International Conference on Engineering Complex Computer Systems, (ICECCS'04), 14-16 April 2004 pp.191-198, [Elektroniskais resurss] http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=01310917 - Resurss aprakstīts 2008. g. 18.okt.
5. Agent technology. Chapter 4. [Elektroniskais resurss] - Resurss aprakstīts 2008. g. 18.okt. http://www.srdc.metu.edu.tr/webpage/courses/ceng520/lecture_notes/AgentChapter.pdf
6. Luck M., Ashri R., d'Inverno M. Agent-Based Software Development. – Boston, London: Artech House inc.- 2004, - 208 p.
7. Agentsheets2.6 programmatūras aprīkojums dalītu sistēmu modelēšanai [Elektroniskais resurss] http://www.agentsheets.com/products/trial/index.html - Resurss aprakstīts 2008. g. 18.okt.
8. Ermuiza A. Using multi-agent simulation tools for modeling distributed systems// Proceedings of the International Scientific Conference "Applied Information and Communication Technologies", Jelgava, Latvia, April 10-12, (2008), - pp. 29 -33.
9. Fokkink W. Modelling Distributed Systems. – Berlin, Heidelberg, New York: Springer-Verlag, 2007- 151 p.
10. Krakowiak S. What is Middleware? Object Web. Open Source Middleware. -2003 [Elektroniskais resurss] http://middleware.objectweb.org/ - Resurss aprakstīts 2008. g. 18.okt.
11. Purvis M., Cranefield S., Ward R. Distributed Software Systems: From Objects to Agents [Elektroniskais resurss] //Software Engineering: Education & Practice, Proceedings. International Conference.Volume , Issue , 26-29 Jan (1998) – pp. 158 – 165- Resurss aprakstīts 2008. g. 18.okt.
12. Ermuiža A**.** Draudu novēršanas sistēmu modeļa interpretācija daudz aģentu sistēmu ontoloģijā// Rīgas Tehniskās universitātes zinātniskie raksti*,* 5. sēr., Datorzinātne, Datorvadības tehnoloģijas. - 24. sēj. (2005), 77.-83. lpp.
13. BitTorrent is a Multiagent System | Multiagent Systems. José M. Vidal. 2007. [Elektroniskais resurss] http://www.multiagent.com/bittorrentismultiagent - Resurss aprakstīts 2008. g. 18.okt.
14. Ritu Chadha; Sze-Ying Wuu Managing distributed systems using OSI systems management Systems Management, 1996., Proceedings of IEEE Second International Workshop on Volume , Issue , 19-21 Jun 1996[Elektroniskais resurss] - Resurss aprakstīts 2008. g. 18.okt. pp. 117 – 126 http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=00534154

Ermuiža A., Iščenko J. Daudzaģentu sistēmu pielietojumi datortīklu starpprogrammatūrā

Daudzaģentu sistēmu teoriju uzskata par vienu no galvenajiem datortīklu attīstības un to visursesmes dzinuļiem. Tā balstās uz realizējama pieņēmuma, ka speciāli programmatūras moduļi – aģenti – spēj sazināties un pieņemt izsvērtus lēmumus, izejot no to rīcībā esošas informācijas. Daudzaģentu sistēmu realizācija tiek veikta ar

speciālu aprīkojumu palīdzību, kas noteic to arhitektūru un sazināšanās procesu standartus. Daudzaģentu sistēmu pielietojumi izplatīti gan dalīto sistēmu modelēšanā, gan to lietošanā datortīklu programmatūrā. Uz tiem sakņotām modelēšanas metodēm ir tāda priekšrocība salīdzinot ar tradicionālām metodēm, ka var atklāt sakarības starp sistēmu komponentu mikrosadarbībām un to makrouzvedības raksturojumiem. Modelēšanu bieži apvieno ar projektēšanu. Datortīklu lietojumos daudzaģentu sistēmas pilda galvenokārt starpprogrammatūras lomu būtiski paplašinot plaši lietojamās CORBA arhitektūras iespējas. Īpaši svarīga loma tām ir ieviešot elastīgu starpprogrammatūru mobilajās iekārtās, kas pēdējā laikā kļūst par datortīklu būtisku sastāvdaļu. Rakstā īsi aplūkoti tādi daudzaģentu sistēmu lietojumu piemēri starpprogrammatūrā kā perspektīva draudu novēršanas sistēmas arhitektūra un BitTorrent datu izplatīšanas sistēma (tiek analizēti tās modelēšanas rezultāti).

Ermuiza A., Ishchenko E. Applications of multi-agent systems in the middleware of computer networks
The theory of multi-agent systems is considered as one of main drivers of computer networks and their ubiquity development. It is based on the implementable presumption that an agent as special software module is possible to communicate with other agents and to make reasonable decisions based on his accessible information. The implementation of multi-agent systems is realized by means of special tools which establish their architecture and standards of their interaction and communication processes. The applications of multi-agent systems are widespread both in modeling distributed systems, and as applications in computer network middleware. The preference of multi-agent systems over traditional modeling methods is that those allow taking into account the impact of individual behaviours of system components in micro level upon the behaviour and characteristics of whole system in macro level. It is attempts modeling often tending to combine with system development in computer networks. In the applications of computer network multi-agent systems often play a part of middleware which essential expand possibilities of traditional middleware, for example CORBA. This consideration is especially essential for development flexible middleware in widespread mobile devices which are getting as the essential part of computer networks. It is shortly considered such examples of multi-agent applications as architecture of perspective thread preventing system and BitTorrent data distribution system. There are considered results of modeling for the last one.

Эрмуйжа А., Ищенко Е. Применение мульти-агентных систем в промежуточном програмном обеспечении вычислительных сетей
Теория мульти-агентных систем рассматривается как один из главных двигателей развития и распространения применения вычислительных сетей. Она опирается на реализуемом предположении о том, что специальные програмные модули – агенты – способны общаться и принимать обоснованные решения, исходя из той информации, которой они обладают. Реализация мульти-агентных систем осуществляется с помощью специальных програмных обеспечений, которые определяют отвечающую стандартам архитектуру и средства общения. Применение мульти-агентных систем распространены как для моделирования различных распределенных систем, так и для вычислительных сетей в качестве промежуточного програмного обеспечения, тем самым расширяя возможности CORBA. Особо важную роль онт играют при внедрении гибких промежуточных програмных обеспечений в мобильных устройствах, которые в последнее время становятся существенной частью вычислительных сетей. В статье коротко рассмотрены такие сетевые применения мульти-агентных систем как перспективные системы предотвращения угроз и системы распространения информации с помощью протокола BitTorrent. Для последнего применения приведены результаты моделирования.