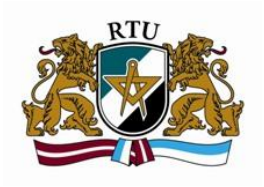


IEGULDĪJUMS TAVĀ NĀKOTNĒ

Projekts „Daudzaģentu robotizētas intelektuālas sistēmas tehnoloģijas izstrāde”
Vienošanās Nr. 2010/0258/2DP/2.1.1.1.0/10/APIA/VIAA/005
PVS ID 1528

Pētījuma apraksts daudzaģentu sistēmu jomā

Saturs

Ievads

Dokumenta mērķis. Dokumenta mērķis ir aprakstīt projekta ietvaros veikto pētījumu par daudzaģentu sistēmu pielietošanu daudzrobotu vadības sistēmas izstrādē. Tas apraksta veikto pētījumu, kurā galvenais uzsvars ir uz sistēmas arhitektūras izvēli.

Darbības sfēra. Dokuments paredzēts projekta dalībnieku lietošanai turpmākajos pētījumos, kā arī jebkuram pētniekam, kas strādā šajā virzienā un var izmantot pētījuma rezultātus savā darbībā. Dokuments ir izstrādāts, izmantojot ERAF finansējumu un līdz ar to ir pieejams ikvienam interesentam.

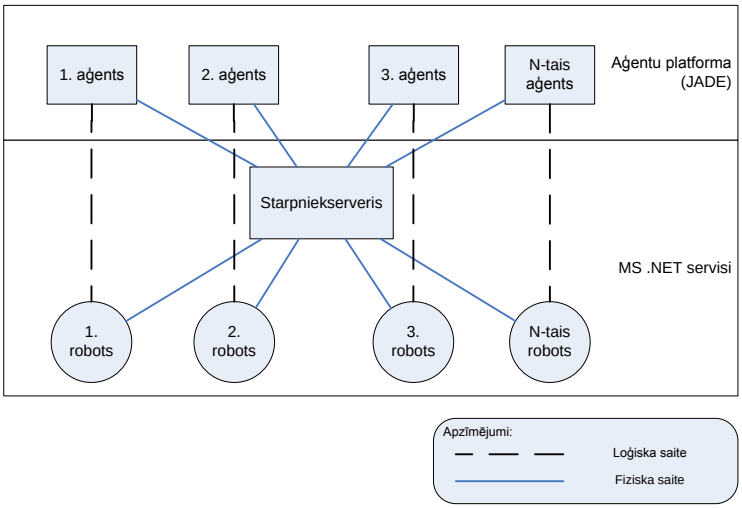
Anotācija. Projekts ietver divas pēc būtības neatkarīgas tehnoloģijas – robotus un aģentus. Aģentu izstrādei mūsdienās izmanto specializētas platformas (JADE, JADEX, JACK, u.c. [1], [2], [3]), kurās ir realizēts aģenta koncepts un realizējot konkrētu sistēmu, nav jādomā par tādām lietā kā aģentu mijiedarbības realizēšana, kas vienkāršo aģentu izstrādi. Visas zināmās aģentu izstrādes platformas balstās valodā Java. Savukārt, robotu vadībai (vismaz STPK) izmanto Microsoft .NET platformu un servisorientētu arhitektūru. Līdz ar to, aģentus un robotus jāizstrādā, izmantojot divas dažādas tehnoloģijas, kuras nepieciešams integrēt. Dokumentā aprakstītā pētījuma uzdevums ir: izstrādāt sistēmas arhitektūru, kas ļautu sistēmā sekmīgi integrēt minētās divas tehnoloģijas, kā arī ļautu dažādās sistēmas daļas (robotu fiziskās vadības apakšsistēma un daudzaģentu sistēma) izstrādāt atsevišķi un arī aizstāt ar citām sistēmām, kam ir tādas pašas saskarnes. Pēdējai prasībai mērķis ir ļaut robotus aizstāt ar imitatoru vai citiem robotiem. Tāpat vadības sistēmu jāvar aizstāt ar vienkāršu lietotāja saskarni.

Pētījumu gaitā ir izstrādātas dažādas kopējās sistēmas un daudzaģentu sistēmas arhitektūras alternatīvas un definēts, kāda veida robotu vadības sistēmas risinājumam katra no tām ir pielietojama. Visbeidzot izvēlēta viena arhitektūra aģentos sakņotai putekļu sūcēju robotu vadības sistēmas izstrādei.

Dokumenta saturs. Pētījuma apraksta pirmajā nodaļā ir aprakstīti analizētie arhitektūras varianti, izvēlētais arhitektūras risinājums un izstrādātais daudzrobotu sistēmas imitators, kas kalpo kā izvēlētās arhitektūras pielietojuma piemērs. Otrajā nodaļā ir doti pētījuma secinājumi, ietverot arhitektūras pielietojuma aprakstu, kā arī tās aprobāciju.

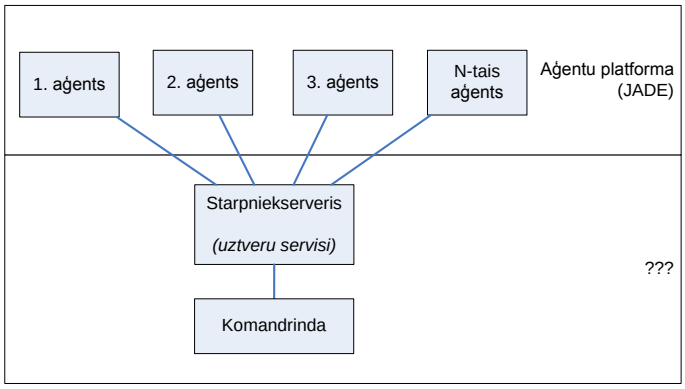
1. Daudzaģentu pētījuma izklāsts

Sistēmas kopējā arhitektūra sastāv no divām galvenajām daļām, kuras jāizstrādā ar dažādām tehnoloģijām (skat 1.1. att.). Par piemērotākajām aģentu izstrādes platformām ir atzītas JADE un JADEX. Loģiskā līmenī katrs aģents vada savu robotu un pārstāv šo robotu daudzaģentu sistēmā (1. aģents vada 1. robotu, 2. aģents vada 2. robotu, utt.). Tomēr tiešas mijiedarbības realizācija katru aģentu un robotu izveido ļoti ciešu sasaisti starp robotu līmeni un aģentu līmeni, kas nav vēlams, jo sarežģī izmaiņu veikšanu sistēmā. Tādēļ sistēmas arhitektūrā ir ieviests starpniekserveris, kas realizē fizisku mijiedarbību starp aģentiem un robotiem. Šī servera uzdevums ir nodot aģentu rīkojumus robotiem un pārsūtīt no robotiem iegūto informāciju atbilstošajam aģentam. Jāpiezīmē, ka katrs aģents var apmainīties ar informāciju ar katru citu aģentu. Šīs saites 1.1. un tālākajos attēlos nav parādītas, lai tie nekļūtu nesalasāmi.

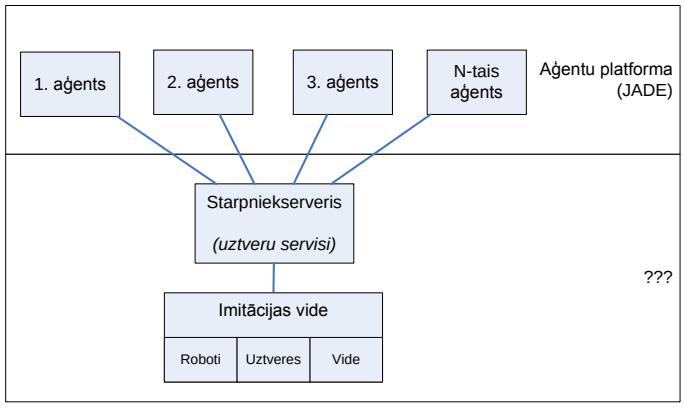


1.1.att. Kopējā sistēmas arhitektūra

Šāda sistēmas arhitektūra ļauj 1.1. attēlā norādītās daļas izstrādāt atsevišķi un arī testēt atsevišķi. Daudzaģentu sistēmas testēšanai starpniekserveri un robotus var aizstāt ar kādu vidi, kurā tiek imitēti roboti vienkāršākā gadījumā var tikt izveidots starpniekserveris ar vienu servisu, kurš saņem aģentu dotās komandas un izvada tās komandrindā. Iespējams, ka šādā gadījumā uz šī starpniekservera ir jāievieto servisi, kas imitē robota uztveres. Šāda pieeja ilustrēta 1.2. attēlā. Šāda vide izmantojama projekta sākumā, tālākā perspektīvā ir iespējams izstrādāt robotu darbības imitācijas vidi, kurā roboti pārvietotos pa virtuālu vidi (skat. 1.3. att.). Šādā gadījumā roboti, vide un robotu uztveres ir realizētas programmatūras veidā. Šādas vides izstrāde ļauj izmēģināt ļoti dažādus aģentu arhitektūras, neveicot darbietilpīgus un dārgus eksperimentus ar reāliem robotiem.

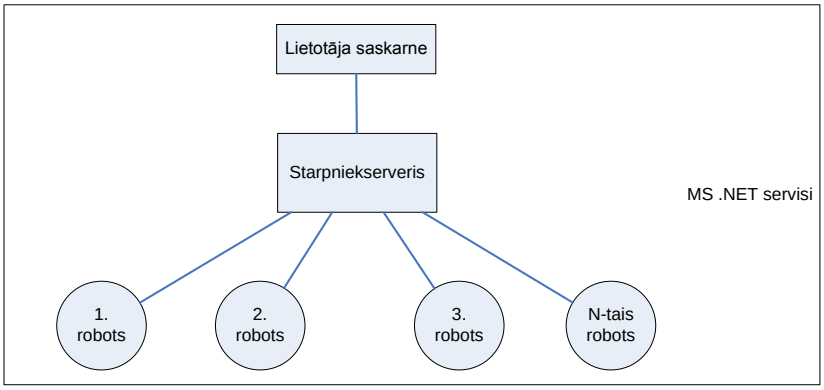


1.2. att. Aģentu testa vides arhitektūra, kas izmanto komandrindu



1.3. att. Aģentu testa vides arhitektūra, kas izmanto vides imitāciju

Līdzīgā veidā arhitektūra ļauj arī izstrādāt robotu vadību neatkarīgi no aģentiem. Šādā gadījumā daudzaģentu sistēmas vietā tiek izveidota lietotāja saskarne, kurā aģentu vietā robotus vada lietotājs. Lietotājs šajā saskarnē ievada komandas un saņem robotu uztveres tādā formā, kādā tās tiktu nodotas aģentiem. Šāda arhitektūra dota 1.4. attēlā.



1.4. att. Robotu testa vide

1.1. Arhitektūras risinājumu analīze

Iepriekš aprakstītā arhitektūra var tikt realizēta ļoti dažādos veidos, saglabājot ideju, ka aģenti ir neatkarīgi izstrādājami no robotiem. Galvenā ideja, kas ir jāsaglabā, ir tas, ka katram robotam atbilst savs aģents un aģenti ar robotiem mijiedarbojas caur starpniekserveri. Līdz ar to šīs arhitektūras ietvaros var realizēt ļoti dažādas aģentu mijiedarbības. Pētījumu ietvaros ir sīkāk analizētas iepriekš aprakstītās vispārīgās arhitektūras realizācijas iespējas. Galīgā sistēmas arhitektūra ir izstrādāta, izveidojot vairāku arhitektūras variantus, izanalizējot tos un izvēloties konkrētās sistēmas izstrādei piemērotāko. Šī sadaļa apraksta dažādās realizācijas iespējas.

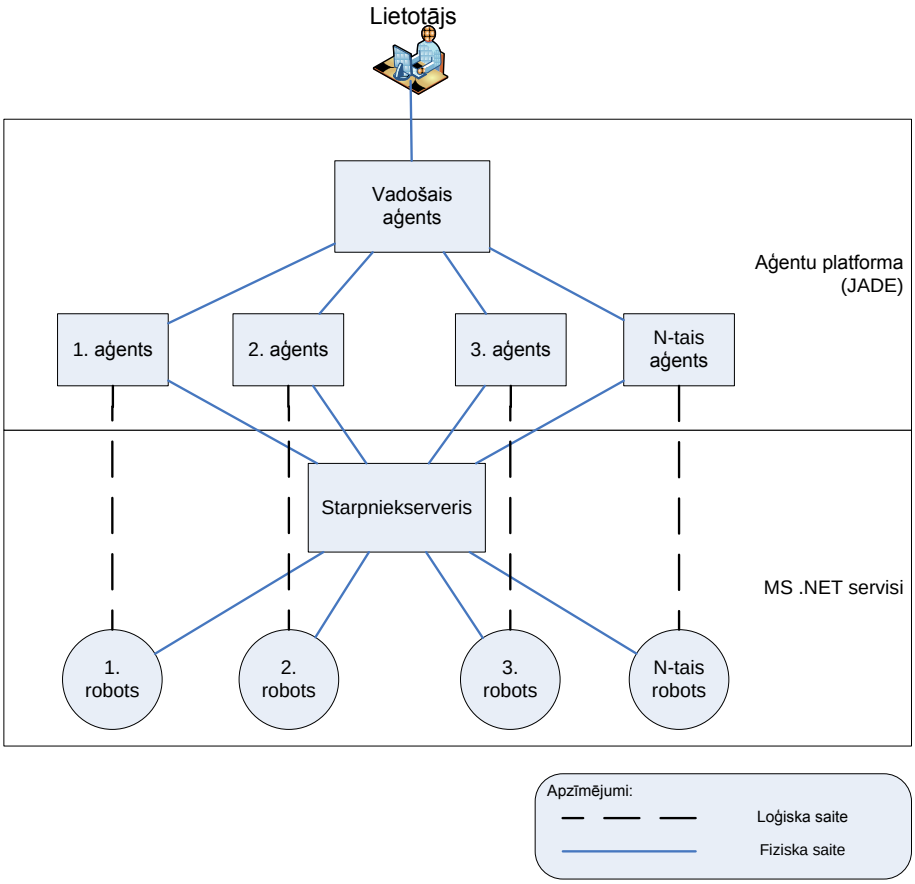
Par pamatā realizējamo variantu ir izvēlēta ideju, ka 1.1. attēlā dotajai arhitektūrai ir pievienots viens vadošais aģents, kurš saņem uzdevumus no lietotāja, tālāk sadala šo uzdevumu un piešķir robotiem atbilstošajiem aģentiem, izmantojot Contract Net protokolu. Šādas sistēmas arhitektūra dota 1.5. attēlā. Šai pieejai atbilst pirmie divi no šajā nodaļā aprakstītajiem aģentu daudzaģentu sistēmu arhitektūras veidiem. Kā sākotnējais realizācijas variants ir izvēlēts tieši 1.5. attēlā redzamais risinājums, jo:

- Tas ir pietiekami intelektuāls un decentralizēts un līdz ar to risinājums nav pārāk triviāls;
- Tas ir izmantojams kā pirmais prototips, kas ir izmantojams tālākajos eksperimentos;
- Šo risinājumu var visai vienkārši pēc tam modificēt, lai ieviestu dažādus citus (sarežģītākus un vairāk decentralizētus mehānismus).

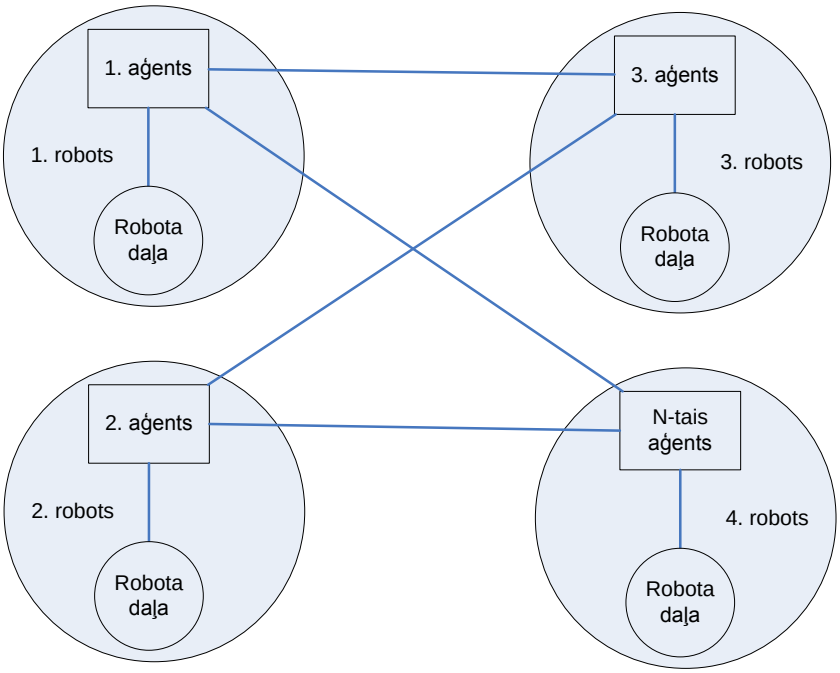
1.1.1. Autonomu robotu ideja

Analizējot iepriekš piedāvāto arhitektūru, ir secināts, ka robotus vada viens centralizēts serveris un līdz ar to tos never uzskatīt par pilnībā autonomiem. Absolūti autonomus robotus var iegūt tikai tādā veidā, ja robotam atbilstošais aģents ir iebūvēts robotā (t.i., robots vadā aģentu sev līdz). Analizējot šo pieeju, ir secināts, ka perspektīvā šis ir ideālais risinājums, uz ko ir jātiecas (skat. 2.2. attēlu). Tomēr tā tūlītēja ieviešana ir sarežģīta, jo:

- Aģenti ir jāizvieto mobilajās iekārtās ar ko ir sarežģīti eksperimentēt un kam tīri praktisku un finansiālu apsvērumu dēļ ir ierobežotas skaitļošanas jaudas;
- Nepieciešami fiziski komunikāciju kanāli starp robotiem;
- Nav skaidrs, kā un vai vispār var veikt eksperimentus, izmantojot 1.2. un 1.3. attēlos dotos imitācijas mehānismus.



1.5. att. Arhitektūra ar vadošo aģentu



1.6. att. Autonomu robotu ideja. Attēlā dota sistēma bez vadošā aģenta. Sistēma ar vadošo aģentu atšķirsies ar vienu elementu: serveri, uz kura būs šis vadošais aģents.

Ja mēs aplūkojam 1.5. attēlu un salīdzinām ar 1.6. attēlā doto autonomo aģentu shēmu, tad var secināt, ka loģiskā līmenī aģentu mijiedarbība nav mainījusies:

- Katrs aģents pārstāv tam atbilstošo robotu;
- Aģenti mijiedarbojas savā starpā;
- Ir iespējams realizēt arhitektūru ar vienu vadošo aģentu. Autonomu robotu gadījumā šis aģents atradīsies uz servera un mijiedarbosies ar visiem robotu aģentiem.

1.1.2. Daudzaģentu sistēmas realizācijas iespējas

Šajā apakšnodaļā aplūkotas dažādas iespējas, kā realizēt pašu daudzaģentu sistēmu, kas atbilst 1.1. attēla augšējai daļai. Iespējamās daudzaģentu sistēmas arhitektūras:

1. Centralizēta plānošana. Viens vadošais aģents, kas zina visu robotu aģentu spējas un sadala uzdevumus starp tiem.
2. Daļēji izkliedēta sistēma ar vienu vadošo aģentu.
3. Pilnībā izkliedēta sistēma. Katram robotam atbilst savs aģents un tie arī ir vienīgie aģenti. Nav vadošā aģenta.
4. Izkliedēta sistēma ar kopīgu informācijas[1] krātuvi jeb „tāfeli". Nav vadošā aģenta, bet ir visiem aģentiem pieejama informācijas krātuve.

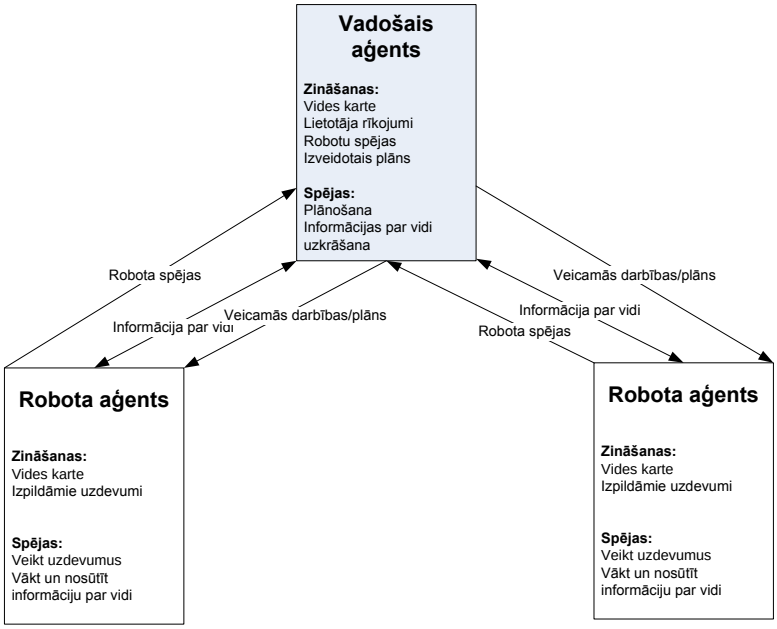
Centralizēta plānošana

Ir viens vadošais aģents, kurš saņem lietotāja uzdevumus un ir atbildīgs par to, lai tie tiktu izpildīti. Vadošajam aģentam ir absolūta kontrole pār robotu aģentiem. Saņemot kādu uzdevumu vadošais aģents veic plānošanu un nosūta katram robota aģentam tā veicamo uzdevumu. Piemēram, robota aģents saņem rīkojumu, ka tam ir jāiztīra 2 konkrētas istabas. Realizācija var atšķirties ar to, ka roboti no vadošā aģenta var saņemt darbību plānu vai tikai uzdevumu. Ja ir saņemts darbību plāns, tad roboti tikai izpilda tiem nodotās darbības un spriešana vai papildus komunikācija ar vadošo aģentu notiek tikai neparedzētos apstākļos. Šim risinājumam lielākais trūkums ir tāds, ka vadošajam aģentam ir jāzina visa informācija par uzdevumiem, piemēram, precīzs priekšmetu izvietojums telpā. Līdz ar to variants uzskatāms par mazperspektīvu. Ja aģents saņem tikai uzdevumu, tādā gadījumā tam jāspēj veikt plānošana, lai noteiktu, kādas darbības tam jāveic.

Centralizētas plānošanas gadījumā visiem robotu aģentiem ir jāpaziņo savas spējas vadošajam aģentam. Robotu aģenti savā starpā nesadarbojas. Sarežģītākais jautājums ir, kā aģenti apmainās ar informāciju par vidi? Viena no iespējamām realizācijām ir tāda, ka roboti nodod visu informāciju vadošajam aģentam, kurš to nodod tālāk pārējiem robotiem, kam tā ir nepieciešama.

[1] Šis atbilst arī terminam zināšanu krātuve, jo tajā ievietotais atbilst aģentu zināšanām.

Pilnīga vadības nodošana vadošajam aģentam rada jautājumu, vai roboti vispār ir aģenti? Tiem nav autonomijas, tie nespēj sadarboties ar citiem aģentiem un veic visai minimālu spriešanu, kā arī neuztur gandrīz nekādas zināšanas. Šis ir vienkāršākais risinājums no realizācijas viedokļa, bet arī vismazāk intelektuālais. Aģentu praktiskai realizēšanai var izmantot kādu vidi, kas koncentrējas uz aģentu mijiedarbību, nevis uz spriešanu aģentos. Iespējams, ka piemērotākā vide ir JADE ar savu paplašinājumu LEAP, kas paredzēts mobilām iekārtām. Centralizētai plānošanai atbilstoša arhitektūra dota 1.7. attēlā. Šajā un turpmākajos attēlos uzskatāmības nolūkā nav iekļautas saites, kas saistītas ar dažādām izņēmuma situācijām, piemēram, informācija, ko aģents nosūta, ja uzdevuma izpilde nav beigusies sekmīgi.



1.7. att. Centralizētas sistēmas arhitektūra

Daļēji izkliedēta sistēma ar vienu vadošo aģentu

Līdzīgi kā centralizētā plānošanā sistēma sastāv no robotu aģentiem un viena vadošā aģenta. Galvenā atšķirība ir tāda, ka vadošajam aģentam nav absolūtas kontroles pār robotu aģentiem. Tāpat kā iepriekš lietotājs uzdevumu piešķir vadošajam aģentam, kuram ir jānodrošina tā izpilde. Šajā gadījumā vadošais aģents zina sistēmā esošos robotus, taču tam nav nepieciešams zināt to spējas. Uzdevumu deleģēšana robotiem notiek, mijiedarbojoties ar to aģentiem. Iespējamie veidi, kā var deleģēt uzdevumus robotiem:

- Contract Net protokols [4], [5] , kas ir labi zināms uzdevumu sadales protokols.
- Dažāda veida izsoles, kurās tiek atrasts robots, kas var visefektīvāk (piemēram, ar vismazākajām izmaksām) veikt nepieciešamo uzdevumu. Šādai pieejai ir jēga, ja ir dažādi roboti, kam ir dažādas spējas un līdz ar to ir nepieciešams atrast piemērotāko. Iespējams šādi var meklēt arī vismazāk noslogoto.

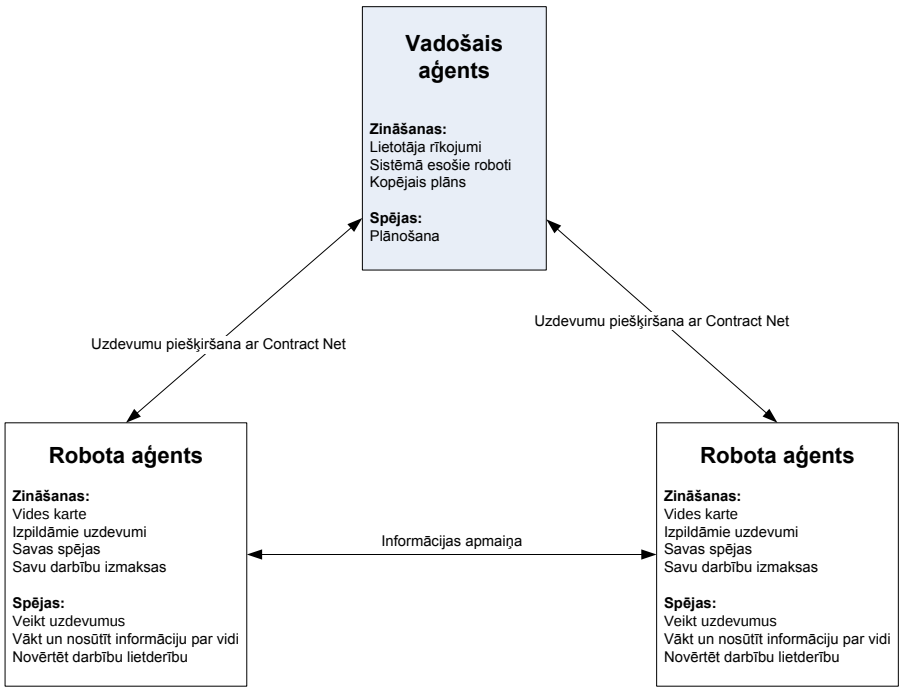
- Dažādi citi uzdevumu piešķiršanas mehānismi [6],[7], [8], [9], [10], [11], [12].

Robotu aģentiem, kas piedalās izsolē vai Contract Net protokolā ir jāspēj spriest par savām iespējām paveikt konkrēto uzdevumu. Līdz ar to roboti kļūst par spriest spējīgiem aģentiem. Vēl jo vairāk, tie ir lietderībā sakņoti aģenti, kas spēj novērtēt, vai piedalīšanās šajā izsolē/protokolā ir lietderīga un kādu solījumu ir jāveic. Līdz ar to šajos aģentos ir jāiebūvē mehānismi, kas ļauj novērtēt savu darbību izmaksas un lietderību.

Šajā realizācijā robotu aģenti vairs nav tikai uzdevumu izpildītāji, bet ir spriest spējīgi, līdz ar to tiem ir nepieciešamas atbilstošas aģentu realizācijas. Par piemērotu ir atzīta BDI (*Belief, Desire, Intension*) aģentu koncepcija, kuru iespējams realizēt JADEX platformā, kas balstās uz JADE.

Izsoles gaitā roboti ir izteikti konkurējoši, kas cenšas darbu paveikt ar mazākām izmaksām, kā citi. Taču, apmainoties ar informāciju, robotiem ir jāsadarbojas, lai kopējā sistēma strādātu efektīvi. Šajā gadījumā var vai nu ieviest tādu pašu centralizētu informācijas apmaiņu, kā centralizētas plānošanas gadījumā, vai arī ir iespējams realizēt tiešu informācijas apmaiņu starp robotiem. Taču tādā gadījumā robotiem jāspēj mijiedarboties savā starpā, radot nepieciešamību pēc servisa, kas katram aģentam ļauj atrast pārējos aģentus.

Daļēji izkliedētas sistēmas arhitektūra parādīta 1.8. attēlā.



1.8. att. Daļēji izkliedētas sistēmas arhitektūra

Pilnībā izkliedēta sistēma

Daudzaģentu sistēmā ietilpst tikai katram robotam atbilstošie aģenti. Vadošā aģenta nav. Katram robota aģentam ir absolūta autonomija. Katrs no robotu aģentiem ir uz mērķi virzīts un spriest spējīgs

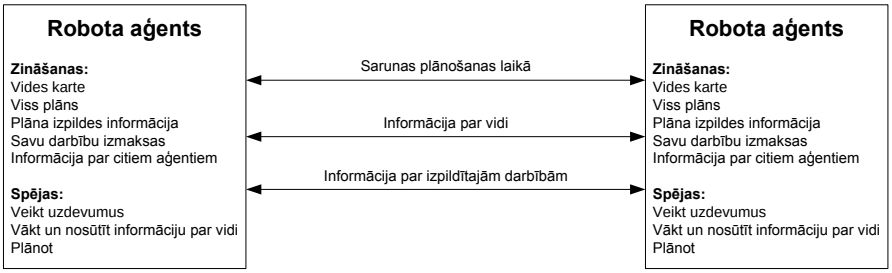
aģents, kas cenšas sasniegt sistēmas mērķi (piemēram, iztīrīt visas istabas). Uzdevumu dalīšana starp aģentiem notiek, izmantojot decentralizētu plānošanu. Aģenti var mijiedarboties, izmantojot dažādus sarunu veidus. Piemēram, aģenti var veidot plānu, izmantojot argumentētas sarunas, kurās aģenti izmanto argumentus, lai ievietotu darbību plānā un piešķirtu kādam aģentam. Viena no pieejām ir tāda, ka plānā var piešķirt darbības tikai sev, bet var izmantot argumentus, lai neļautu piešķirt uzdevumus citiem aģentiem.

Visas zināšanas par vidi un to, ko katrs aģents ir izdarījis vai plāno darīt, aģentiem ir jāvāc un jāglabā decentralizēti. Līdz ar to robotiem nepieciešams apmainīties ar šīm zināšanām, lai katram no tiem nebūtu jāvāc visas zināšanas patstāvīgi. Galvenās zināšanas par vidi ir kartes. Robotiem ir izveidoti specifiski algoritmi, kā tie apvieno savas izveidotās kartes. Līdz ar to ir jāizveido tikai risinājums, ar ko roboti apmainās ar zināšanām par to, ko katrs no tiem ir paveicis. Šo var risināt tā, ka (1) plānošanas laikā visi aģenti ir informēti par to, kādu plānu katrs no tiem izpildīs (2) ja šajos plānos notiek kādas izmaiņas, tad aģents, kurš maina plānu, to paziņo visiem pārējiem aģentiem.

Šis ir visintelektuālākais un vissarežģītākais risinājums, kurā vairākas nopietnas problēmas, kam šobrīd nav zināmu risinājumu:

- Kā lietotājs sistēmai dod uzdevumu jeb uzstāda mērķi? Teorētiski pati pilnība uz ko tiekties putekļu sūcēju robotiem ir tāda, ka tos ievieto kādā vidē (mājā/istabā/dzīvoklī) un tie paši nosaka, kad un cik bieži ir jāveic/var veikt tīrīšanu un kas ir jāveic. Lai to paveiktu, aģentiem ir autonomi jāveic uztveršana un spriešana. Ideālā gadījumā katram no robotiem ir savi mērķi, kuru sasniegšana noved pie tā, ka tiek sasniegt sistēmas mērķis.
- Kā aģenti veic decentralizētu plānošanu? Kāda ir mijiedarbība starp aģentiem? Viena no iespējām ir izmantot minētās argumentētās sarunas, bet arī tās šobrīd nav plaši pielietotas praktiskos projektos.
- Kā aģenti apmainās ar informāciju? Pilnībā izkliedētā sistēmā ir 2 iespējas:
    - Apraides ziņojumi: gan plānošanas, gan darbības laikā visa informācija, ko nepieciešams nodot citiem aģentiem tiek nosūtīta visiem aģentiem, izmantojot apraides ziņojumus.
    - Darbību izpildes laikā aģenti var apmainīties ar informāciju tikai ar tuvumā esošiem aģentiem, ja vide ierobežo komunikācijas attālumu.

Šajā realizācijā aģentiem noteikti ir jābūt spriest spējīgiem. Tie jārealizē kā BDI aģenti, kas uztur plānus un vides kartes, kā arī pārliecības par citu aģentu plāniem. Pilnībā izkliedētas sistēmas arhitektūra dota 1.9. attēlā.



1.9. att. Pilnībā izkliedēta arhitektūra

Plānošanas laikā ir iespējams arī, ka aģenti izvēlas vienu no aģentiem, kas vada plānošanas procesu un veic uzdevumu sadali. To var veikt, izmantojot, piemēram, balsošanas mehānismu. Taču tad visiem robotiem ir jābūt spējīgiem veikt plānošanu.

Izkliedēta sistēma ar kopīgu informācijas krātuvi jeb „tāfeli"

Arī šāda veida izkliedēta sistēma sastāv tikai no robotu aģentiem, taču šiem aģentiem ir viena kopēja informācijas krātuve jeb „tāfele". Visi procesi sistēmā notiek decentralizētā veidā tāpat kā iepriekšējā risinājumā. Taču aģenti var apmainīties ar informāciju, izvietojot to visiem redzamā vietā. Lai arī pēdējo gadu laikā šāda pieeja ir zaudējusi popularitāti, tai ir vairākas priekšrocības:

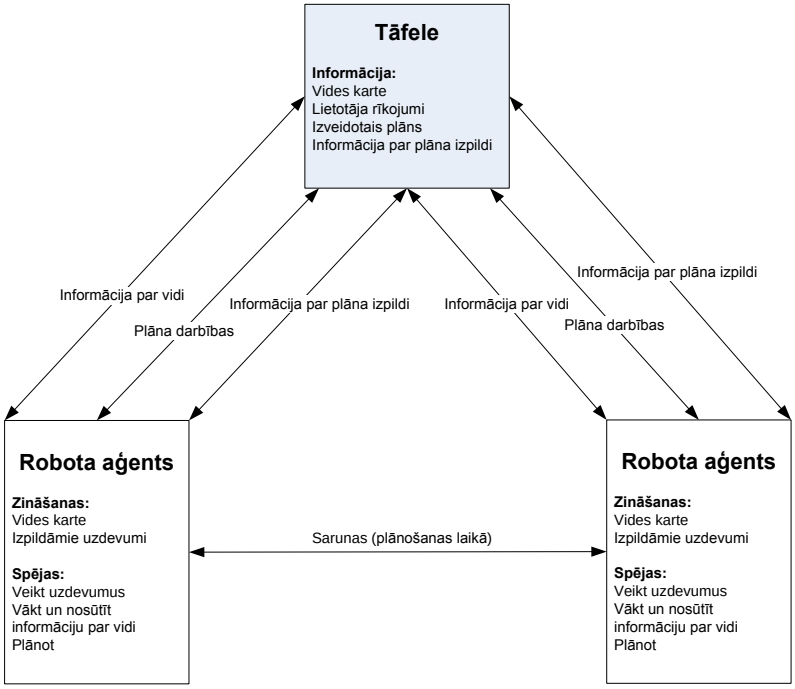
- Lietotājam ir iespēja tāfelē izvietot sistēmas mērķi un dažādus konfigurēšanas parametrus (piemēram, istabas tīrīšanas laikus). Līdz ar to šie parametri nav jāmaina katrā aģentā.
- Vides karti ir iespējams uzglabāt un mainīt tāfelē. Līdz ar to nav nepieciešami mehānismi, kā aģentiem apvienot savas kartes vai papildināt vienam otra kartes.
- Plānošanas procesu var realizēt tā, ka katrs aģents var pieteikties uz kopīgā uzdevuma daļām, veidojot plānu tāfelē. Un gadījumā, ja kāda aģenta plāns konfliktē ar cita aģenta mērķiem, tad tiek izmantotas argumentētas sarunas konfliktu novēršanai.
- Nav nepieciešami apraides ziņojumi, lai visiem aģentiem nosūtītu kādu informāciju par darbību izpildi. Var arī vienkārši izveidot sistēmu, ka katrs aģents pēc kāda plāna posma pabeigšanas par to ievieto informāciju tāfelē.
- Aģentiem nav nepieciešams glabāt daudz zināšanas, jo tās var glabāties tāfelē un būt visiem robotiem pieejams.

Robotu aģenti šādā realizācijā ir ievērojami vienkāršāki nekā iepriekšējā variantā. Aģentiem gan ir jāspēj piedalīties plānošanā un jābūt virzītiem uz mērķi. Taču plānošanu un argumentētas sarunas šajos aģentos var realizēt ar nedaudz likumiem un argumentu prioritātēm. Arī mehānismi apmaiņai ar zināšanām ir ievērojami vienkāršāki.

Atliek jautājums, kā realizēt tāfeli, lai tai varētu ērti piekļūt un būtu vienkārši sekot līdzi izmaiņām tajā. Viena iespēja ir tāfeli realizēt kā datubāzi, taču tad tai būs lēna piekļuve un sarežģīti sekot līdzi izmaiņām tajā. Otra iespēja ir tāfeli realizēt kā atsevišķu aģentu, kam var nosūtīt 3 veidu ziņojumus:

- Pieprasījums ieviest izmaiņas konkrētā informācijā, piemēram, plānā vai kartē.
- Pieprasījums pārbaudīt, vai ir kādas izmaiņas tāfelē vai kādā no datiem. Uz ko tāfeles aģents atbild ar notikušajām izmaiņām.
- Pieprasījums pēc konkrētiem datiem, uz ko tāfeles aģents atbild, nosūtot konkrētos datus.

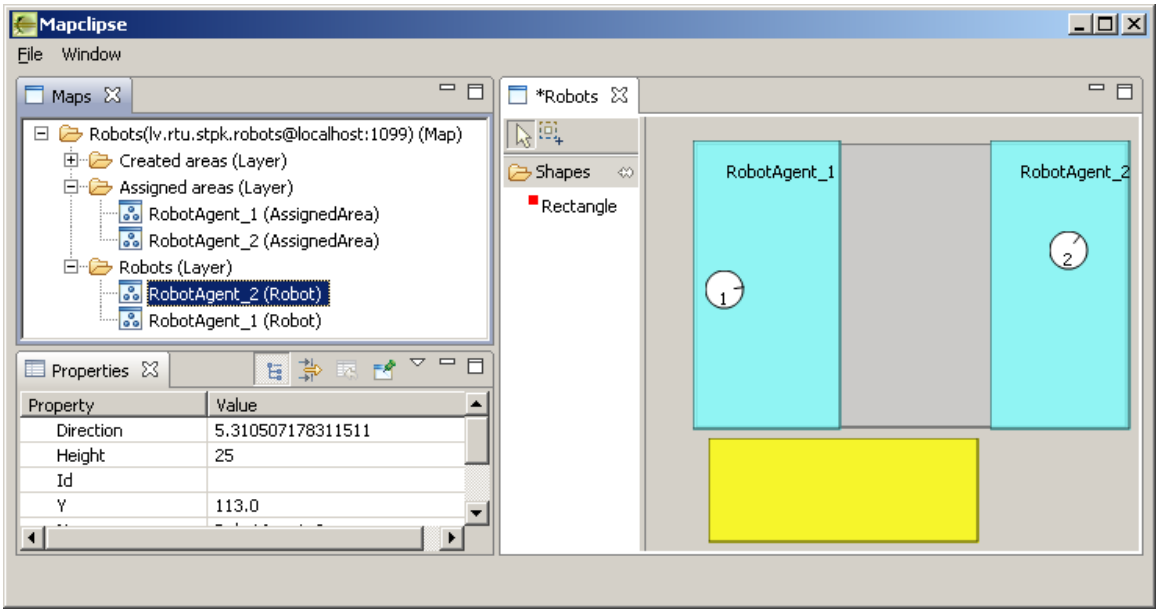
Izkliedēta arhitektūra ar kopīgu zināšanu krātuvi attēlota 1.10. attēlā.



1.10. att. Izkliedēta arhitektūra ar kopīgu zināšanu krātuvi

1.2. Izstrādātās arhitektūras pielietojuma piemērs

Balstoties uz piedāvāto arhitektūru, ir izstrādāta imitācijas vide, kuras centrālais elements ir daudzaģentu sistēma, kas veic uzdevumu dekompozīciju un piešķiršanu robotiem, izmantojot Contract Net protokolu. Daudzaģentu sistēmas darbības pārbaudei tika izveidots vienkāršs robotu imitators. Līdz ar to faktiski tika izstrādāta 1.3. attēlā redzamā daudzaģentu sistēmas un imitācijas vides kombinācija. Izstrādātās sistēmas ekrāna attēls dots 1.11. attēlā. Sīkāks izstrādātā imitatora apraksts ir pieejams [13].



1.11. att. Izstrādātā imitatora ekrāna attēls (aizgūts no [13])

Veicot daudzaģentu sistēmā balstītās imitācijas vides prototipa izstrādi, tika pētīta sistēmas projektēšanas un realizācijas sarežģītība aģentorientētā veidā. Konkrētāk, tika analizēts, cik piemērota ir uzvedībā sakņotu aģentu paradigma un to realizējošā JADE aģentu izstrādes vide robotu vadības sistēmas realizācijai. Kā rezultātā ir secināts, ka robotizētā sistēma atbilst visām aģentu īpašībām (autonomija, reaktivitāte, proaktivitāte, sociālās spējas, apmācība) un līdz ar to aģentiem saskaņā ar teorētiskajiem pētījumiem ir jābūt piemērotai pieejai šādu sistēmu realizēšanā. Līdz ar to pirmais sistēmas prototips (daudzaģentu sistēma kopā ar robotu imitatoru) tika izstrādāta JADE vidē. Tomēr izstrādes laikā tika izdarīti šādi secinājumi par praktisku robotu vadības sistēmu projektēšanu un izstrādi, izmantojot uzvedībā sakņotu aģentu pieeju [15]:

- Sistēmas zema līmeņa projektēšana kļūst sarežģītāka, jo ir jāveido liels skaits uzvedības klašu. Piemēram, viena ziņojuma nosūtīšanai un saņemšanai var būt nepieciešamas līdz 6 klasēm (pa vienai ziņojuma nosūtīšanai un saņemšanai, lēmumu pieņemšanai nosūtītāja un saņēmēja aģentos un gaidīšanas mehānismi abos aģentos). Līdz ar to sistēmā ir liels klašu skaits, kas rada uzturēšanas problēmas.
- Nav skaidrs, kā pareizi organizēt ziņojumu apstrādi. Ja katram aģentam veido vienu ziņojumu saņemšanas uzvedību, tad tajā būs jāapstrādā daudz dažādus gadījumus un liela saņemamo ziņojumu skaita gadījumā kods saturēs daudz secīgus IF-Then operatorus, kas apgrūtina koda izprašanu un uzturēšanu. Alternatīva ir katru ziņojumu saņemt uzreiz atbilstošā uzvedībā. Šādā projektējumā rodas divas citas problēmas. Pirmkārt, saņemot katru ziņojumu, darbu jāatsāk visām uzvedībām, kas gaida uz ziņojumiem. Tas nozīmē, ka ir jāveic liels skaits darbību un

pārslēgšanos starp šīm uzvedībām. Otrkārt, rodas problēmas gadījumos, ja vienu ziņojumu vajag vairākām uzvedībām, piemēram, ja tiek ieviesta ziņojumu žurnalēšana. Tādā gadījumā, ja viena uzvedība izmaina ziņojumu, tad otra uzvedība jau var šo ziņojumu palaist garām.

- Rezultātu nodošana no vienas uzvedības citai ir sarežģītāka kā metodes izsaukuma gadījumā, jo uzvedību izpildes modelis ir asinhrons un ir nepieciešams projektēt speciālu gaidīšanas mehānismu, kas atkal palielina uzvedību skaitu un sarežģī projektējumu.

Līdz ar to ir izdarīts secinājums, ka šādu uzvedībā sakņotu aģentu izmantošana konceptuāli vienkāršo aģentu īpašību implementēšanu robotos. Tomēr šīs pieejas praktiskai pielietošanai ir ne tikai ieguvumi, bet arī trūkumi. Viens no svarīgākajiem faktoriem tieši robotu vadības sistēmas gadījumā ir lielā skaita uzvedību klašu ietekme uz reālā laika darbību, jo rodas aizkave pie katras uzvedības izpildes sākšanas. Līdz ar to ir secināts, ka pieeju var izmantot tikai, ja aizkave ir mazāka kā to pieļauj reālā laika ierobežojumi, kas ir atkarīgi no katras konkrētās sistēmas.

Izmantojot izstrādāto imitatoru ir veikti pētījumi, cik efektīvi var izmantot secīgu Contract Net protokola realizāciju uzdevumu piešķiršanai. Tā rezultātā ir secināts, ka šādu pieeju var izmantot uzdevumu piešķiršanai, bet tomēr eksistē vairāki speciāli gadījumi, kad uzdevumi netiek piešķirti piemērotākajiem robotiem un rezultātā tiek iegūts neoptimāls uzdevumu sadalījums [14].

2. Secinājumi

Pētījumu rezultātā ir izstrādāti dažādi daudzaģentu arhitektūru varianti, sākot no pilnībā centralizētām pieejām, beidzot ar pilnībā decentralizētām. Katra izstrādātā arhitektūra ir piemērota sava veida risinājuma izstrādāšanai. Pētījuma ietvaros ir veikta analīze ar mērķi noteikt, kura no arhitektūrām ir piemērota daudzrobotu vadības sistēmas izstrādei, kā rezultātā par piemērotāko daudzrobotu sistēmas, kas sastāv no putekļu sūcēju robotiem, vadības sistēmas realizācijai ir atzīta 1.5. attēlā dotā arhitektūra. Šai arhitektūrai ir arī tāda priekšrocība, ka balstoties uz to implementēta sistēma ir salīdzinoši vienkārši pārveidojama, lai tā atbilstu vairāk decentralizētām pieejām. Līdz ar to dokumentā aprakstīta daudzaģentu sistēmu pētījumu rezultātā ir rekomendēta šāda pieeja turpmākajiem projekta pētījumiem:

1) Vispirms ir jāizstrādā sistēma, kas balstās uz 1.5. attēlā redzamo arhitektūru ar vienu vadošo aģentu. Pēc šādas arhitektūras realizācijas visa loģiskā puse ir gatava un to var bez izmaiņām ieviest autonomo robotu sistēmā.

2) Projekta beigu fāzē vai turpmākajos pētījumos var pāriet uz autonomiem robotiem. Autonomo robotu izstrādei atliks atrisināt tādus jautājumus:
   a) aģenta fiziska izvietošana robotam pieejamajos skaitļošanas resursos (iespējamais risinājums ir JADE LEAP izmantošana);
   b) aģentu fiziska mijiedarbība;
   c) mijiedarbības starp aģentu un robotu pārtaisīšana, kurai šajā arhitektūrā jānotiek bez servera starpniecības.

Pētījumu ietvaros ir analizētas arī piedāvātās arhitektūras praktiskas realizācijas iespējas. Šīs analīzes rezultātā ir secināts, ka lai arī, balstoties uz teorētiskām īpašībām uzvedībā sakņoti aģenti ir piemēroti robotu vadības sistēmas realizācijai, šādu sistēmu realizācija aģentorientētas programmatūras veidā un izmantojot uzvedībā sakņotu aģentu izstrādes vidi JADE var būt sarežģīta un apgrūtināt sistēmas darbību reālā laikā. Līdz ar to pirms šādas izstrādes pieejas izvēles ir rūpīgi jāizvērtē, vai tā netraucēs realizēt reālā laikā strādājošu sistēmu un vienlaicīgi ļaus izstrādāt sistēmu ar tādu projektējumu, kas ļauj to vienkārši attīstīt.

Tāpat pētījumu ietvaros ir analizēti aģentu mijiedarbības mehānismi uzdevumu sadalei daļēji decentralizēta risinājuma gadījumā [5]. Šādām situācijām klasiska pieeja ir izmantot Contract Net protokolu [4]. Tomēr pētījuma rezultātā ir secināts, ka tas nedod optimālu uzdevumu sadalījumu. Tādēļ ir jāizmanto kāds sarežģītāks uzdevumu sadales protokols, piemēram, [6], [8], [9], [11].

2.1. Izstrādātās arhitektūras pielietojuma apraksts

Izstrādātā arhitektūra ir pielietojama daudzrobotu vadības sistēmu izstrādei. Tā pamatā ir izstrādāta daudzrobotu sistēmām, kas sastāv no mobiliem robotiem un risina teritorijas nosegšanas uzdevumu (*area coverage*), kas ir labi zināms mobilo robotu pētījumos. Tomēr izstrādātā arhitektūra ir pielietojama arī citām līdzīgām daudzrobotu sistēmām, kam ir līdzīgas īpašības:

- Sistēmai ir centralizēti jāsaņem liels uzdevums, ko tai ir jāatrisina, izmantojot sistēmā esošos robotus.
- Sistēmai ir jāveic uzdevumu dekompozīcija un apakšuzdevumu piešķiršana robotiem.
- Sistēmai ir jākomunicē ar robotiem kāda iemesla dēļ, piemēram, lai sekotu līdz darbu progresam un/vai robotu atrašanās vietai.
- Robotos ir pieejamas ierobežotas skaitļošanas jaudas un tādēļ globālas plānošanas līmeņa lietas, kā piemēram, uzdevumu sadale ir jāveic uz servera.

Lai arhitektūru sekmīgi pielietotu cita veida sistēmās, ir nepieciešams specificēt šādas lietas:

- Jāprecizē, kāda mijiedarbība un starp kuriem aģentiem ir nepieciešama, piemēram, definēt, ka robotiem ir jāziņo sava atrašanās vieta reizi sekundē.
- Jādefinē mijiedarbības protokoli, kas izmantojami uzdevumu sadalē un citai mijiedarbībai starp aģentiem.
- Jāizanalizē, kādi ir tehniski ierobežojumi sistēmas praktiskai implementācijai, piemēram, reāla laika ierobežojumi, kas var ierobežot izstrādes vides izvēli.

2.2. Nodevuma aprobācija

Pētījuma ietvaros izstrādātā arhitektūra ir aprobēta starptautiski atzītā publikācijā:

1. Lavendelis E., Liekna A., Ņikitenko A., Grabovskis A., Grundspeņķis J. Multi-Agent Robotic System Architecture for Effective Task Allocation and Management // Recent Researches in Communications, Electronics, Signal Processing & Automatic: Proceedings of the 11th WSEAS International Conference on Signal Processing, Robotics and Automation (ISPRA '12), United Kingdom, Kembridža, 22.-24. February, 2012. - pp 167-174.

Tāpat vairākas turpmākās projekta ietvaros izstrādātās publikācijas balstās šī pētījuma rezultātos:

1. Liekna A., Lavendelis E., Grabovskis A. Experimental Analysis of Contract NET Protocol in Multi-Robot Task Allocation // Scientific Journal of RTU. 5th series, Computer Science. - 213. vol., 2012, pp 6-14.
2. Grabovskis A. The Generic Map Visualization Framework // Scientific Journal of RTU. 5th series, Computer Science. – Vol. 13. (2012), pp. 15.-21.

[1] Liekna A., Lavendelis E. Ņikitenko A. Challenges in Development of Real Time Multi-Robot System Using Behaviour Based Agents. In Proceedings of 10th International Symposium on Distributed Computing and Artificial Intelligence, Salamanca, Spain, 22nd-24th May, 2013 (Accepted for publication).

Atzinība

Dokumentā aprakstītais pētījums ir finansēts no ERAF projekta „Daudzaģentu robotizētas intelektuālas sistēmas tehnoloģijas izstrāde”, projekta ieviešanas numurs 2010/0258/2DP/2.1.1.1.0/10/APIA/VIAA/005.

Literatūra

[1] Bellifemine F. et al, *Developing Multi-Agent Systems With JADE*, Wiley, 2004. 286 p.

[2] Bordini R. H. et al, *Multi-Agent Programming: Languages, Platforms and Applications.* Springer, 296 p., 2010.
[3] Luck M. et al., *Agent Based Software Development*. Artech House, 208 p., 2004.
[4] *FIPA Contract Net Interaction Protocol Specification.* Foundation for Intelligent Physical Agents, Geneva, Switzerland, 2002.
[5] Liekna A., et al, Analysis of Contract NET Protocol in Multi-Robot Task Allocation. *Scientific Journal of RTU* (accepted), 2012.
[6] Brian P. Gerkey. On Multi-Robot Task Allocation. PhD Dissertation. University of Southern California Computer Science Department, August 2003.
[7] Brian P. Gerkey and Maja J Mataric´ "A Framework for Studying Multi-Robot Task Allocation". In A.C. Schultz and others, editors, Multi-Robot Systems: From Swarms to Intelligent Automata, Volume II, pages 15-26, Kluwer Academic Publishers, the Netherlands, 2003
[8] M Bernardine Dias. TraderBots: A New Paradigm for Robust and Efficient Multirobot Coordination in Dynamic Environments, doctoral dissertation, tech. report , Robotics Institute, Carnegie Mellon University, January, 2004
[9] Zlot R.M. and Stentz A. Market-based Multirobot Coordination for Complex Tasks. International Journal of Robotics Research, Special Issue on the 4th International Conference on Field and Service Robotics, Vol. 25, No. 1, January, 2006, pp. 73-101.
[10] Zlot R.M. and Stentz A. Complex Task Allocation For Multiple Robots. Proceedings of the
[11] Dias M.B. et al. Market-based multirobot coordination: a survey and analysis. Proceedings of the IEEE, Vol. 94, No. 7, July, 2006, pp. 1257 – 1270.
[12] Matthew E. Taylor, Manish Jain, Christopher Kiekintveld, Jun-young Kwak, Rong Yang, Zhengyu Yin, and Milind Tambe. Two Decades of Multiagent Teamwork Research: Past, Present, and Future. Book Chapter published at CARE workshop, 2010.
[13] Grabovskis A. The Generic Map Visualization Framework // Scientific Journal of RTU. 5th series, Computer Science. – Vol. 13. (2012), pp. 15.-21.
[14] Liekna A., Lavendelis E., Grabovskis A. Experimental Analysis of Contract NET Protocol in Multi-Robot Task Allocation // Scientific Journal of RTU. 5th series, Computer Science. - 213. vol., 2012, pp 6-14.
[15] Liekna A., Lavendelis E. Nikitenko A. Challenges in Development of Real Time Multi-Robot System Using Behaviour Based Agents. In Proceedings of 10th International Symposium on Distributed Computing and Artificial Intelligence, Salamanca, Spain, 22nd-24th May, 2013 (Accepted for publication).