VIDZEMES AUGSTSKOLA
INŽENIERZINĀTŅU FAKULTĀTE

Artis Aizstrauts

HETEROGĒNU UN SADALĪTU IMITĀCIJAS MODEĻU VIENKĀRŠOTAS KOMUNIKĀCIJAS ĪSTENOŠANA

PROMOCIJAS DARBS
Tematiski vienota zinātnisko publikāciju kopa

Zinātnes doktora grāda (Ph. D.) iegūšanai Elektrotehnikas, elektronikas, informācijas un komunikācijas tehnoloģiju nozarē

Promocijas darba zinātniskais vadītājs
Prof., Dr.sc.ing. Egils Ginters

Valmiera, 2024

PROMOCIJAS DARBS IZSTRĀDĀTS:

Vidzemes Augstskolas
Inženierzinātņu fakultāte
Doktora studiju programma “Sociotehnisko sistēmu modelēšana”

ZINĀTNISKAIS VADĪTĀJS:

Egils Ginters
Profesors, Dr.sc.ing.
Rīgas Tehniskā universitāte

RECENZENTI:

Pēteris Grabusts
Profesors, Dr.sc.ing.
Rēzeknes Tehnoloģiju akadēmija

Arnis Lektauers
Asociētais profesors, Dr.sc.ing.
Rīgas Tehniskā universitāte

Emilio Jiménez Macías
Profesors, Ph.D.
La Rjohas Universitāte

APSTIPRINĀJUMS

Apstiprinu, ka esmu izstrādājis šo promocijas darbu, kas iesniegts izskatīšanai Vidzemes Augstskolā zinātnes doktora grāda (*Ph. D.*) iegūšanai. Promocijas darbs zinātniskā grāda iegūšanai nav iesniegts nevienā citā universitātē.

Artis Aizstrauts ……………………………. (paraksts)

Datums: ………………………

Promocijas darbs sagatavots kā tematiski vienota zinātnisko publikāciju kopa. Promocijas darbs sastāv no kopsavilkuma un 13 publikācijām, kas apkopotas B. pielikumā. Publikācijas ir uzrakstītas angļu valodā, bet to kopējais apjoms ir 128 lpp.

ANOTĀCIJA

Autors: Artis Aizstrauts

Darba vadītājs: Profesors, Dr. sc. ing. Egils Ginters

Heterogēnu un sadalītu imitācijas modeļu vienkāršotas komunikācijas īstenošana**.** - Promocijas darbs, Valmiera: Vidzemes augstskola, 2024.- 169 lpp., 2 tabulas, 9 attēli, 2 pielikumi.

Atslēgvārdi: Vienkāršotas komunikācijas vide; Sadalītas imitāciju modelēšanas sistēmas; Imitācijas modeļu komunikācijas vides; Imitāciju modelēšanas tehnoloģijas; Sociotehniskas sistēmas

Straujā un visaptverošā digitālo tehnoloģiju attīstība ne tikai ietekmē cilvēku dzīvesveida izvēles, bet arī lielā mērā nosaka sabiedrību un ekonomiku attīstību. Sabiedrību var raksturot kā kompleksu sociotehnisku sistēmu, un līdz ar to sabiedrības modelis vienmēr būs heterogēns un stohastisks, un tā rezultāti – grūti paredzami. Lai pētniecībā pielietotu imitāciju modelēšanas metodes un analizētu šādas sistēmas funkcionēšanu, nepieciešams sadalīts un heterogēns modelis. Tomēr šāda modeļa izveide ir sarežģīta, jo trūkst pieejamu un lietošanā vienkāršu tehnoloģiju, lai nodrošinātu komunikāciju starp vairākiem atšķirīgiem modeļiem, kas tajā pašā laikā modelētājam neprasa būtiskus darba laika ieguldījumus vai specifiskas programmatūras izstrādes zināšanas. Promocijas darba mērķis ir nodrošināt iespējas modelētājiem bez specifiskām zināšanām un prasmēm programmēšanas inženierijā, konstruēt sadalītus un heterogēnus imitācijas modeļus, balstoties uz atvērtas un vienkāršotas komunikāciju vides konceptuālu ietvaru. Promocijas darbā ir aplūkota Vienkāršotas komunikācijas vides (turpmāk tekstā ECE - *Easy Communication Environment*) arhitektūra un funkcionalitāte. ECE ir izstrādāta un validēta ilgākā laika periodā, mainot izmantotās tehnoloģijas, bet vienmēr saglabājot nemainīgu ideju par tādas komunikācijas vides izstrādi, kas nodrošinātu sadalītu un heterogēnu imitāciju modeļu izveidošanu un savietojamību. Promocijas darba rezultāti būs noderīgi kā pētniekiem, tā dažādu jomu profesionāļiem, kas nodarbojas ar sadalītu imitācijas modeļu un datu apstrādes sistēmu projektēšanu.

ABSTRACT

Author: Artis Aizstrauts

Scientific supervisor: Profesor, Dr. sc.ing. Egils Ginters

An Easy Communication Implementation of Heterogeneous and Distributed Simulation Models. – Doctoral thesis, Valmiera: Vidzemes University of Applied Sciences, 2024.- 169 p., 2 tables, 9 figures, 2 appendices.

Keywords: Easy Communication Environment; Distributed simulation systems; Simulation model communication environments; Simulation technologies; Socio-technical systems

The rapid and pervasive development of digital technologies not only affects but determines the life of society and the economy, as well as changing our living models. Today, society can be described as a complex socio-technical system, the model of which is heterogeneous, stochastic, and the results are therefore difficult to predict. To simulate the functioning of such a system, a distributed and heterogeneous model is needed. However, the designing of such simulation models is difficult due to the lack of easy-to-use communication tools and mechanisms that are not laborious and do not require specific software engineering knowledge from the modeler. The aim of the doctoral thesis is to provide opportunities for modelers without specific knowledge and skills in programming engineering to construct distributed and heterogeneous simulation models based on the conceptual framework of an open and simplified communication environment. The thesis dealt with the Easy Communication Environment (ECE) architecture and functionality that has been developed and validated over time, changing implementation stack but keeping the basic idea of designing a communication and data exchange environment unchanged, to ensure that distributed and heterogeneous simulation models can be easily developed, and interoperability can be achieved. The results of the thesis are useful for both researchers and professionals in various fields who design distributed simulation models and data processing systems.

SATURS

SAĪSINĀJUMI

ABM	–	Uz aģentiem balstīts modelis (*Agent Based Model*)
ALSP	–	*Aggregate Level Simulation Protocol*
AMQP	–	*Advanced Message Queuing Protocol*
API	–	*Application Programming Interface*
CLD	–	*Causal Loop Diagram*
CORBA	–	*Common Object Request Broker Architecture*
DDS	–	*Data-Distribution Service for Real-Time Systems*
DIS	–	*Distributed Interactive Simulation*
EC2	–	*Amazon Elastic Compute Cloud*
ECE	–	*Easy Communication Environment* (Vienkāršotas komunikācijas vide)
FIPA	–	*The Foundation for Physical Intelligent Agents*
FOM	–	*Federation Object Model*
HLA	–	*High Level Architecture*
HTTP	–	*Hypertext Transfer Protocol*
IoT	–	*Internet of Things*
ISO OSI 7498	–	*International Organization for Standardization Open Systems Interconnection 7498 standard*
JSON	–	*JavaScript Object Notation*
LCM	–	*Lightweight Communications and Marshalling*
MI	–	Mākslīgais intelekts (*Artificial Intelligence*)
MQTT	–	*Message Queue Telemetry Transport*
ORB	–	*Object Request Broker*
QoS	–	*Quality of Service*
ROS	–	*Robot Operating System*
SOA	–	*Service Oriented Architecture*

1. PROMOCIJAS DARBA VISPĀRĒJS RAKSTUROJUMS

1.1. Zinātnes nozare un apakšnozare

Promocijas darbs ir attiecināms uz nozari: “Elektrotehnika, elektronika, informācijas un komunikāciju tehnoloģijas”, kā arī piekritīgo apakšnozari: “Sistēmu analīze, modelēšana un projektēšana”.

1.2. Tēmas aktualitāte

Digitālās tehnoloģijas ir kļuvušas par mūsdienu sabiedrības un ekonomikas stūrakmeni. Šo tehnoloģiju kvalitāte ietekmē mūsu labbūtību un dzīves apstākļus. Digitālā transformācija skar cilvēku ikdienu, indivīdu uzvedību un komunikācijas veidu. Tāpat tehnoloģijas maina ražošanas, publiskās pārvaldes, lauksaimniecības, izglītības u.c. nozares un lēmumu pieņemšanu tajās. Īpaši nozīmīga ietekme ir tādām tehnoloģijām kā mākslīgais intelekts (MI) un lielo datu analītika, mākoņdatošana, mobilās tehnoloģijas, biznesa informācijas sistēmas un lietu internets (IoT) (Aizstrauts un Ginters, 2024).

Digitālo tehnoloģiju tirgus izplešas ievērojamā ātrumā un apjomā. Globālais digitālās transformācijas tirgus 2023.gadā bija 880,28 miljardi USD, un saskaņā ar prognozēm 2024.gadā tas sasniegs 1070,43 miljardus USD. Tirgus izaugsmi raksturo kopējais gada pieauguma temps par aptuveni 27.6 %. Prognozes pie šāda pieauguma tempa sasniedz 4717.78 miljardus USD līdz 2030.gadam (Grand View Research, 2024).

Sabiedrība mūsdienās raksturojama kā sociotehniska sistēma, kuras eksistence ir atkarīga no digitālo tehnoloģiju ilgtspējīgas attīstības. Digitālās tehnoloģijas, kas ietver būtisku stohastisko komponenti un slēptu ietekmju kopumu (Ginters, Revathy, 2021), ievērojami ietekmē indivīdu ikdienu, kā arī sabiebrības, ekonomikas un apkārtējās vides attīstību.

Attīstoties dažādām tehnoloģijām, kas var ievākt, apstrādāt, analizēt un kontekstualizēt datus, pieaug to nozīme lēmumu pieņemšanā. Datos balstīti lēmumi un pamatotas prognozes ir neatņemama dažādu institūciju darbības daļa Līdz ar to arvien biežāk rodas nepieciešamība pēc pamatotām prognozēm un datos balstītiem lēmumiem. Savukārt, imitāciju modelēšana noder kā metode ticamu prognožu izstrādē un dažādu scenāriju attīstības validācijai.

Sociotehniskas sistēmas modelis ir objektīvās realitātes vienkāršots attēlojums, tomēr, saglabājot stohastisko ietekmes faktoru ievērojamo īpatsvaru, modelis nezaudē arī savu sarežģītību. Kompleksas problēmas nevar izskaidrot ar primitīviem un/vai homogēniem modeļiem. Sociotehnisku sistēmu modeļi ir heterogēni, un tas parasti nozīmē, ka tiek izmantotas dažādas un grūti savietojamas imitācijas modelēšanas tehnoloģijas. Diskrētu notikumu modelēšana ir piemērota procesu, piegāžu un rindu modelēšanai, savukārt, uzvedības un individuālu objektu mijiedarbības modelēšanai parasti tiek izmantoti uz aģentiem balstīti imitāciju modeļi, bet vispārīgas izmaiņas plašākā mērogā vislabāk ir aprakstīt ar sistēmu dinamikas vienādojumiem. Gadījumos, kad ir jāpievieno arī datu apstrādes un analīzes sadaļas, neizbēgami ir jākonstruē heterogēna un sadalīta modelēšanas sistēma (Aizstrauts un Ginters, 2024).

2012. gadā, veicot sadalītu imitāciju modeļu platformu un rīku analīzi, autori (Aizstrauts et al., 2012) nonāca pie secinājuma, ka pieejamo iespēju klāsts aprobežojas ar DIS (*Distributed Interactive Simulation*), HLA (*High Level Architecture*), CORBA (*Common Object Request Broker Architecture*) vai pielāgotiem, uz tīmekļa pakalpojumiem balstītiem, risinājumiem. Universāls un lietotājam draudzīgs rīks sadalītu imitāciju modelēšanas sistēmu veidošanai diemžēl neeksistēja.

Arī šodien nekas būtiski nav mainījies. Ir radīti atsevišķi, pietiekami ātri komunikācijas rīki ROS (*Robot Operating System*) un LCM (*Lightweight Communications and Marshalling*). Tomēr tie ir veidoti reālā laika robotikas risinājumiem, turklāt augstāk minētā iemesla dēļ – pietrūkst rīku sadalītu imitācijas modeļu konstruēšanai (Xu et al., 2021). IoT tirgus izaugsme ir veicinājusi tādu komunikācijas rīku attīstību, kas nodrošina datu apmaiņu starp tehniskām ierīcēm, piemēram, IoTivity (Xu et al., 2021). Tomēr šie rīki ir slikti piemēroti heterogēnu un daudzveidīgu modeļu saziņas nodrošināšanai. Cerības raisīja protokols un standarts DDS (*Data-Distribution Service for Real-Time Systems*) (OMG, 2015) reālā laika sadalītu operētājsistēmu konstruēšanai. Tomēr tas vairumā gadījumu ir palicis instrukciju un vadlīniju līmenī par to, kā sasniegt modeļu savietojamību, diemžēl nenonākot pie praktiskiem risinājumiem (Aizstrauts un Ginters, 2024).

Diemžēl ir jāatzīst, ka dominējošais rīks sadalītu imitācijas modeļu mijiedarbības nodrošināšanā aizvien ir HLA, ko izveidoja ASV Aizsardzības departaments 1995. gadā (Dahmann, Fujimoto and Weatherly, 1997). Kopš tā laika HLA ir papildināts, uzlabots un plaši pielietots arī civilajā dzīvē (Jabbour, Possik and Zacharewicz, 2023; Possik, 2021). HLA faktiski ir kļuvis par standarta risinājumu sadalītu modeļu savietojamības nodrošināšanai.

Viens no tā galvenajiem trūkumiem bija un joprojām ir tas, ka tas uzskatāms par pārlieku "*..*"*smagu*" *jeb ļoti sarežģītu, grūti apgūstamu un tādējādi laikietilpīgu, lai to ieviestu un izmantotu*" (Strassburger, Schulze and Fujimoto, 2008). Likumsakarīgi, ka bez labām programmatūras izstrādes prasmēm, tas praktiski nav iespējams.

Tātad citu jomu profesionāļiem ceļš līdz sadalīta imitāciju modeļa izveidei, neiesaistot programmēšanas speciālistus, joprojām ir stipri ierobežots. Turklāt jaunizveidotā modeļa dzīves ciklu visdrīzāk noteiks programmēšanas inženiera ieinteresētība. Papildu apgrūtinājums šādam risinājumam ir nepieciešamais finansējums, kura apjoms būtiski pieaugs, piesaistot projektiem augstas kvalifikācijas programmēšanas inženierus. Papildus izmaksas, protams, radīs arī komerciālo rīku licenču nosacījumi, apgrūtinot izveidotā modeļa izplatīšanu un ieviešanu (Aizstrauts un Ginters, 2024).

Šo problēmu risinājumam simulāciju vides izstrādātāji bieži apvieno un iekļauj produktā vairākas modelēšanas tehnikas, kas ļauj veidot heterogēnus modeļus, piemēram, AnyLogic (Borshchev, 2014), un izmantot modeļu komunikācijai iekšējos mehānismus. Tomēr paliek atklāts jautājums, kā AnyLogic integrēt iepriekš citās imitāciju modelēšanas vidēs veidotus modeļus, kā FlexSim, (Wang, Wang un Zhang, 2022), NetLogo (Wilensky un Rand, 2015), u.c., vienlaikus nodrošinot saikni ar citām datu apstrādes apakšsistēmām.

Ir akūta nepieciešamība pēc metodes un tehnoloģijas, kas ļautu modelētājiem padarīt iespējamu līdz šim neiespējamo un atļautu industrijas profesionāļiem ar pamata zināšanām informācijas tehnoloģijās un modelēšanā veidot sadalītus heterogēnus imitāciju modeļus pašu spēkiem, nepiesaistot nedz programmētājus, nedz papildu finansējumu.

Ja modelētājs izmanto uz aģentiem balstītus modeļus (ABM), tad viņam ir vismaz minimāla izpratne par objektorientētu programmēšanu. Ja modelētājs izmanto diskrētu notikumu imitācijas rīkus, tad viņam ir pamata zināšanas statistikas datu apstrādē un izpratne par varbūtību sadalījumiem. Ja modelētājs izstrādā sistēmu dinamikas modeļus, tad viņš ir pazīstams ar diferenciālvienādojumiem. Proti, tās ir pietiekamas zināšanas, lai izveidotu sadalītu un heterogēnu imitācijas modeli, izmantojot Vienkāršotas komunikācijas vidi (ECE).

1.3. Pētījuma priekšmets un objekts

Pētījuma priekšmets ir sadalītas imitāciju modelēšanas komunikāciju vides, bet pētījuma objekts ir komunikācijas un datu apmaiņas mehānisms.

1.4. Pētījuma jautājums

Vai un kā ir iespējams pielāgot agrāk eksistējošas sadalītas imitācijas modelēšanas vai arī datu apstrādes komunikācijas vides, kuras būtu piemērotas modelētājiem bez īpašām zināšanām programmēšanas inženierijā (atbilstība Programmēšanas inženiera NACE 62.01 profesionālās kvalifikācijas standarta 6.līmenim un augstāk), lai izstrādātu sadalītus un heterogēnus imitācijas modeļus?

1.5. Hipotēze

Vienkāršotas komunikācijas vide (ECE) ir piemērota un universāla alternatīva modelētājiem bez specifiskām zināšanām un prasmēm programmēšanas inženierijā, kas spēj aizvietot augsta līmeņa arhitektūras (HLA) risinājuma lietojumu sadalītos un heterogēnos ierobežotas drošības imitāciju modeļos.

1.6. Promocijas darba mērķis un uzdevumi

Darba mērķis. Nodrošināt iespējas modelētājiem bez specifiskām zināšanām un prasmēm programmēšanas inženierijā, konstruēt sadalītus un heterogēnus imitācijas modeļus, balstoties uz atvērtas un vienkāršotas komunikāciju vides konceptuālu ietvaru.

Lai sasniegtu darba mērķi tiek izpildīti turpmāk minētie **uzdevumi**:

1. Veikt eksistējošu sadalītas imitācijas modelēšanas un datu apstrādes komunikācijas mehānismu analīzi, lai noskaidrotu to lietošanas piemērotību un atbilstību modelētāja prasmēm, ja nozares speciālists nestrādā programmēšanas inženierijā.
2. Izveidot atvērtu un vienkāršotu konceptuālo ietvaru, kas nodrošina iespējas modelētājiem bez specifiskām zināšanām un prasmēm programmēšanas inženierijā, konstruēt sadalītus un heterogēnus imitāciju modeļus.
3. Pārbaudīt un validēt sadalītu un heterogēnu modeļu vienkāršotas komunikācijas vidi dažādos lietojumos.

1.7. Zinātniskā novitāte

Vienkāršotas komunikāciju vides (ECE) atvērts vairākslāņu ietvars nodrošina sadalītu un heterogēnu imitācijas modeļu sadarbību, nodrošinot, modelētājiem bez īpašām programmēšanas inženierijas prasmēm, iespējas veidot daudzveidīgus sociotehnisko sistēmu attīstības scenāriju imitācijas modeļus.

Priekšrocības. Vienkāršotas komunikācijas vide (ECE) dod iespējas modelētājiem bez specifiskām prasmēm programmēšanas inženierijā, izveidot sadalītus un heterogēnus imitāciju modeļus un savienot tos ar datu apstrādes sistēmām. Līdz šim esošā alternatīva ir slēgtas imitāciju modelēšanas vides lietojums, atsakoties no agrāk radītu heterogēnu modeļu izmantošanas.

Ierobežojumi. ECE izmanto visaptverošas apraides AMQP (*Advanced Message Queuing Protocol*) protokolu, bet nenodrošina apstiprinājuma saņemšanu, kā rezultātā tehnoloģija nav rekomendējama, izmantošanai paaugstinātas drošības prasību apstākļos.

1.8. Pamata tēzes

Promocijas darba izstrādes rezultāti atļauj aizstāvēt turpmāk minētās pamata tēzes:

1. Esošās sadalītas imitāciju modelēšanas komunikāciju vides nav piemērotas modelētājiem bez labām prasmēm programmēšanas inženierijā, kas apgrūtina sadalītu un heterogēnu imitāciju modeļu izveidošanu un lietošanu dažādu scenāriju pārbaudei, kas negatīvi ietekmē lēmumu pieņemšanas pamatotību.
2. Vienkāršotas komunikācijas vides konceptuālais ietvars atļauj modelētājiem bez specifiskām zināšanām programmēšanas inženierijā, konstruēt heterogēnus un sadalītus imitāciju modeļus daudzveidīgiem lietojumiem.
3. Vienkāršotas komunikācijas vides ietvars ir atvērts un nodrošina ērtas izmantošanas iespējas ne tikai imitāciju modelēšanas, bet arī datu sadalītas apstrādes sistēmu veidošanai.

1.9. Darba struktūra un apjoms

Promocijas darbs ir sagatavots, kā vienota tematisko publikāciju kopa, kas ietver kopsavilkumu un autora sagatavotās zinātniskās publikācijas Vienkāršotas komunikāciju vides (ECE) izstrādes laikā.

Pirmā daļa ietver promocijas darba vispārīgu raksturojumu, kas ir uzlūkojams kā darba ievada daļa. Savukārt, otrā daļa satur informāciju par darba rezultātu aprobāciju dažādos zinātniskas pētniecības projektos, nacionālās un starptautiskās zinātniskās konferencēs, kā arī piekritīgo publikāciju uzskaitījumu. Trešā daļa analizē pētnieka sasniegtos darba rezultātus, un konkrēti, aplūko ECE attīstības posmus, strukturālo arhitektūru un funkcionalitāti, kā arī apskata konkrētu ECE lietojuma piemēru. Daļas beigās tiek iniciēta neliela diskusija par ECE piemērotību dažādu lietojumu modelēšanai. Darba beigās ir pievienoti autora secinājumi un kopsavilkumā izmantoto izziņas avotu saraksts. Promocijas darbam ir divi pielikumi, kas ietver funkcionalitātes BPMN2 diagrammu, kā arī piekritīgo publikāciju kopijas.

2. DARBA APROBĀCIJA UN PUBLIKĀCIJAS

2.1. Projekti

Sadalītas un heterogēnas imitācijas modelēšanas vienkāršotas komunikācijas vides versijas ir pārbaudītas daudzveidīgos lietojumos un dažādos nacionālos un starptautiskos projektos.

1. Nr. 2006/11 "Simulation Tools EXTEND and NetLogo use for Ecosystems Analysis" (Latvijas Izglītības un zinātnes ministrijas finansējums).
2. Nr. 2007/1-17/26 "Communication Environment of Hybrid Simulation Systems" (Latvijas Izglītības un zinātnes ministrijas finansējums).
3. No.2DP/2.1.1.2.0/10/APIA/VIAA/001 "Support for preparation of IST FP7 STRE project "Simulation Highway"" (Eiropas Reģionālās Attīstības Fonda finansējums).
4. FP7 No.287119 FUPOL „Future Policy Modelling" (Eiropas Komisijas finansējums).
5. FP7 FLAG-ERA FuturICT 2.0 (2017-2020) "Large scale experiments and simulations for the second generation of FuturICT" (Eiropas Komisijas finansējums).

Pētījumu rezultāti tika validēti un izmantoti studiju kursā "Sociotehnisku sistēmu modelēšana" akadēmiskajā maģistra studiju programmā "Kiberdrošības inženierija" (Rīgas Tehniskā universitāte), kā arī profesionālajā maģistra un doktorantūras studiju programmā "Sociotehnisku sistēmu modelēšana" (Vidzemes Augstskola).

2.2. Konferences

Par vienkāršotas komunikācijas vides izstrādes problemātiku ir sagatavoti un sniegti ziņojumi 16 konferencēs no 2006.-2024.gadam:

1. 6th WSEAS International Conference on Applied Computer Science (ACS'06), December 16-18, Puerto de la Cruz, Tenerife, Spain, 2006.
2. 3rd International Scientific Conference "Information Society and Modern Business", September 21-22, Ventspils, Latvia, 2007.
3. 6th WSEAS International Conference on Systems Science and Simulation in Engineering, November 21-23, Venice, Italy, 2007.
4. 4th WSEAS/IASME International Conference on Educational Technologies (EDUTE'08), October 26-28, Corfu, Greece, 2008.
5. International Conference of Modelling and Simulation in Engineering, Economics and Management (MS'10), July 15-17, Barcelona, Spain, 2010.
6. 13th WSEAS International Conference on Automatic Control, Modelling & Simulation (ACMOS '11), May 27-29, Lanzarote, Canary Islands, Spain, 2011.

7. 23rd European Modelling & Simulation Symposium (EMSS 2011), September 12-14, Rome, Italy, 2011.
8. 5th WSEAS World Congress on Applied Computing Conference (ACC '12), May 2-4, Faro, Portugal, 2012.
9. CBU International Conference, April 4-17, Prague, Czech Republic, 2013.
10. 9th International Workshop on Enterprise and Organizational Modeling and Simulation (EOMAS 2013), June 17, Valencia, Spain, 2013.
11. 26th European Modelling & Simulation Symposium (EMSS 2014), September 10-12, Bordeaux, France, 2014.
12. World Conference on Information Systems and Technologies (WorldCIST'15), April 1-3, Ponta Delgada, São Miguel, Azores, Portugal, 2015.
13. 27th European Modelling & Simulation Symposium (EMSS 2015), September 21-23, Bergeggi, Italy, 2015.
14. City Planning and Urban Design Conference (CPUD'16), 07-09 April, Istanbul, 2016.
15. 61st International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS'2020), October 15-16, Riga, Latvia, 2020.
16. 36th European Modeling & Simulation Symposium, September 18-20, Santa Cruz de Tenerife, Spain, 2024.

2.3. Publikācijas

Zinātnes doktora grāda pretendentam Artim Aizstrautam (Scopus ID: 54413897200) ir 30 zinātniskas publikācijas, no kurām 18 ir iekļautas Scopus (Artis Siliņš – 2, Artis Aizstrauts – 16). Tematiski vienoto publikāciju kopā ir iekļautas turpmāk minētās 13 publikācijas (skat. arī B. pielikums), kuru kopijas ir pievienotas. Pēdējā publikācija ir izvilkums no promocijas darba kopsavilkuma. Arta Aizstrauta ieguldījums publikācijās ir vismaz 90%, un korespondējošie līdzautori neiebilst pret šādu sadalījumu, par ko ir pievienots atsevišķs apliecinājums. Kopējais citējumu skaits Scopus ir 140. Hirša indekss Scopus ir 5 (Oktobris 2024).

1. Ginters, E., & Silins, A. (2007). Multi-level approach for environmental systems modelling in the Ligatne Natural Trails. *WSEAS Transactions on Systems*, 6(4), 795-801. Izgūts no https://www.researchgate.net/publication/297118169_Multi-level_approach_for_environmental_systems_modelling_in_the_Ligatne_Natural_Trails
2. Ginters, E., Silins, A., & Andrusaitis, J. (2007). Communication in Distributed Simulation Environment. In *ICOSSSE'07: Proceedings of the 6th WSEAS international conference on System science and simulation in engineering* (pp. 217-221). Stevens Point: World Scientific and Engineering Academy and Society. doi: 10.5555/1974442.1974475 Izgūts no http://www.wseas.us/e-library/conferences/2007venice/papers/600-139.pdf

3. Ginters, E., & Silins, A. (2008a). Exchange Mechanisms in Distributed Simulation of Sociotechnical Systems. In *Information Society and Modern Business. Proceedings of 3rd International conference, 21-22 September 2007, Ventspils* (p. 362). Ventspils: Ventspils University College. Izgūts no https://doi.org/10.5281/zenodo.13925129
4. Ginters, E., & Silins, A. (2008b). Simulation Data Exchange in Distributed E-learning Environment. In *Proceedings of the 4th WSEAS/IASME International Conference on Education Technologies (EDUTE'08)* (pp. 138-143). Corfu: WSEAS. Izgūts no http://www.wseas.us/e-library/conferences/2008/corfu/edute/edute23.pdf
5. Silins, A., Ginters, E., & Aizstrauta, D. (2010). Easy Communication Environment for Distributed Simulation. In *Computational Intelligence in Business and Economics* (pp. 91-98). https://doi.org/10.1142/9789814324441_0014. Izgūts no https://www.researchgate.net/publication/282288675_Easy_Communication_Environment_for_Distributed_Simulation#fullTextFileContent
6. Ginters, E., Sakne, I., Lauberte, I., Aizstrauts, A., Dreija, G., Aquilar Chinea, R.-M., Merkuryev, Y., Novitsky, L., & Grundspenkis, J. (2011). Simulation Highway – Direct Access Intelligent Cloud Simulator. In *Proceedings of 23rd European Modelling & Simulation Symposium (EMSS 2011)* (pp. 62-72). Izgūts no https://doi.org/10.5281/zenodo.13925109
7. Aizstrauts, A., Ginters, E. & Aizstrauta, D. (2011). Easy Communication Approach for Data Exchange in Distributed Simulation Environment. In *Proceedings of the 13th WSEAS international conference on Automatic control, modelling & simulation (ACMOS'11)* (pp. 34-38). Stevens Point: World Scientific and Engineering Academy and Society (WSEAS). Izgūts no http://www.wseas.us/e-library/conferences/2011/Lanzarote/ACMOS/ACMOS-04.pdf
8. Aizstrauts, A., Ginters, E., Aizstrauta, D., & Sonntagbauer, P. (2012). Easy Communication Environment on the Cloud as Distributed Simulation Infrastructure. In *Proceedings of the 5th WSEAS congress on Applied Computing conference, and Proceedings of the 1st international conference on Biologically Inspired Computation (BICA'12)* (pp. 173-178). Stevens Point, Wisconsin: World Scientific and Engineering Academy and Society (WSEAS). Izgūts no http://www.wseas.us/e-library/conferences/2012/Algarve/BICA/BICA-29.pdf
9. Aizstrauts, A., Ginters, E., Lauberte, I., & Piera Eroles, M.A. (2013). Multi-level Architecture on Web Services Based Policy Domain Use Cases Simulator. In *Lecture Notes in Business Information Processing* (vol 153, pp.130-146). Berlin: Springer. doi:10.1007/978-3-642-41638-5_9. Izgūts no https://www.researchgate.net/publication/282289690_Multi-level_Architecture_on_Web_Services_Based_Policy_Domain_Use_Cases_Simulator#fullTextFileContent

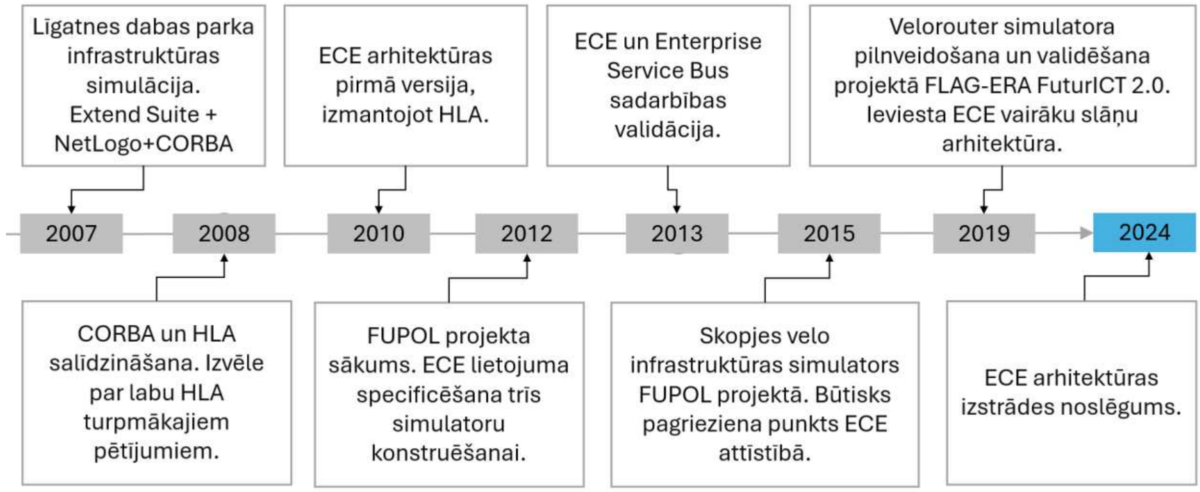
10. Aizstrauts, A., Ginters, E., Baltruks, M., & Gusev, M. (2015). Architecture for Distributed Simulation Environment. *Procedia Computer Science*, *43*, pp. 18-25. doi: 10.1016/j.procs.2014.12.004. Izgūts no https://www.sciencedirect.com/science/article/pii/S1877050914015725
11. Ginters, E., Aizstrauts, A., Baltruks, M., Merkuryev, Y., Novickis, L., Grundspenkis, J., & Grabis, J. (2016). VeloRouter - Technology for Urban Transport Intermodal Sustainability. In *City Planning and Urban Design Conference, 07-09 April (CPUD'16)* (pp. 211-220). Istanbul: DAKAM. ISBN 978-605-9207-21-8. Izgūts no https://doi.org/10.5281/zenodo.13925061
12. Aizstrauts, A., Burkhardt, D., Ginters, E., & Nazemi, K. (2020). On Microservice Architecture Based Communication Environment for Cycling Map Developing and Maintenance Simulator. In *2020 61st International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS)* (pp. 1-4). doi: 10.1109/ITMS51158.2020.9259299. Izgūts no https://www.researchgate.net/publication/347072223_On_Microservice_Architecture_Based_Communication_Environment_for_Cycling_Map_Developing_and_Maintenance_Simulator#fullTextFileContent
13. Aizstrauts, A. & Ginters, E. (2024). Easy Communication Environment for Heterogeneous and Distributed Simulation Models Design. In *2024 36th European Modeling & Simulation Symposium* (p. 1-13). Tenerife. doi: 10.46354/i3m.2024.emss.010. Izgūts no https://www.cal-tek.eu/proceedings/i3m/2024/emss/010/pdf.pdf

3. PROMOCIJAS DARBA REZULTĀTI

3.1. Vienkāršotas komunikācijas risinājuma attīstības posmi

Vienkāršotas komunikācijas vides (ECE) attīstības laika līnija (skat. 1.attēls) ilgst vairāk nekā 15 gadus, kuri ir pagājuši iespējami efektīvāka risinājuma meklējumos, vienlaikus cerot, ka informācijas tehnoloģiju produktu klāstā parādīsies tehnoloģija, kas atrisinās sadalītu un nehomogēnu imitācijas modeļu komunikācijas problēmas.

2007. gadā autors veica pētījumu Līgatnes dabas parkā (Ginters, Silins, 2007). Līgatnes dabas takas ir labiekārtotu taku kopums meža ainavā Gaujas Nacionālajā parka. Tur, savā dabiskajā, bet ierobežotā vidē, dzīvo aļņi, brieži, mežacūkas, vilki, lāči un citi šai vietai tipiski faunas pārstāvji. Lai saudzētu šos dabas resursus, bet arī vienlaikus nodrošinātu apmeklēšanas iespējas, dažādi apskates punkti ir savienoti ar gājējiem paredzētām takām.



1.attēls. ECE izstrādes laika līnija.

Tomēr maršrutos ir iekārtoti arī asfaltēti ceļi un vairākas autostāvvietas. Līdz ar to Līgatnes dabas parkā pa noteiktu maršrutu un noteiktā secībā var pārvietoties arī apmeklētāju automašīnas. Toreiz pētījuma jautājums bija, kāda ir pieļaujamā apmeklētāju radītā noslodze, lai dabas resursiem neradītu neatgriezenisku kaitējumu, un tie spētu atjaunoties noteiktā laika intervālā?

Bija nepieciešams izveidot diskrētu notikumu imitācijas modeli, kas veic autotransporta kustības analīzi, lai saprastu, cik lielas stāvvietas ir nepieciešamas, kā arī, kādai ir jābūt to infrastruktūrai (tualetes, dzērienu automāti, atkritumu urnas, u.tml.). Modelis tika izstrādāts Extend Suite vidē un kalpoja par datu avotu otra, uz aģentiem balstīta modeļa, darbināšanai NetLogo vidē. Šis modelis savukārt novērtēja dabas resursa reģeneratīvo kapacitāti.

Risinājums radīja modeļu savstarpējās savietojamības problēmu, tādēļ tika veikta esošo sadalītās imitāciju modelēšanas komunikācijas rīku analīze. Rezultātā par izmantojamiem tika atzīti DIS un HLA. Tomēr HLA izmantošana, lai nodrošinātu modeļu sadarbības mehānismu, bija sarežģīta un darbietilpīga. Bija skaidrs, ka sadalītu imitācijas modeļu izstrāde ar HLA nav iespējama bez padziļinātām programmēšanas zināšanām, līdz ar to šāds risinājums vides pētniekiem nav piemērots (Aizstrauts un Ginters, 2024).

Autors turpināja analizēt sadalītās imitāciju modelēšanas komunikācijas rīkus, kas būtu pieejami profesionāļiem, kas nestrādā programmēšanas inženierijā (Ginters, Silins un Andrusaitis, 2007).

Tika īstenota dažādu rīku piemērotības analīze ALSP (*Aggregate Level Simulation Protocol*), CORBA, FIPA (*The Foundation for Physical Intelligent Agents*) un HLA. Īpašu uzmanību raisīja CORBA mehānisms, kura piemērotība Līgatnes modeļa komunikācijas nodrošināšanai tika rūpīgi pārbaudīta. Tika veikti eksperimenti, kuros CORBA komunikācijas mehānisms ORB (*Object Request Broker*) nodrošināja datu apmaiņu starp Extend Suite un NetLogo modeļiem. Tika izstrādāti imitācijas modeļu paplašinājumi C++ (Extend Suite) un Java (NetLogo). Risinājums atļāva modeļiem komunicēt, tomēr atkal prasīja labas prasmes programmēšanā (Aizstrauts un Ginters, 2024).

2008. gadā turpinājās CORBA ORB mehānisma pielāgošana dažādām sadalītām imitāciju modelēšanas vidēm, tomēr sāka radīt šaubas CORBA nosacīti zemā veiktspēja, kas atstāja iespaidu uz modelēšanas kvalitāti (Ginters un Silins, 2008a).

Autors (Ginters un Silins, 2008b) atgriezās pie CORBA un HLA veiktspējas mērījumiem. Aizdomas apstiprinājās, kad, palielinoties ziņojuma pakešu garumam, būtiski pieauga HLA pārsvars pār CORBA rezultātiem. Tā kā HLA veiktspēju var uzskatīt par kritisku indikatoru, tad izvēlēties vēl kaut ko lēnāku par HLA nebūtu saprātīgi, jo tas turpmāk būtiski ierobežotu komunikācijas rīka ieviešanas iespējas. Tika izlemts, ka CORBA nav pietiekami piemērota lietošanai sadalītos imitāciju modeļos, un tika turpināti HLA adaptēšanas iespēju pētījumi.

2010.gadā autors publicēja risinājumu (Silins, Ginters un Aizstrauta, 2010), kas atviegloja piekļuvi HLA federācijām. Risinājums balstījās uz komunikācijas adapteri, kas nodrošināja savienojumu starp imitācijas modeli un HLA. Adapteris bija daļa no HLA vides un interpretēja HLA datus imitācijas modelim saprotamā formātā. Komunikācijas adapteris "klausījās" modeļus un akumulēja to ziņojumus datu krātuvē. Modelis, kam šie dati bija vajadzīgi, varēja iegūt krātuves datus pēc pieprasījuma. Datu krātuve vienlaikus nodrošināja laika sinhronizācijas iespējas. Katrai imitāciju modelēšanas videi vajadzēja bibliotēku, lai mijiedarbotos ar adapteri. Sadarbība starp imitācijas rīku un adapteri iniciēja datu pieprasījumi XML formātā. Saskaņā ar modeļa parametriem adapteris ģenerēja FOM (*Federation Object Model*), kas atļāva to savienot ar HLA federācijām. Šādam adapterim bija ne tikai jāspēj strādāt ar dažāda veida mainīgajiem no atšķirīgiem modeļiem, bet tam bija arī jābūt pietiekami vienkārši pārvaldāmam. Šīs prasības izpilde izrādījās sarežģītāka, nekā sākotnēji šķita, jo tas savukārt radīja izmaiņas attiecībā uz savietojamību ar HLA. Tika radīta pirmā ECE beta versija.

2011. gadā turpinājās ECE beta versijas pilnveide, galvenokārt papildinot komunikācijas bibliotēkas. Tika izveidots komunikācijas vārtejas mehānisms (*Communication Gateway*), kas varēja nodrošināt datu apmaiņu starp komunikācijas adapteriem pirms pieslēgšanās HLA videi (Aizstrauts, Ginters un Aizstrauta, 2011). Uzlabojumi samazināja modelētāja darba apjomu, izveidojot HLA balstītus sadalītās imitācijas modeļus (Aizstrauts un Ginters, 2024).

Neraugoties uz pirmo ECE versiju, kas bija balstītas uz HLA, verifikāciju un validāciju, turpinājās piemērotākas komunikācijas vides, arhitektūru, protokolu un ziņojumu datu formātu meklējumi (Ginters et al., 2011).

Autors izstrādāja Simulation Highway konceptu kā vairāku slāņu atvērtu arhitektūru, iedzīvinot ISO OSI 7498 (*International Organization for Standardization Open Systems Interconnection*) principus sadalītā imitāciju modelēšanā. Simulation Highway katra modelēšanas uzdevuma izpildi nodrošināja simulācijas elementu ķēde. Prasības pēc specifiskām programmēšanas zināšanām samazinājās, jo modelēšanas pieprasījumu formēšanai, saskaņā ar klients-serveris pieeju, varēja izmantot augsta līmeņa valodas, piemēram SimAL un SimQL. Tomēr komunikācijas pamata mehānisms joprojām bija HLA. Turklāt Simulation Highway implementācija prasīja ievērojamus datu apstrādes resursus. Pēc Simulation Highway ilgtspējas pašnovērtējuma, autors šo ideju atmeta (Aizstrauts un Ginters, 2024).

2012.gadā Eiropas Komisijas 7. ietvara programmas ietvaros tika uzsākts FUPOL projekts (No. 287119). Projekta ietvaros bija jāveic vairāki modelēšanas pētījumi. Proti, publiskā parka plānošana un zonējums Horvātijas galvaspilsētā Zagrebā un tūrismam paredzētu pakalpojumu noslodzes prognozēšana un veloceliņu maršrutu plānojums Ziemeļmaķedonijas galvaspilsētā Skopjē.

FUPOL projekts ir uzlūkojams kā nākamais ECE attīstības posms (Aizstrauts et al., 2012), jo tas deva iespēju validēt iepriekš izstrādātās ECE versijas kvalitāti. Lai ieviestu un testētu atbildes reakciju laika intervālus teritoriāli sadalītā modelī, ECE bija jāizvieto EC2 (*Amazon Elastic Compute Cloud*) virtuālās skaitļošanas vidē. Šajā un nākamajās ECE versijās tika pārtraukta HLA izmantošana un paplašināta iepriekš izveidotās komunikācijas vārtejas funkcionalitāte.

Viens no FUPOL projekta pamata uzdevumiem bija izveidot simulācijā balstītu atbalsta mehānismu rīcībpolitikas lēmumu pieņemšanai. Konsorcijs veica vairāk nekā 60 unikālu un problēmorientētu imitācijas modelēšanas ietvaru analīzi. Tās rezultāti apstiprināja agrākās aizdomas, ka problēmorientēti imitāciju modelēšanas risinājumi ir slēgti un tiem ir grūti pievienot ārējus modeļus. Diemžēl slēgtām vidēm parasti ir ierobežota funkcionalitāte, kas savukārt apgrūtina heterogēnu sadalītu imitācijas modeļu izstrādāšanu.

Tika izstrādāts FUPOL simulatora pamata koncepcija (Aizstrauts et al., 2013), kur augšējā specifikāciju slānī tika izmantotas nestriktās kognitīvās kartes (*Fuzzy Cognitive Maps)* un krāsainie Petri tīkli. Imitācijas modeļi tika īstenoti RePast Symphony. Savukārt STELLA, sistēmu dinamikas imitāciju modelēšanas vide, tika lietota, lai modelētu sistēmas iespējamās izmaiņas laikā. Datu apstrādei un analīzei tika izmantots Python, bet datu glabāšanai - PostgreSQL datubāze. Lai uzlabotu simulāciju rezultātu vizualizāciju un nodrošinātu labāku modelēšanas sesiju vadību, tika izmantota NetLogo vide. Datu apmaiņa FUPOL simulatorā notika ar ECE palīdzību, bet, lai nodrošinātu savietojamību ar citām FUPOL apakšsistēmām, bija jāizveido ECE savienojums ar FUPOL centrālo kopni (*Enterprise Service Bus*). Kopumā var sacīt, ka FUPOL projekts bija ļoti noderīgs ECE koncepta validācijai.

Viens no FUPOL lietojumiem, kas atļāva validēt ECE, bija Skopjes velo infrastruktūras simulators (Aizstrauts et al., 2015). Simulatora pamata uzdevums bija velobraucēju maršruta izvēles iespēju modelēšana, atkarībā no katra veloceliņa posma noslodzes, veloceliņa seguma, pieejamās infrastruktūras un meteoroloģiskās prognozes. ECE šajā gadījumā vajadzēja nodrošināt savietojamību starp NetLogo, RePast un Python vidēm.

Tika veikta vairāku ziņojumu komunikācijas starpniekmehānismu (*Message Brokers*) analīze, vadoties pēc veiktspējas, ziņojumu garuma un protokolu parametriem. Tika izvēlēts RabbitMQ, kas attiecīgi turpmāk tika izmantots kā ECE komponente. Imitāciju modelis komunicēja ar ziņojumu komunikācijas starpniekmehānismu, izmantojot AMQP protokolu.

No 2016.gada ECE vairs netika izmantoti mākoņdatošanas servisi, jo tie nenodrošināja garantētu reakcijas laiku. FUPOL validācijas rezultāti papildināja ECE prasību kopumu – svarīgas ir ne tikai modelētāja ērtības, bet arī komunikāciju vides universālums un drošums.

Turpmāk ECE tika validēta veloceliņu maršrutu plānošanas rīka prototipa Velorouter konstruēšanā (Aizstrauts et al., 2020). Pētījumu iniciēja iepriekšējie FUPOL projekta rezultāti un Eiropas komisijas 7. ietvara FLAG-ERA FuturICT 2.0 (2017-2020) projekts. Velorouter lietotājiem bija iespēja, izstrādāt savus maršrutus, kā arī ieteikt tos pašvaldībai, izmantojot reglamentētu ziņojumu apmaiņas (*ticketing*) mehānismu.

Projektā pirmo reizi tika prezentēta ECE vairāku slāņu arhitektūra un noteikta katra slāņa funkcionalitāte. Pamatojoties uz universāluma un saderības prasībām, ECE komunikācijas un datu apmaiņas mehānismā tika iekļauti JSON (*JavaScript Object Notation*) datu formāts un AMQP protokols, atsakoties no turpmākas XML izmantošanas. Tika respektēta galvenā ECE izstrādes prasība, nodrošināt iespējas nozares speciālistam bez specifiskām prasmēm programmēšanas inženierijā, izveidot un darbināt heterogēnus un sadalītus imitāciju modeļus, kurus var izmantot attīstības scenāriju validācijai dažādās nozarēs un daudzveidīgos lietojumos (Aizstrauts un Ginters, 2024).

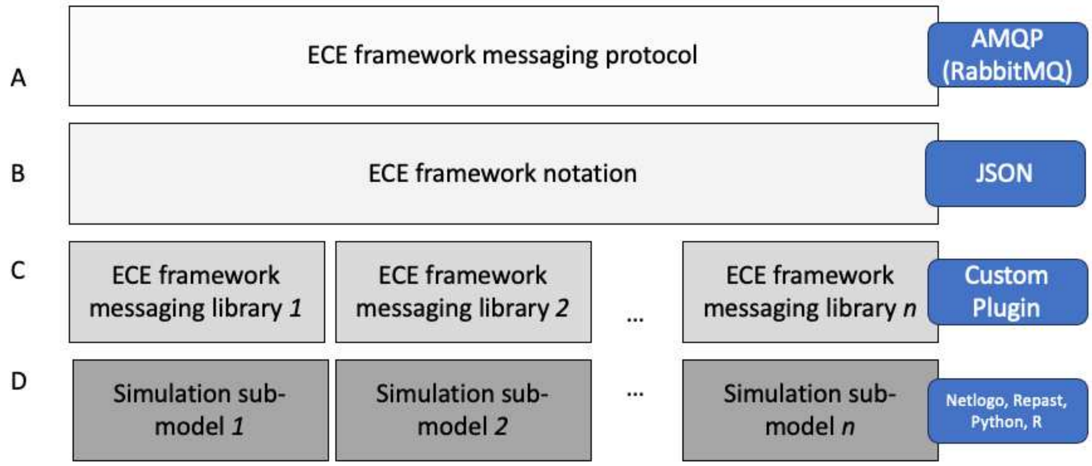
Turpmāk promocijas darbā tiek izskaidroti ECE arhitektūras strukturālais modelis un funkcionalitāte.

3.2. Vairākslāņu vienkāršotas komunikācijas vides strukturāls modelis

Vienkāršotas komunikācijas vide (ECE) ir daudzpusīgs un modelētājam draudzīgs komunikācijas rīks. Tā galvenais uzdevums ir atvieglot saziņu starp heterogēniem simulācijas modeļiem. Turklāt tam ir jābūt lietotājam draudzīgam – pat pret tiem sadalītu imitācijas modeļu izstrādātājiem, kuru zināšanas programmēšanas inženierijā ir minimālas.

ECE strukturālajam vairāku slāņu modelim ir atvērtas sistēmas arhitektūra, kuras pamatā ir OSI ISO 7498 (ISO/IEC 7498-1:1994(E), 2024; Ginters et al., 2011) un Dijkstra mašīnas (Dijkstra, 1968) pamata idejas. Tomēr idejas būtība seko no plašāka Zimmermann (1980) atvērto sistēmu vadlīniju apraksta.

ECE strukturālais modelis (1) (skat. 2. attēls) sastāv no četriem slāņiem, no kuriem katrs ir atbildīgs par konkrētiem un funkcionāliem uzdevumiem – A-slānis: ECE ziņojumapmaiņas protokols (A), B-slānis: ECE notācijas slānis (B), C-slānis: ECE ziņojumapmaiņas bibliotēkas (C) un D-slānis: Imitācijas modeļu kopa (D) (Aizstrauts un Ginters, 2024).



2.attēls. ECE strukturālais modelis.

ECE strukturālo ietvaru nosaka:

$$ECE = \langle A, B, C, D \rangle \quad (1)$$

kur A, B, C un D – arhitektūras funkcionālie slāņi.

Tomēr ECE slāņu steku var aprakstīt kā atvērtu virtuālu mašīnu kopu

$$ECE = \bigcup_{j=1}^{J} OVM_j \tag{2}$$

kur J -virtuālo mašīnu skaits ECE vidē.

Savukārt katra virtuālā mašīna

$$OVM_j = \langle A_j, B_j, C_j, D_j \rangle \tag{3}$$

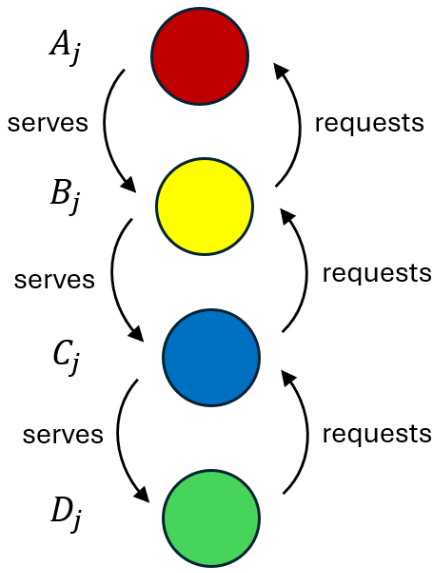
kur A_j, B_j, C_j, D_j ir atbilstošas j-virtuālās mašīnas servisu nodrošinošs slānis.

Atvērtas virtuālas mašīnas funkcionalitāti raksturo cēloņsakarību diagramma CLD (*Causal Loop Diagram*) (skat. 3. attēls).

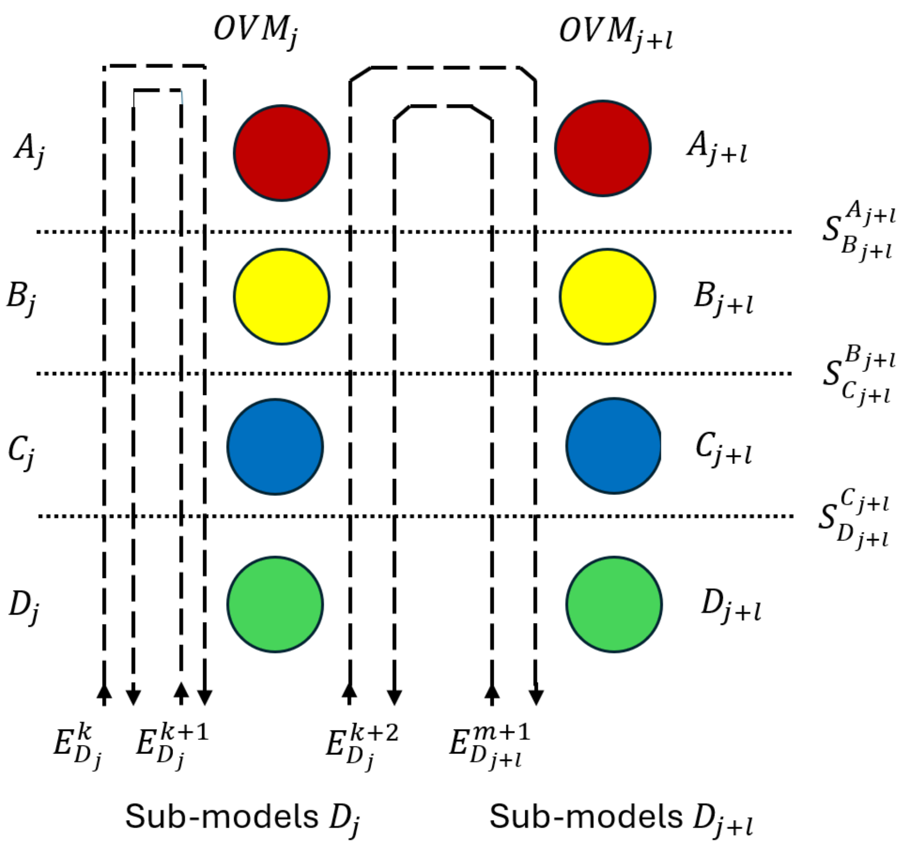
Atbilstoši CLD diagrammai, katrs slānis sniedz servisa pakalpojumu otram blakus esošajam slānim.

D-slānī izvietoto imitācijas modeļu mijiedarbība sadalītā imitāciju modelī, kas ir izvietots uz vienas vai vairākām OVM_j, ir parādīta 3.attēlā.

4.attēlā ir redzamas divas virtuālās mašīnas OVM_j un OVM_{j+l} . Abas mašīnas veido ECE vidi. Uz katras no virtuālajām mašīnām ir izvietota sadalīta imitācijas modeļu kopa $Sub - models\ D_j$ un $Sub - models\ D_{j+l}$.



3.attēls. Atvērtas ECE OVM_j slāņu mijiedarbības CLD.



4.attēls. Imitācijas modeļu kopu mijiedarbības attēlojums.

Katrā D_j ir iespējams sekojošu imitācijas modeļu kopas izvietojums:

- Atsevišķi un patstāvīgi modeļi, kuri faktiski neizmanto ECE pakalpojumus, jo nekomunicē ar citiem imitāciju modeļiem.
- Homogēni un sadalīti modeļi, kuri funkcionē uz vienas virtuālās mašīnas, bet to mijiedarbību nodrošina iekšējais lietotās imitāciju modelēšanas tehnoloģijas komunikācijas mehānisms, lai gan tie var izmantot arī ECE servisus.
- Heterogēni modeļi, kas ir izvietoti uz vienas vai vairākām virtuālām mašīnām un, kuriem ECE izmantošana ir vienīgā sazināšanās iespēja.

Katra ECE j-slāņa pamata uzdevums ir servisa funkcijas S nodrošināšana kaimiņu slānim, apstrādājot kaimiņu slāņa izejas O datu kopu. ECE translācijas mehānismu raksturo turpmāk attēlotās analītiskās sakarības:

$$\begin{cases} I_{A_{j+l}} = S_{B_{j+l}}^{A_{j+l}}(O_{B_{j+l}}) \\ I_{B_{j+l}} = S_{C_{j+l}}^{B_{j+l}}(O_{C_{j+l}}) \\ I_{C_{j+l}} = S_{D_{j+l}}^{C_{j+l}}(O_{D_{j+l}}) \end{cases} \quad (4)$$

kur $I_{C_{j+l}}$ – slāņa C_{j+l} ieeja, $O_{D_{j+l}}$– slāņa D_{j+l} izeja, bet $S_{D_{j+l}}^{C_{j+l}}$ – slāņa C_{j+l} servisa funkcija slānim D_{j+l} (piekļuve konkrētas imitāciju modelēšanas tehnoloģijas ECE bibliotēkām),

un $I_{B_{j+l}}$ – slāņa B_{j+l} ieeja, $O_{C_{j+l}}$– slāņa C_{j+l} izeja, bet $S_{C_{j+l}}^{B_{j+l}}$ – slāņa B_{j+l} servisa funkcija slānim C_{j+l} (piekļuve datu formātu pārveidošanai),

bet $I_{A_{j+l}}$ – slāņa A_{j+l} ieeja, $O_{B_{j+l}}$– slāņa B_{j+l} izeja, bet $S_{B_{j+l}}^{A_{j+l}}$ – slāņa A_{j+l} servisa funkcija slānim B_{j+l} (piekļuve datu transportēšanai).

Ierobežojumi. Atšķirībā no OSI ISO 7498 koncepta ECE netiek nodrošināta dažādu virtuālo mašīnu tieša sadarbība viena un tā paša slāņa ietvaros, un konkrēti, C_j un C_{j+l} nevar sadarboties bez citu slāņu līdzdalības. Tas ir iespējams tikai ar saišu kopas E līdzdalību.

Modeļu komunikāciju raksturo saišu kopa E. ECE var nodrošināt uz vienas virtuālās mašīnas izvietotu modeļu sadarbību, piemēram, $E_{D_j}^{k}$ un $E_{D_j}^{k+1}$, kā arī uz dažādām virtuālām mašīnām izvietotu $Sub - models\ D_j$ un $Sub - models\ D_{j+l}$ sadarbību, piemēram, $E_{D_j}^{k+2}$ un $E_{D_{j+l}}^{m+1}$.

Tas nozīmē, ka sadalītā un heterogēnā imitāciju modelī sadarbību starp modeļiem D_j^{k+2} un D_{j+l}^{m+1}, kas ir izvietoti uz dažādām virtuālām mašīnām OVM_j un OVM_{j+l}, nodrošina servisu ķēde $E_{D_j}^{k+2}$

$$E_{D_j}^{k+2} = \left\{D_j^{k+2}, S_{D_j}^{C_j}, S_{C_j}^{B_j}, S_{B_j}^{A_j}, S_{B_{j+l}}^{A_{j+l}}, S_{C_{j+l}}^{B_{j+l}}, S_{D_{j+l}}^{C_{j+l}}, D_{j+l}^{m+1}\right\} \quad (5)$$

Katrs servisu ķēdē iekļautais slānis A, B, C vai D izpilda atbilstošus un turpmāk aprakstītus funkcionālus uzdevumus.

A – slānis: ECE ziņojumapmaiņas protokols (*ECE framework messaging protocol*). Protokols sastāv no divām komponentēm – ziņojuma pārsūtīšanas un ziņojuma nesēja formāta.

Ziņojumu pārsūtīšanai ECE izmanto AMQP protokolu. Tas ir atvērta standarta lietojumprogrammu slāņa protokols uz ziņojumiem orientētai starpprogrammatūrai. AMQP pamatiezīmes ir ziņojumu orientācija, rindas, maršrutēšana (*point-to-point* un *publish-and-subscribe*) un drošums (Naik, 2017). AMQP piedāvā divus sākotnējos pakalpojumu kvalitātes QoS (*Quality of Service*) līmeņus ziņojumu piegādei – "Ne vairāk kā vienreiz" un "Vismaz vienu reizi" (Aizstrauts un Ginters, 2024).

ECE izmanto pakalpojuma līmeni "Ne vairāk kā vienreiz", bet adresātam nav jāapstiprina ziņojuma saņemšana. Datu uzglabāšanas funkcionalitāte tiek realizēta ECE arhitektūras C-slānī. ECE ziņojumos tiek izmantots JSON datu formāts.

ECE izmanto AMQP un JSON, lai nodrošinātu komunikācijas arhitektūru, kas ir droša un viegli uzturama. AMQP nodrošina, ka ziņojums tiek piegādāts pareizajam klientam/abonentam, kā arī garantē protokola noturību. JSON ļauj pārsūtīt sarežģītas struktūras datus, kurus spēj analizēt un saprast vairums programmēšanas valodu.

B – slānis: ECE notācijas slānis (*ECE framework messaging notation*). Notācijas slānis ir viens no augšējiem līmeņiem ECE komunikācijas arhitektūrā. Šis slānis nodrošina, ka katrs mezgls (imitācijas modelis) saprot viens otru un runā vienā valodā. ECE notācija ir balstīta uz JavaScript objektu notācijas (JSON) datu formātu. Ziņojums sastāv no trīs laukiem (skat. 1.tabula).

1.tabula. ECE starpslāņu ziņojuma formāts (Aizstrauts un Ginters, 2024).

Lauks	Apraksts
tick	Imitāciju modelēšanas laiks, kad šis ziņojums ir nosūtīts.
datatype	Datu tips.
data	Pārraidāmie dati.

C – slānis: ECE ziņojumapmaiņas bibliotēkas (*ECE framework messaging library*). Ziņojumapmaiņas bibliotēka ir speciāli veidota programmatūra, kas tiek integrēta modelēšanas rīku komplektā kā paplašinājums vai spraudnis. ECE arhitektūras abstrakcijas slāņi ir atvērti, tāpēc ir iespējams izstrādāt šādus paplašinājumus/spraudņus dažādiem imitāciju modelēšanas rīkiem.

ECE ziņojumapmaiņas bibliotēka (C-slānis) ietver pamata funkcijas, kas iespējo savietojamību ar D-slāni (skat. 2.tabula).

2.tabula. ECE ziņojumapmaiņas bibliotēkas piemērs (C-slānis) (Aizstrauts un Ginters, 2024).

Funkcija	Apraksts
init	Inicializēt pieslēgšanos pie AMQP servera.
close	Slēgt pieslēgšanos pie AMQP servera.
isConnected	Atgriež stāvokļa raksturojumu savienojumam ar AMQP servera.
Subscribe	Parakstīšanās uz komunikācijas tēmu. Turpmāk tiks saņemti un saglabāti visi ziņojumi, kas būs piekritīgi šai tēmai.

Unsubscribe	Parakstīšanās atcelšana. Turpmāk tiks ignorēti visi ziņojumi, kas būs piekritīgi šai tēmai, kā arī tiek iznīcināti jau agrāk saņemtie ziņojumi.
sendString	Nosūta String tipa ziņojumu noteiktai tēmai.
sendInteger	Nosūta Integer tipa ziņojumu noteiktai tēmai.
sendDouble	Nosūta Double tipa ziņojumu noteiktai tēmai.
sendBoolean	Nosūta Boolean tipa ziņojumu noteiktai tēmai.
getStringData	Atgriež String tipa datus par noteikto tēmu un konkrētā laika momentā (*tick*).
getIntegerData	Atgriež Integer tipa datus par noteikto tēmu un konkrētā laika momentā (*tick*).
getDoubleData	Atgriež Double tipa datus par noteikto tēmu un konkrētā laika momentā (*tick*).
getBooleanData	Atgriež Boolean tipa datus par noteikto tēmu un konkrētā laika momentā (*tick*).
getLastStringData	Atgriež String tipa datus pēdējā laika momentā (*latest tick*) par noteikto tēmu.
getLastIntegerData	Atgriež Integer tipa datus pēdējā laika momentā (*latest tick*) par noteikto tēmu.
getLastDoubleData	Atgriež Double tipa datus pēdējā laika momentā (*latest tick*) par noteikto tēmu.
getLastBooleanData	Atgriež Boolean tipa datus pēdējā laika momentā (*latest tick*) par noteikto tēmu.
getLastTick	Atgriež pēdējā laika momentu (*latest tick*) par noteikto tēmu.

D – slānis: Imitācijas modeļu kopa (*Simulation sub-model*). Apakšmodelis vai imitācijas modelis ir jebkura programma vai algoritms, kas saņem vai nosūta jebkādus datus. Šāds apakšmodelis izmanto C-slāņa paplašinājumu/spraudņu API (*Application Programming Interface*), lai nosūtītu vai saņemtu datus no citiem modeļiem. Lai nodrošinātu divu vai vairāku simulācijas modeļu (D-slānis) savstarpēju sadarbību un datu apmaiņu sesijas laikā, tiek izveidota ECE slāņu servisa ķēde *E* (skat. 4. attēls).

Turpmāk tiek aprakstīts ECE starpslāņu mijiedarbības algoritms.

3.3. Funkcionalitātes apraksts

Vienkāršotas komunikāciju vides (ECE) A, B, C un D slāņu sadarbība ir attēlota BPMN2 diagrammā (skat. A. pielikums). Diagrammā ir attēlota sadarbība starp diviem simulācijas modeļiem (D-slānis). Procesi, kas notiek katrā slānī imitāciju modelēšanas sesijas laikā, notiek paralēli. Procesi apmainās ar ziņojumiem un signāliem, un process, kas notiek vienā slānī, var ierosināt un/vai ietekmēt procesu citā slānī (Aizstrauts un Ginters, 2024).

ECE inicializācijas darbība tiek aktivizēta D-slānī, modelim nosūtot signālu uz C-slāņa atbilstošo bibliotēku. Ja modelis vēlas nosūtīt datus, tad tiek izsaukta atbilstošā bibliotēkas komanda C-slānī. Ja modelim nav nosūtāmu datu, vai arī simulācijas sesija ir beigusies, tad tiek nosūtīts signāls, kas aptur modeļa spraudni C-slānī.

C-slāņa procesus aktivizē signāls, kas tiek saņemts no D-slāņa modeļa. Saņemot signālu par simulācijas sesijas uzsākšanu, C-slānis nodrošina savienojumu ar AMQP serveri. Kad tiek saņemts datu ziņojums no D-slāņa, no tā tiek izveidots neapstrādātu datu objekts, kas tiek pārsūtīts uz B-slāni. Datu objekta saņemšana aktivizē B-slāni, kurā objekts tiek pārveidots ECE notācijas JSON datu formātā, bet pēc tam kā ziņojums nosūtīts atpakaļ uz C-slāni. Turpmāk ziņojums JSON formātā tiek nosūtīts uz AMQP serveri A-slānī, atbilstoši noteiktajai tēmai (*AMQP topic exchange*). Ja C-slānis nav saņēmis signālu par simulācijas sesijas beigām, tad datu nosūtīšanas process turpinās, pretējā gadījumā savienojums ar AMQP serveri tiek pārtraukts. A-slāni aktivizē datu ziņojums, kas saņemts no C-slāņa. Ziņojums tiek ierakstīts AMQP krātuves rindā, kas atbilst noteiktai datu tēmai. Pēc tam AMQP serveris pārraida rindā ievietoto datu ziņojumu.

Nosūtītais ziņojums turpmāk nonāks pie modeļa, kuram tas ir nepieciešams jeb kurš to ir abonējis. Otra modeļa C-slānī saņemtais AMQP ziņojums tiek pārvērsts iekšējos neapstrādātos datos un atpazīts pirms atkal tas tiks pārvērsts JSON formātā B-slānī un nosūtīts uz C-slāni. C-slānī ziņojumu saglabā attiecīgās tēmas vietējā krātuvē. Ja šo procesu nepārtrauc no D-slāņa saņemts simulācijas sesijas beigu signāls, tas tiek sadalīts divos virzienos – vaicājums pēc datiem D-slānī, kā arī tiek pagarināts konkrētās tēmas abonēšanas laiks. Datu objekts iekšējā formātā tiek nosūtīts uz simulācijas modeli (D-slānis), bet ziņojums JSON formātā nonāk A-slānī abonētās tēmas AMQP krātuvē.

Turpmāk tiks aplūkots ECE lietojuma piemērs-fragments no Skopje velomaršrutu plānošanas simulatora. ECE darbības demonstrēšanas nolūkā šim modeļa fragmentam ir izveidota lietotāja saskarne, kas nav pieejama Skopje modelī.

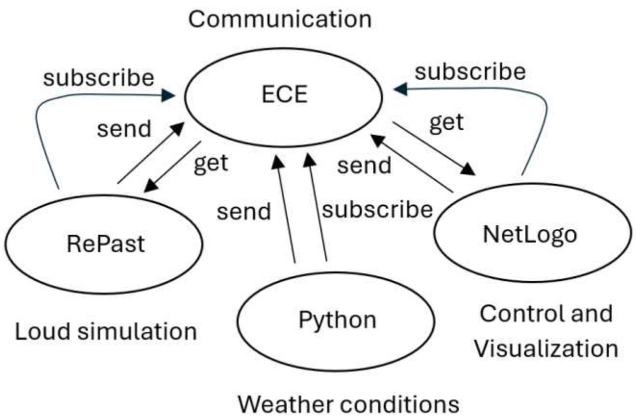
3.4. Lietojums un diskusija

ECE lietojuma piemērs. Tiek aplūkots FUPOL projekta Skopje velomaršrutu plānošanas heterogēnā un sadalītā imitācijas simulatora fragments, kurā komunikāciju nodrošina ECE. Tas nav Skopje velomaršrutu simulators, bet gan tā starpprogrammatūra, kuras mērķis ir demonstrēt ECE lietojumu. Fragmenta uzdevums ir modelēt noslodzi dažādos maršruta segmentos un laika apstākļus, lai velosipēdists varētu izveidot sev piemērotu maršrutu un izvēlēties ceļojuma laiku.

Modeļa konceptkarte (skat. 5. attēls) attēlo simulatora fragmenta objektus, kas nodrošina nepieciešamo funkcionalitāti. Velomaršruta segmentu noslodzes simulāciju nodrošina uz aģentiem balstīts RePast modelis. Laikapstākļu dati tiek ģenerēti Python vidē, savukārt imitācijas modelēšanas vadība un informācijas vizualizācija ir īstenota NetLogo.

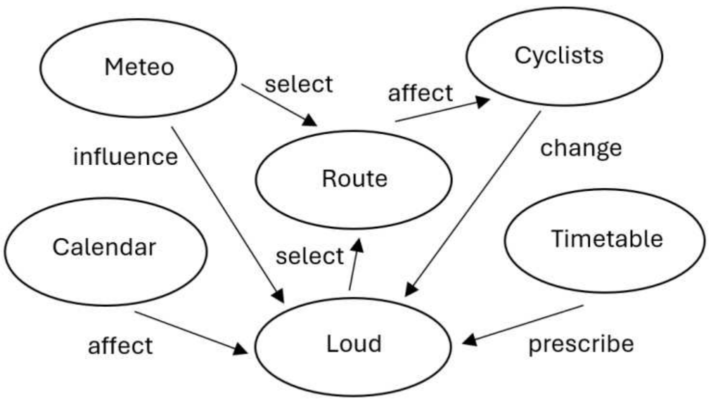
Visi simulatora apakšmodeļi ir savienoti ar ECE, kas nodrošina komunikācijas un datu apmaiņas iespējas. Modeļi un pieprasījumi bibliotēkām veido ECE D-slāni. Katra modeļa specifikācija (D-slānis) ietver komandas, kas nodrošina paziņojumu/datu nosūtīšanu un saņemšanu (abonēt/sūtīt/saņemt).

Modelētāja uzdevums ir iekļaut atbilstošās komandas modeļa programmas D-slāņa kodā. ECE daļas A, B un C ir ārpus modelētāja interešu loka.



5.attēls. Sadalīta imitācijas modeļa konceptkarte.

Cēloņsakarību diagramma (CLD) (sk. 6. attēls) apraksta simulatora fragmenta funkcionalitātes pamatus.



6.attēls. Maršrutu plānošanas funkcionalitātes CLD diagramma.

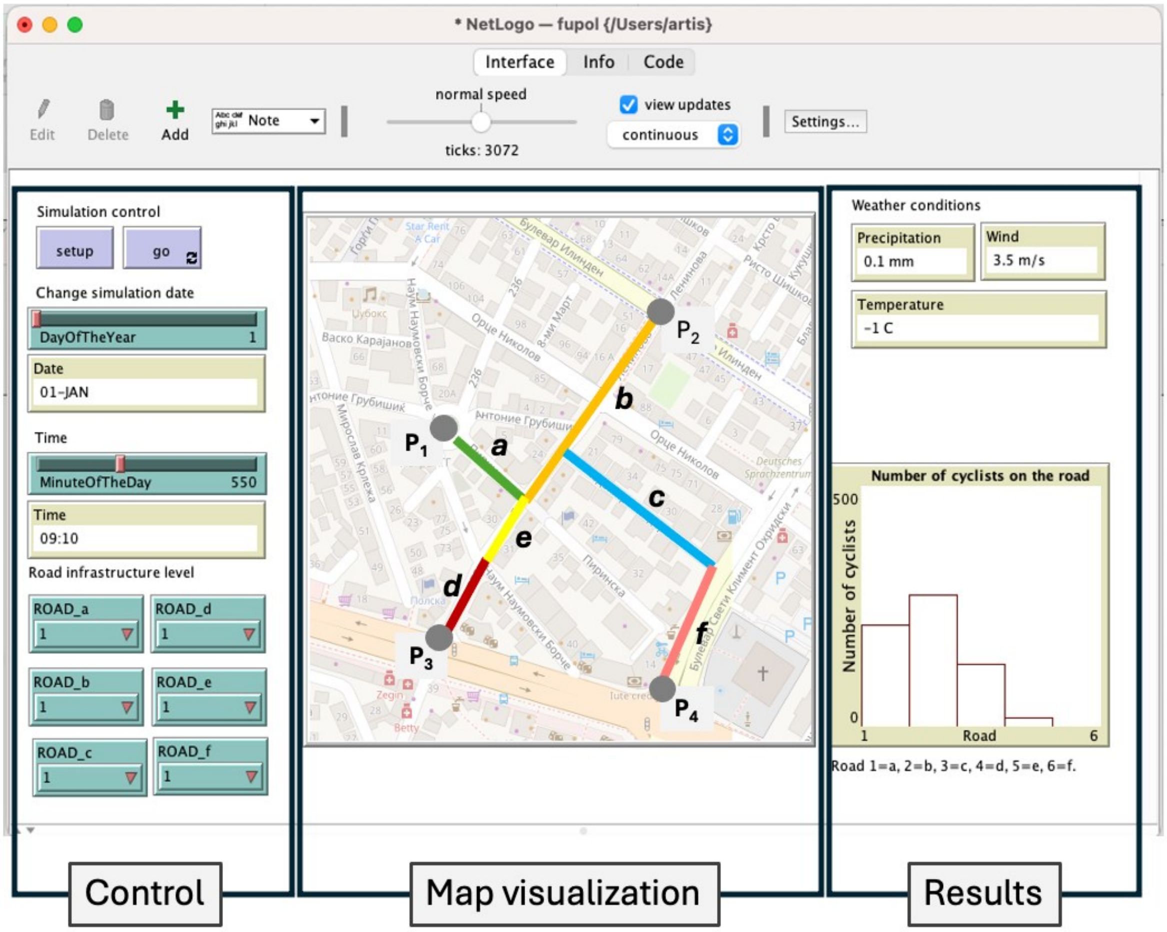
Imitāciju modelis palīdz velosipēdistam izvēlēties atbilstošākos maršruta segmentus un plānot kopējo maršrutu. Piemēram, maršruta segmenta izvēli ietekmē meteoroloģiskie apstākļi, kas var pasliktināt ceļa virsmas kvalitāti. Savukārt sliktos laikapstākļos noslodze maršrutos samazinās, jo samazinās velobraucēju skaits. Maršruta noslogojumu ietekmē gadalaiks (ziema, vasara) un kalendārs (darba dienas vai brīvdienas). Konkrēta maršruta izvēli ietekmē arī tā noslodze, jo velosipēdista profili ir atšķirīgi un tos nosaka atribūtu kopums: prasmes, velosipēda veids, vecums, grupas lielums, bērnu klātbūtne grupā utt. Katrs atlasītais parametrs var izmainīt velosipēdistu skaitu segmentā, kas savukārt noved pie noslodzes izmaiņām. Noslodzes palielināšanās savukārt samazina maršruta izvēles iespējamību turpmākajās iterācijās. Simulācijas sesijas sākotnējie nosacījumi tiek koriģēti ar reāliem noslodzes datiem, veidojot līdzsvarotu sistēmu.

Simulatora vadības panelis (skat. 7.attēls) ir izstrādāts NetLogo, un nodrošina nepieciešamās pārvaldības un vizualizācijas iespējas. Lietotāja saskarnei ir trīs sadaļas – vadības rīki (Control), kartes vizualizācija (Map visualization) un rezultātu panelis (Results) (Aizstrauts un Ginters, 2024).

Iestatīšanas poga inicializē modeļu un ECE sadarbību, tas ir, nosaka globālos mainīgos un abonē ECE sesiju. Turpmāk visas globālo mainīgo vērtības tiks saņemtas no citiem modeļiem, izmantojot ECE. ECE Netlogo spraudnis (C-slānis) asinhroni saņems datus par katru no abonētajām tēmām un saglabās tos lokāli ar laika zīmogu (ziņojuma *tick*) (skat. 8. attēls).

C-slāņa bibliotēkām simulācijas sesijas sākumā ir jātiek pievienotām ECE AMQP serverim. Tas tiek paveikts ar komandu *ece: init,* un šajā gadījumā savienojums ir ar *localhost*.

Poga Go iniciē un aptur simulācijas procesus. Ceļojuma datums un laiks ir manuāli maināmi. Laiks tiek mērīts kā minūšu skaits diennaktī, bet datums ir dienas kārtas skaitlis kalendārajā gadā.



7.attēls. Velomaršrutu izvēles simulatora vadības panelis (Aizstrauts un Ginters, 2024).

Lietotājs var noteikt prasības veloceliņu pārklājuma un infrastruktūras kvalitātei ar izvēlnēm $ROAD_i$. Šajā modelī lietotājs simulē noslodzi sešos iespējamā maršruta segmentos. Kartē ir parādīti četri sākuma punkti/galamērķi (P1, P2, P3 un P4), ceļš starp šiem punktiem ir sadalīts sešos segmentos (a, b, c, d, e un f). Segmentu noslodze tiek modelēta programmā RePast katram segmentam atsevišķi.

```
extensions [ece]
globals[
  last_temperature
  last_wind
  last_precipitation
  new_cyclist_data
  last_new_cyclist_data_tick
]
```

```
to setup_ece
  ece:init "localhost"
  ece:subscribe "FUPOL_SIMULATION_weather_temp"
  ece:subscribe "FUPOL_SIMULATION_weather_wind"
  ece:subscribe "FUPOL_SIMULATION_weather_precip"
  ece:subscribe "FUPOL_SIMULATION_new_cyclist_agent"
end
```

8.attēls. NetLogo modeļa ECE iestatīšanas programmas kods.

NetLogo modeļa programmatūra (D-slānis) iekļauj vaicājumus C-slāņa bibliotēkām (skat.

9.attēls).

```
extensions [ece]
globals[last_temperature last_wind last_precipitation new_cyclist_data last_new_cyclist_data_tick]

to setup_ece
  ece:init "localhost"
  ece:subscribe "FUPOL_SIMULATION_weather_temp"
  ece:subscribe "FUPOL_SIMULATION_weather_wind"
  ece:subscribe "FUPOL_SIMULATION_weather_precip"
  ece:subscribe "FUPOL_SIMULATION_new_cyclist_agent"
end

to send_time_and_date
  ece:sendNumber "FUPOL_SIMULATION_time" ticks MinuteOfTheDay
  ece:sendNumber "FUPOL_SIMULATION_date" ticks DayOfTheYear

end

to send_road_configuration
  ece:sendNumber "FUPOL_SIMULATION_road_a" ticks ROAD_a
  ece:sendNumber "FUPOL_SIMULATION_road_b" ticks ROAD_b
  ece:sendNumber "FUPOL_SIMULATION_road_c" ticks ROAD_c
  ece:sendNumber "FUPOL_SIMULATION_road_d" ticks ROAD_d
  ece:sendNumber "FUPOL_SIMULATION_road_e" ticks ROAD_e
  ece:sendNumber "FUPOL_SIMULATION_road_f" ticks ROAD_f
end

to get_last_weather_data
  set last_temperature ece:getLastNumberData "FUPOL_SIMULATION_weather_temp"
  set last_wind ece:getLastNumberData "FUPOL_SIMULATION_weather_wind"
  set last_precipitation ece:getLastNumberData "FUPOL_SIMULATION_weather_precip"
end

to get_new_cyclist_data
  if last_new_cyclist_data_tick != ece:getLastTick("FUPOL_SIMULATION_new_cyclist_agent")[
    set new_cyclist_data ece:getLastStringData("FUPOL_SIMULATION_new_cyclist_agent")
    set last_new_cyclist_data_tick ece:getLastTick("FUPOL_SIMULATION_new_cyclist_agent")
  ]
end
```

9.attēls. NetLogo modeļa D-slāņa specifikācija (Aizstrauts un Ginters, 2024).

Rezultātu panelī tiek parādīti Python programmatūras ģenerētie iespējamie laikapstākļi izvēlētajai dienai un laikam (nokrišņi, vēja ātrums un temperatūra), un tiek parādīta stabiņu diagramma, kas attēlo potenciālo noslodzi katrā modelētajā segmentā. Atbilstoši noslodzes rezultātiem, velosipēdists var pieņemt lēmumu iekļaut šo segmentu savā maršrutā vai arī to ignorēt.

Tā kā laikapstākļu simulācija ir izstrādāta Python programmēšanas valodā, tad ECE bibliotēka (C-slānis) nodrošina Python modelim (D-slānis) nepieciešamo ECE klašu kopumu. Šī modeļa galvenais uzdevums ir pārraidīt iespējamos laikapstākļus, kas atbilst ieplānotā ceļojuma datumam un laikam.

Noslodzes simulācija tiek realizēta RePast Symphony pēc iepriekš validēta algoritma (Ginters, Aizstrauts et al., 2016). RePast C-slāņa bibliotēka ir universāla, un to var iekļaut jebkurā Java lietojumprogrammā. Lietotāja RePast modelis (D-slānis) ietver pieprasījumus C-slāņa bibliotēkām, piemēram, NetLogo un Python.

Aplūkotajā piemērā, modelētājs uzdevums ir iekļaut ECE D-slāņa komandas apakšmodeļa programmatūras kodā, bet augstākie ECE līmeņi nodrošinās nepieciešamos komunikācijas servisus (Aizstrauts un Ginters, 2024).

Diskusija. Ja modelētājam būs pietiekamas zināšanas, lai izstrādātu uz NetLogo, RePast vai citu imitācijas modelēšanas tehnoloģiju balstītu modeli, tad viņam būs arī nepieciešamās prasmes, lai iestrādātu piekritīgās ECE komandas modeļa kodā. Veicamo darbību sarežģītības līmenis nav salīdzināms ar prasmēm, kuras ir vajadzīgas, lai strādātu ar HLA vidi.

Sadalītā imitāciju modeļa veiktspēju nosaka modelētāja izstrādātais iekļauto modeļu mijiedarbības algoritms, jo ECE nodrošina tikai komunikācijas iespējas. To pašu var attiecināt uz HLA. Tomēr var rasties jautājums, kurai no komunikācijas vidēm – HLA vai ECE – ir augstāka veiktspēja?

HLA reti izmanto reāllaika lietojumprogrammās, jo HLA stūrakmens nav maksimāla veiktspēja, bet gan drošība. HLA nodrošina atzvanīšanas (apstiprināšanas) funkciju, kas ir iekļauta katrā federācijā. Tas palielina drošību, bet vienlaikus arī skaitļošanas resursu noslodzi, kas savukārt izraisa kavējumu komunikācijā. ECE apraides mehānisms ir vienkāršāks un tādēļ ātrāks. Tā kā ECE ir balstīta uz AMQP protokolu, kas ir viens no ātrākajiem, tad iespējamais kavējums ECE būs mazāks nekā HLA vidē, jo ECE ietaupa ziņojuma apstrādes laiku, nepieprasot ziņojuma saņemšanas apstiprinājumu. AMQP ir būtiski ātrāks arī par HTTP (*Hypertext Transfer Protocol*), tādēļ, ja modelētājs izvieto imitācijas modeli tīmeklī, ECE nebūs iemesls veiktspējas zudumam (Aizstrauts un Ginters, 2024).

Vai ECE ir pietiekami ātra reāllaika lietojumam tehniskās sistēmās, kad modelētājs projektē digitālos dvīņus IoT objektu kontrolei reāllaikā?

IoT lietojumprogrammas parasti izmanto "primitīvākus" protokolus, piemēram, MQTT (*Message Queue Telemetry Transport*), jo tehnisku sistēmu vadībai tiek lietoti īsāki datu ziņojumi. Ja ECE tiktu balstīta uz MQTT protokolu, tad tās pilnvērtīga izmantošana sadalītu un heterogēnu imitāciju modelēšanas sistēmu projektēšanai būtu stipri ierobežota. Tomēr RabbitMQ uztur arī MQTT, tāpēc ECE pielāgošana tehnisku sistēmu vadībai neradītu būtisku apgrūtinājumu.

ECE nodrošina modelētājam saprātīgu komunikācijas savlaicīgumu, mērogojamību, modularitāti, drošību un veiktspēju. Tomēr, ja modelētājs vēlas izstrādāt sadalītas imitācijas modelēšanas sistēmas, kuras tiks izmantotas digitālo dvīņu projektos kritiskas infrastruktūras vadībai vai dzīvībai svarīgu procesu nodrošināšanai, tad labāk izmantot industrijā labi un ilgi validēto, standartizēto, neērto, sarežģīto, dārgo un "smago" HLA mehānismu, savlaicīgi iesaistot projektēšanas procesos augstas kvalifikācijas programmēšanas inženierus.

Mēģinājumi "humanizēt" HLA un aizstāt to ar uz pakalpojumu orientētu arhitektūru SOA (*Service Oriented Architecture*) kritiskos imitāciju modelēšanas lietojumos joprojām ir bijuši neveiksmīgi (Iagaru, 2022).

4. SECINĀJUMI

Sociotehniskās sistēmas, kas caurvij mūsu sabiedrību un tautsaimniecību, ir sarežģītas, jo tās ietekmē dažādi stohastiskie faktori, no kuriem visnozīmīgākā, bet arī visgrūtāk paredzamā ir cilvēka ietekme. Īpaši sarežģīti ir modelēt atsevišķu objektu uzvedību un to mijiedarbību. Lai pārbaudītu dažādus situāciju attīstības scenārijus un novērtētu mūsu rīcības iespējamās sekas, var izmantot imitāciju modelēšanu. Sarežģītus procesus un parādības nevar imitēt ar viendabīgiem jeb homogēniem simulācijas modeļiem, tādēļ tiek lietoti sadalīti un heterogēni imitācijas modeļi. Būtiska problēma ir datu apmaiņas nodrošināšana starp šiem modeļiem un to darbības koordinēšana modelēšanas sesijas laikā.

Tas ir pārsteidzoši, taču sadalītu imitācijas modeļu projektēšanai joprojām tiek izmantota HLA vide, kas nav pieejama modelētājiem bez īpašām zināšanām un labām prasmēm programmēšanas inženierijā (Aizstrauts un Ginters, 2024).

Augstāk minētā problēma kavē adekvātu rīcībpolitikas lēmumu pieņemšanu un dažādu attīstības scenāriju novērtēšanu jebkurā nozarē, jo modelēšana kļūst atkarīga no specifiskām programmatūras izstrādes prasmēm. Lai strādātu ar HLA vidi ir papildus jāiesaista augstas kvalifikācijas programmēšanas inženierijas speciālisti. Šī ir problēma, kas traucē ieviest imitāciju modelēšanu lēmumu un seku novērtēšanā, it īpaši organizācijās, kas nav tehniski orientētas, kā, piemēram, pašvaldības, ministrijas vai citas pārvaldes institūcijas.

Otra iespēja - izmantot kādu no daudzfunkcionālajām, bet slēgtajām simulācijas vidēm, apdraud izstrādāto simulatoru ilgtspēju, jo kļūst atkarīga no ekskluzīva produkta piedāvātāja, un neļauj iekļaut sadalītā modelī ar citiem simulācijas rīkiem radītus modeļus.

17 gadu laikā ir izstrādāta un validēta sadalītu un heterogēnu imitācijas modeļu Vienkāršotas komunikācijas vide (ECE), kas ir pieejama un ērti lietojama dažādu nozaru profesionāļiem bez īpašām programmēšanas inženierijas prasmēm.

Ir izstrādāta elastīgu protokolu un datu formātu kopa, kuru izmaiņas neradīs problēmas simulācijas modeļu ilgtspējas nodrošināšanā.

Vairāku slāņu atvērtā ECE arhitektūra nodrošina dažādu simulācijas tehnoloģiju savietojamību un heterogēnu modeļu mijiedarbību.

ECE priekšrocības ir tās izmantošanas iespējas ne tikai sadalītu imitācijas modelēšanas sistēmu izstrādē, bet arī datu apstrādes apakšsistēmu sadarbības nodrošināšanā, tāpēc ECE auditorija ir ne tikai pētnieki un modelētāji, bet arī sadalītu datu apstrādes sistēmu projektētāji.

ECE koncepts un funkcionalitāte ir validēta vairākos nacionālos un Eiropas Komisijas finansētos zinātniskas pētniecības projektos, savukārt pētījuma rezultāti ir publicēti dažādos avotos un par tiem ir ziņots starptautiskās zinātniskās konferencēs.

Autors uzskata, ka promocijas darba uzdevumi ir izpildīti, un mērķis "*Nodrošināt iespējas modelētājiem bez specifiskām zināšanām un prasmēm programmēšanas inženierijā, konstruēt sadalītus un heterogēnus imitācijas modeļus, balstoties uz atvērtas un vienkāršotas komunikāciju vides konceptuālu ietvaru*" ir sasniegts. Promocijas darba izstrādes rezultāti atļauj aizstāvēt formulētās pamata tēzes.

Pagaidām pietrūkst ECE metodisko un mācību materiālu, kā arī nav izstrādātas pietiekami plašas simulācijas rīku bibliotēkas, tomēr autors turpinās darbu pie šo nodevumu sagatavošanas.

Nākamais pētījumu attīstības solis būs ECE piemērotības validācija sadalītu un heterogēnu simulācijas modeļu savienošanai ar Beijesa tīklu, lai modelētu iespējamā kiberapdraudējuma izplatīšanos datu pārraides tīklā.

PATEICĪBAS UN APLIECINĀJUMI

Autors izsaka pateicību Vidzemes Augstskolai un Rīgas Tehniskās universitātes Informācijas tehnoloģiju institūtam par atbalstu ECE pētījumu īstenošanā.

ECE izstrāde ir bijis ļoti ilgs projekts, tāpēc autors vēlas pateikties ne tikai saviem kolēģiem, kuri izturēja nebeidzamos ECE uzlabojumus un modifikācijas, bet arī viņu ģimenēm, kuras samierinājās un atļāva viņiem veltīt laiku ECE pētījumiem.

LITERATŪRAS SARAKSTS

Aizstrauts, A., Burkhardt, D., Ginters, E., & Nazemi, K. (2020). On Microservice Architecture Based Communication Environment for Cycling Map Developing and Maintenance Simulator. In *2020 61st International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS)* (pp. 1-4). doi: 10.1109/ITMS51158.2020.9259299. Izgūts no https://www.researchgate.net/publication/347072223_On_Microservice_Architecture_Based_Communication_Environment_for_Cycling_Map_Developing_and_Maintenance_Simulator#fullTextFileContent

Aizstrauts, A. & Ginters, E. (2024). Easy Communication Environment for Heterogeneous and Distributed Simulation Models Design. In *2024 36th European Modeling & Simulation Symposium* (p. 13). Tenerife. (Publicēšanai pieņemts modificēts promocijas darba kopsavilkums).

Aizstrauts, A., Ginters, E. & Aizstrauta, D. (2011). Easy Communication Approach for Data Exchange in Distributed Simulation Environment. In *Proceedings of the 13th WSEAS international conference on Automatic control, modelling & simulation (ACMOS'11)* (pp. 34-38). Stevens Point: World Scientific and Engineering Academy and Society (WSEAS). Izgūts no http://www.wseas.us/e-library/conferences/2011/Lanzarote/ACMOS/ACMOS-04.pdf

Aizstrauts, A., Ginters, E., Aizstrauta, D., & Sonntagbauer, P. (2012). Easy Communication Environment on the Cloud as Distributed Simulation Infrastructure. In *Proceedings of the 5th WSEAS congress on Applied Computing conference, and Proceedings of the 1st international conference on Biologically Inspired Computation (BICA'12)* (pp. 173-178). Stevens Point, Wisconsin: World Scientific and Engineering Academy and Society (WSEAS). Izgūts no http://www.wseas.us/e-library/conferences/2012/Algarve/BICA/BICA-29.pdf.

Aizstrauts, A., Ginters, E., Baltruks, M., & Gusev, M. (2015). Architecture for Distributed Simulation Environment. *Procedia Computer Science*, *43*, pp. 18-25. doi: 10.1016/j.procs.2014.12.004. Izgūts no https://www.sciencedirect.com/science/article/pii/S1877050914015725

Aizstrauts, A., Ginters, E., Lauberte, I., & Piera Eroles, M.A. (2013). Multi-level Architecture on Web Services Based Policy Domain Use Cases Simulator. In *Lecture Notes in Business Information Processing* (vol 153, pp.130-146). Berlin: Springer. doi:10.1007/978-3-642-41638-5_9. Izgūts no https://www.researchgate.net/publication/282289690_Multi-level_Architecture_on_Web_Services_Based_Policy_Domain_Use_Cases_Simulator#fullTextFileContent

Borshchev, A. (2014). Multi-method modelling: AnyLogic. In *Discrete-Event Simulation and System Dynamics for Management Decision Making* (eds S. Brailsford, L. Churilov and B. Dangerfield). doi: 10.1002/9781118762745.ch12

Dahmann, J.S., Fujimoto, R.M., & Weatherly, R.M. (1997). The Department of Defense High Level Architecture. In *Proceedings of the 29th conference on Winter simulation (WSC '97)* (pp. 142-149). IEEE Computer Society. doi: 10.1145/268437.268465

Dijkstra, E.W. (1968). The Structure of the “THE” - Multiprogramming System. *Commun. ACM*, *11*, 342–346.

Dissanayake, S. T. M. (2016). Using STELLA Simulation Models to Teach Natural Resource Economics. *The Journal of Economic Education, 47*(1), 40–48. doi: 10.1080/00220485.2015.1106358

Ginters, E., Aizstrauts, A., Baltruks, M., Merkuryev, Y., Novickis, L., Grundspenkis, J., & Grabis, J. (2016). VeloRouter - Technology for Urban Transport Intermodal Sustainability. In *City Planning and Urban Design Conference, 07-09 April (CPUD’16)* (pp. 211-220). Istanbul: DAKAM. ISBN 978-605-9207-21-8. Izgūts no https://www.dakamconferences.org/_files/ugd/bac820_3252301ff80b453e928fb159e78a9910.pdf

Ginters, E., Sakne, I., Lauberte, I., Aizstrauts, A., Dreija, G., Aquilar Chinea, R.-M., Merkuryev, Y., Novitsky, L., & Grundspenkis, J. (2011). Simulation Highway – Direct Access Intelligent Cloud Simulator. In *Proceedings of 23rd European Modelling & Simulation Symposium (EMSS 2011)* (pp. 62-72). Izgūts no https://docplayer.net/16307931-Simulation-highway-direct-access-intelligent-cloud-simulator.html

Ginters, E., & Silins, A. (2007). Multi-level approach for environmental systems modelling in the Ligatne Natural Trails. *WSEAS Transactions on Systems*, 6(4), 795-801. Izgūts no https://www.researchgate.net/publication/297118169_Multi-level_approach_for_environmental_systems_modelling_in_the_Ligatne_Natural_Trails

Ginters, E., & Silins, A. (2008a). Exchange Mechanisms in Distributed Simulation of Sociotechnical Systems. In *Information Society and Modern Business. Proceedings of 3rd International conference, 21-22 September 2007, Ventspils* (p. 362). Ventspils: Ventspils University College. Izgūts no https://daugavpils.biblioteka.lv/alisepac/Details?title=Information-Society-and-Modern-Business.-3rd-International-conference-%3A-proceedings-21-22-September,-2007,-Ventspils&Id=85013&Ident=1218592&InstanceId=13&LibraryId=0#bibliographic

Ginters, E., & Silins, A. (2008b). Simulation Data Exchange in Distributed E-learning Environment. In *Proceedings of the 4th WSEAS/IASME International Conference on*

Education Technologies (EDUTE'08) (pp. 138-143). Corfu: WSEAS. Izgūts no http://www.wseas.us/e-library/conferences/2008/corfu/edute/edute23.pdf

Ginters, E., Silins, A., & Andrusaitis, J. (2007). Communication in Distributed Simulation Environment. In *ICOSSSE'07: Proceedings of the 6th WSEAS international conference on System science and simulation in engineering* (pp. 217-221). Stevens Point: World Scientific and Engineering Academy and Society. doi: 10.5555/1974442.1974475 Izgūts no http://www.wseas.us/e-library/conferences/2007venice/papers/600-139.pdf

Ginters, E., & Revathy, J. G. (2021). Hidden and Latent Factors' Influence on Digital Technology Sustainability Development. *Mathematics, 9*(21), 2801. doi: 10.3390/math9212801

Grand View Research,. (2024). *Digital Transformation Market Size, Share, Growth & Trends Analysis Report by Solution, By Deployment, By Service, By Enterprise Size, By End-use, By Region, And Segment Forecasts, 2024 – 2030. Report ID: GVR-1-68038-851-0.* Izgūts no https://www.grandviewresearch.com/industry-analysis/digital-transformation-market

Iagăru, E. (2022). Comparative Analysis Between High Level Architecture (HLA) and Service Oriented Architecture (SOA) in the Field of Military Modelling and Simulation. *Scientific Bulletin, 27(1)*, pp. 30 - 40. doi: 10.2478/bsaft-2022-0004

ISO/IEC 7498-1:1994(E). (2024). Information Technology – Open Systems Interconnection – Basic Reference Model: The Basic Model. Second Edition 1994-11-15. 68 p. Izgūts no https://www.ecma-international.org/wp-content/uploads/s020269e.pdf

Jabbour, J., Possik, J., & Zacharewicz, G. (2023). A Modeling and Distributed Simulation Platform for Immersive Experience. In *JIAE 2023 - Journées sur l'Interopérabilité des Applications d'Entreprise, Pôle Grand Sud-Ouest (PGSO)*. Lille, France. Izgūts no https://hal.science/hal-04199808/document

Naik, N. (2017). Choice of Effective Messaging Protocols for IoT Systems: MQTT, CoAP, AMQP and HTTP. In *2017 IEEE International Systems Engineering Symposium (ISSE)* (pp. 1-7). IEEE. doi: 10.1109/SysEng.2017.8088251

OMG (Object Management Group),. (2015). *About the Data Distribution Service Specification Version 1.4*. Izgūts no https://www.omg.org/spec/DDS/1.4

Possik, J., Gorecki, S., Asgary, A., Solis, A.O., Zacharewicz, G., Tofighi, M., Shafiee, M.A., Merchant, A.A., Aarabi, M., Guimaraes, A., & Nadri, N. (2021). A Distributed Simulation Approach to Integrate AnyLogic and Unity for Virtual Reality Applications: Case of COVID-19 Modelling and Training in a Dialysis Unit. In *2021 IEEE/ACM 25th International Symposium on Distributed Simulation and Real Time Applications (DS-RT)* (pp. 1-7). Valencia: IEEE.

Silins, A., Ginters, E., & Aizstrauta, D. (2010). Easy Communication Environment for Distributed Simulation. In *Computational Intelligence in Business and Economics* (pp. 91-98). https://doi.org/10.1142/9789814324441_0014. Izgūts no https://www.researchgate.net/publication/282288675_Easy_Communication_Environment_for_Distributed_Simulation#fullTextFileContent

Strassburger, S., Schulze, T., & Fujimoto, R. (2008). Future Trends in Distributed Simulation and Distributed Virtual Environments: Results of a Peer Study. In *2008 Winter Simulation Conference* (pp. 777-785). Miami, FL, USA. doi: 10.1109/WSC.2008.4736140. Izgūts no https://ieeexplore-ieee-org.resursi.rtu.lv/stamp/stamp.jsp?tp=&arnumber=4736140

Versteegt, C., & Verbraeck, A. (2002). The Extended Use of Simulation in Evaluating Real-Time Control Systems of AGVs and Automated Material Handling Systems. In *Proceedings of the 2002 Winter Simulation Conference* (pp. 1659-1666). San Diego. doi: 10.1109/WSC.2002.1166448

Xu, L., Lin, SY., Hlynka, A.W. *et al.* (2021). Distributed Simulation Platforms and Data Passing Tools for Natural Hazards Engineering: Reviews, Limitations, and Recommendations. *Int J Disaster Risk Sci*, *12*, 617–634. doi: 10.1007/s13753-021-00361-7

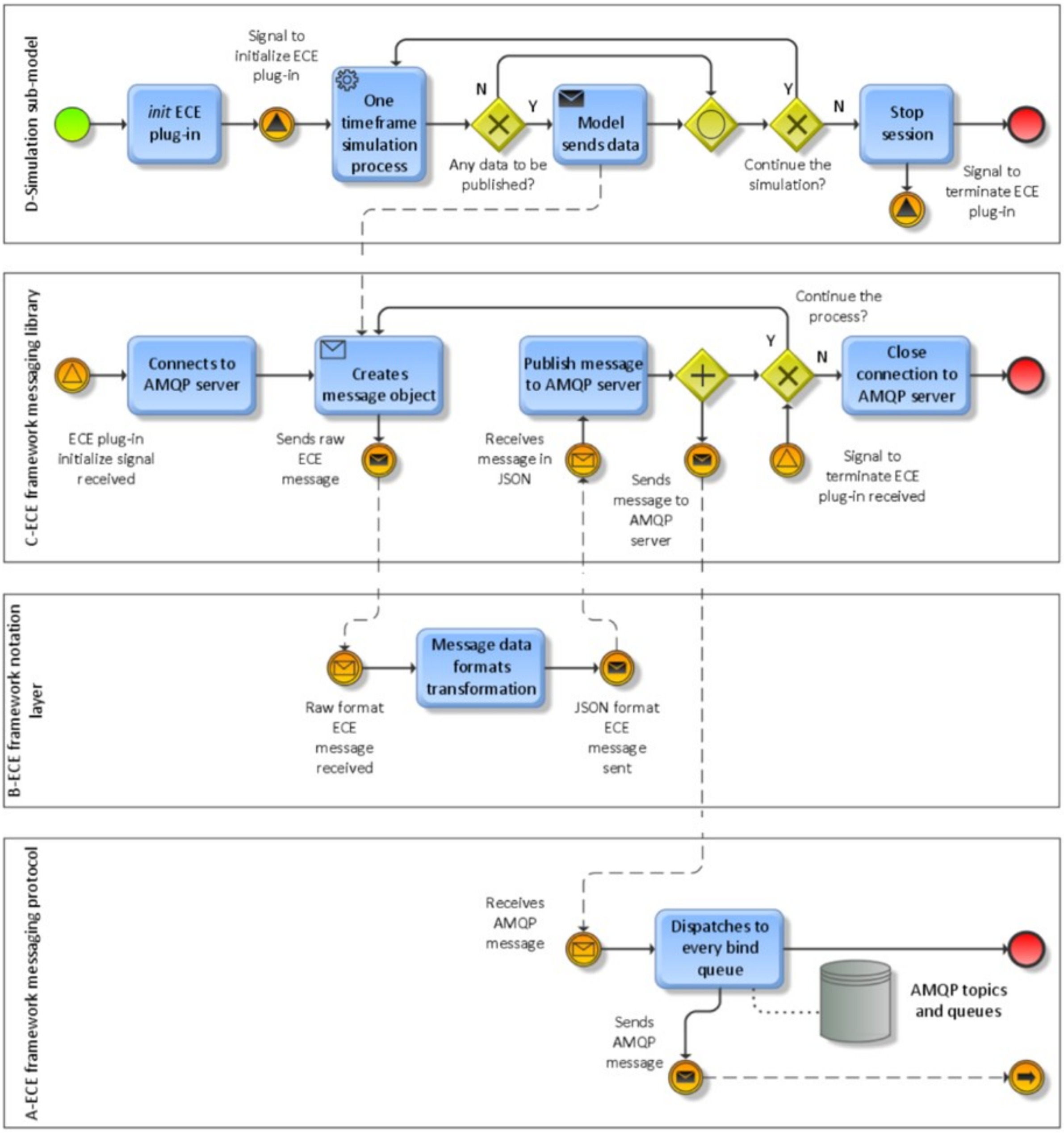
Wang, S., Wang, S.M., & Zhang, N. (2022). Flexsim-based Simulation and Optimization of Green Logistics Distribution Center. In *Proceedings of the 14th International Conference on Computer Modeling and Simulation (ICCMS '22)* (pp.76-82). New York: Association for Computing Machinery. doi: 10.1145/3547578.3547590

Wilensky, U., & Rand, W. (2015). An Introduction to Agent-Based Modeling. Modeling Natural, Social, and Engineered Complex Systems with NetLogo. MIT Press.

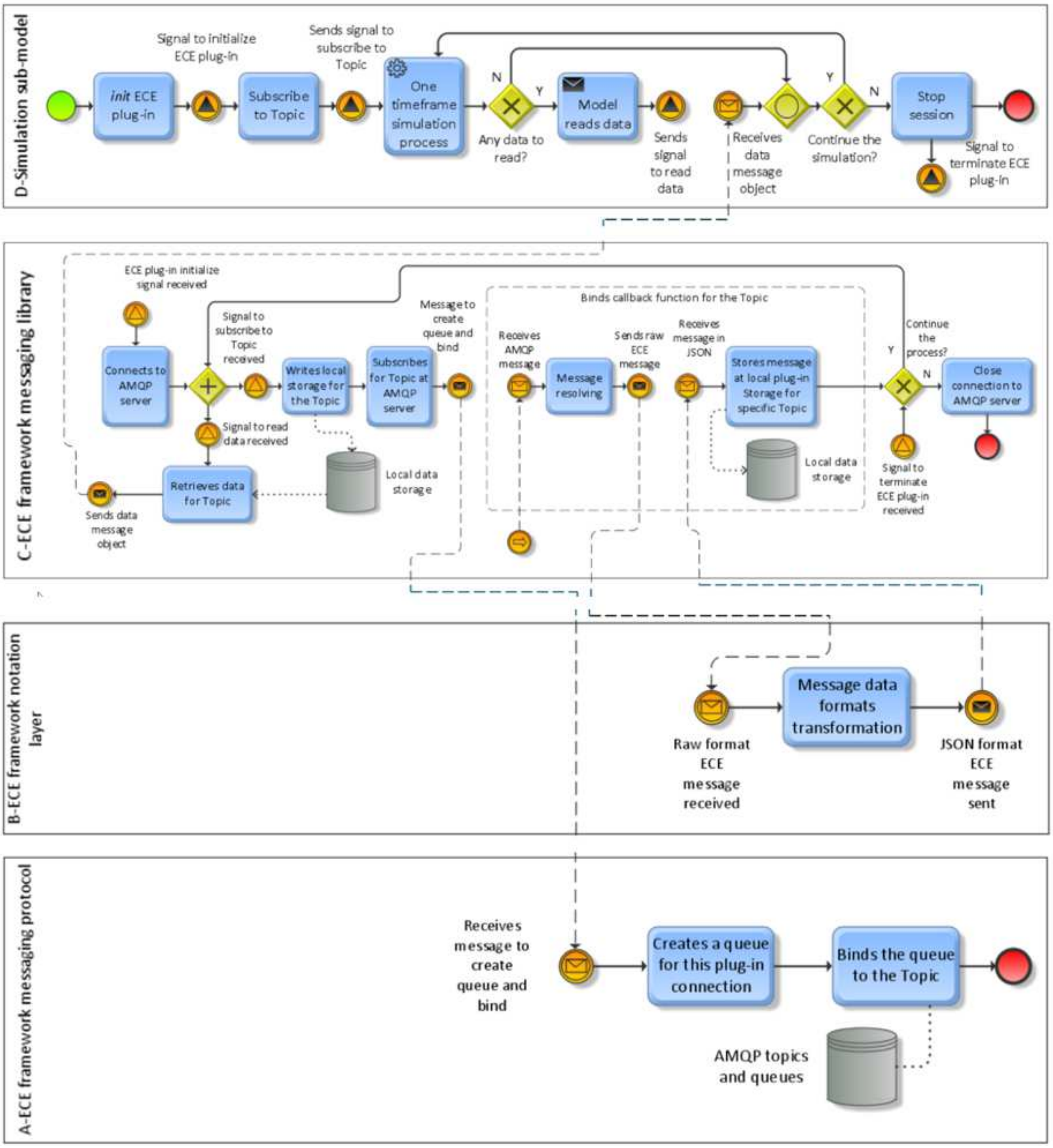
Zimmermann, H. (1980). OSI Reference Model – The ISO model of Architecture for Open Systems Interconnection. *IEEE Transactions on Communications, 28* (4), 425-432.

A. PIELIKUMS. VIENKĀRŠOTAS KOMUNIKĀCIJAS VIDES (ECE) FUNKCIONALITĀTES BPMN2 DIAGRAMMA

ECE slāņu sadarbības BPMN2 diagramma



ECE slāņu sadarbības BPMN2 diagramma (turpinājums)



B. PIELIKUMS. PIEKRITĪGĀS PUBLIKĀCIJAS

Aizstrauts, A. (2024). An Easy Communication Implementation of Heterogeneous and Distributed Simulation Models [Zenodo]. https://doi.org/10.5281/zenodo.13925452

1. Ginters, E., & Silins, A. (2007). Multi-level approach for environmental systems modelling in the Ligatne Natural Trails. *WSEAS Transactions on Systems*, 6(4), 795-801. Retrieved from https://www.researchgate.net/publication/297118169_Multi-level_approach_for_environmental_systems_modelling_in_the_Ligatne_Natural_Trails

WSEAS TRANSACTIONS on SYSTEMS

Issue 4, Volume 6, April 2007

ISSN 1109-2777 http://www.wseas.org

Multi-level approach for environmental systems modelling in the Ligatne Natural Trails

EGILS GINTERS, ARTIS SILINS
Sociotechnical Systems Engineering Institute
Vidzeme University College
4 Cesu Street, Valmiera LV-4200
LATVIA
{egils.ginters, artis.silins}@va.lv http://www.va.lv

Abstract: - The Gauja National Park (Latvia) was established in 1973. The Gauja National Park focuses mainly on nature protection, but it is also used for educational and leisure tourism. The Ligatne Natural Trails was founded in 1975 as an integrated tourism object for recreation and ecosystems research giving possibility for introduction with flora and fauna, which is typical for Latvian regions. The Ligatne Natural Trails is a part of the Gauja National Park. There exist different routes ensuring movement the visitors either by foot or by car. The total amount of visitors and cars are permanently growing. Therefore, one of the tasks is intelligent planning of exploiting of the nature resources, which can be solved using multi-level approach involving the structural modelling (LISTechnology), simulation of macro processes (EXTEND) and estimation of the critical resources using agent-based simulation tools (NetLogo).

Key-Words: - Agent-based simulation, Discrete event systems simulation, Environmental systems, EXTEND, LISTechnology, NetLogo, Sociotechnical systems

1 Introduction

The Gauja National Park (Latvia) was founded in 1973 [1]. The park includes the primeval valley of the Gauja River with its tributaries forming a particularly original 91745 ha landscape. On the banks of the rivers and brooks, there are the biggest Devonian outcrops in Latvia – sandstone cliffs, rocks, and caves. The Gauja National Park focuses mainly on nature protection but it is also used for educational and leisure tourism.

The Ligatne Natural Trails was founded in 1975 as an integrated tourism object for recreation and ecosystems research giving possibility for introduction with flora and fauna, which is typical for Latvian regions. The Ligatne Natural Trails is a part of the Gauja National Park. There exist different routes ensuring movement the visitors either by foot or by car. The total amount of visitors and cars are permanently growing. The problem mentioned above asks for expanding of parking and traffic dispatching. Permanently growing amount of visitors makes pressure on functioning of ecosystem and regeneration possibilities of lawns in sightseeing places. Therefore, potential workload forecasting of tourism object, which depending of tourists flows intensity, becomes important.

The modelling can be done at static and dynamic stages (LISTechnology). The infological modelling can be implemented in the static stage to recognise the main tourist and car flows. The operating of the tourism system firstly can be analysed at macro process level granting possibilities for transparency of the all processes carried out in the tourism object.

At this level, the tourism system can be simulated as discrete event system (EXTEND) for determination of the throughput of the system. Other question is the influence of the tourist flows intensity on the regenerating capabilities of nature resources.

To analyse and forecast the critical situations in ecosystem at the micro process level another approach based on agent-based simulation (NetLogo) can be used.

2 Multi-level approach in sociotechnical systems modelling

Simulation professionals have significant experience in simulation of the technical systems due invested serious funding into development of the simulation environment, because any upgrade or optimisation of the manufacturing operations ensure appropriate increase of the profit or competitiveness.

The technical systems mostly cover mechanisms, processes, procedures, and systems operating in conformity with predefined algorithms. The simulation tools oriented to such a system modelling are prevalent. Many tools and environments existing, which are oriented to discrete, continuous and hybrid systems simulation: Arena, EXTEND, Witness, AutoMod, Promodel, Simul8, and other.

Social systems involve people with their manner, values, behaviour, style and relations, laws and rules or other like biologic and environmental systems. These are systems for which changeable algorithm of functioning is typical and controling of the external perturbations is very significant unlike technical systems simulation, where human factor is respected minimally.

Sociotechnical systems are manageable systems respecting human and/or biologic and/or environmental factor role, mutual assistance and influence on joint functioning of the both systems.

The Ligatne Nature Trails was established in 1975 in order to acquaint visitors with nature, mammal, and plant species characteristic for Latvia, nature diversity, and protection. It is located in the territory of the Gauja National Park. On the banks of the Gauja River and in the middle of wooded glens, trails of more than 5 km are arranged.

By choosing the trail along the Gauja River, wild nature trail, or the route for visitors with cars, it is possible to get to know landscapes characteristic for the Gauja River valley and to see the animals have been brought from various places of Latvia [2].

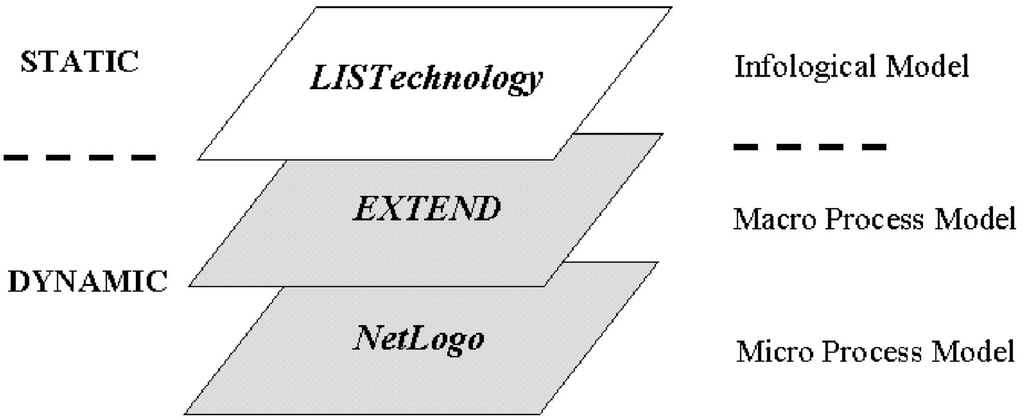


Fig.1. The architecture of Ligatne Natural Trails model

In this case are two kind resources, which are critical for operating of the tourism object. Firstly, those are the parking places along trails and scenic routes. Each route has their throughput (cars or tourists per time unit). Functional model in this case can be described as discrete event system and can be simulated in EXTEND environment for instance. Other critical resource is capacity and quality of the sight place. Under the quality we understand the capabilities of regenerating of the nature resources (lawn, for instance) in given area influenced by the flow of the tourists with predefined intensity. In this case would not be so easy to create a useful model in EXTEND, therefore agent based approach can be used and NetLogo environment have been selected (see Fig.1).

The tourism system mentioned above is a typical sample of the sociotechnical system.

3 LISTechnology – methodology for the systems structural modelling

Due to the development of distributed and complicated tourism systems, designing of which requires remarkable financial and time resources, the necessity for designing methodologies in order to ensure the transparency of the project, is becoming topical. After all, the persons who pay for the project wants also to understand what is going on in this project. For this reason the systems structure analysis and designing methodology LISTechnology some years ago was created [3].

The methodology named LISTechnology was developed to promote a clearness and transparency of the project for the customer's administration and serves for both systems' analysis and designing needs. One of the typical application fields of LISTechnology is logistics, therefore it well suited for the tourist, and cars flow description in the Ligatne Natural Trails.

Analysis of the system by means of LISTechnology can involve of three stages: STATIC, DYNAMIC, and EXPERTISE stage (see Fig.2), because in some cases more detailed exploring of the system elements does not necessary and last step can be left.

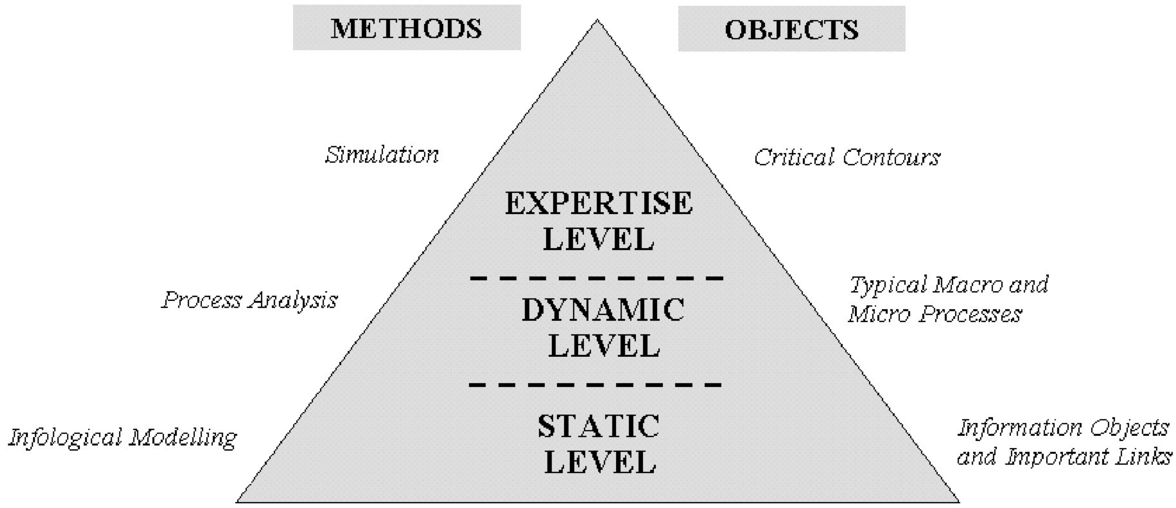


Fig. 2. The architecture of the LISTechnology

Splitting of the goal system into sets of the objects and the important information links is implemented at the STATICS stage, where the logical structure of the goal system is described in the form of the infological model. The main task at the current level is determination of important objects and describing of an initial structure of the goal system.

The graphical representation is accomplished in the forms of *Communication Diagrams*, *Data Diagrams* and *Interface Tables*. Communication Diagrams are used for graphical and structural presentation of the goal system model with a necessary level of abstraction, and consist of information objects and information channels among them. Data Diagrams reveal precise hierarchical subordination of the objects, but Interface Tables involve most important data transferred among objects. At this modelling level is possible to consider about total capacity of the modelling object and to reduce some redundant information flows, which only bother normal operating the object.

The representation of the goal system by the set of typical processes is implemented at the second, DYNAMIC stage of analysis, where the dynamic or processes form of the logical structure is presented. The graphical representation mostly is performed using *Macro Processes Level Diagrams (MPLD)*, but in case, if more detailed description is required, the *Processes Diagrams* are used. It is the main level at which the overall goal system model functioning is described. As input for those process models serves infological model created at higher level of abstraction at STATIC level.

At the EXPERTISE level a quantitative parameters of the structural elements are estimated, and it would be done, if necessary, using simulation means. Usually in such a way the quantitative analysis some of resources or contours, which are critical for some of criterions: expenses, capacity, dimensions, viability, service time, and other parameters are carried out.

4 Logistics modelling at macro process level in Ligatne Natural Trails

The architecture of the Ligatne Nature Trails system at STATIC level can be described as $\boldsymbol{S} = \langle \boldsymbol{L}, \boldsymbol{R} \rangle$, where $\boldsymbol{L}$ - the set of trails and routes, which linked objects of the $\boldsymbol{R}$ set, but $\boldsymbol{R}$ - the set of parking places $\boldsymbol{R}_p^i$ marked by squares and sight places $\boldsymbol{R}_S^j$ marked by grey circles (see Fig.3).

The set of links $\boldsymbol{L}$ consists of Nature trail with wild animals 5.5km, Gaujmala trail 2.0km, Wild nature trail 1.3 km and Scenic route for visitors with cars 5.1 km.

In conformity with the Communication Diagram's created before functioning of the logistics system at the macro process level can be described using Buhr notation given detailed enough presentation about process going on in the tourism object [4].

Macro Process Level Diagram shows general model of operating allowing description and modelling throughput of routes and capacity of parking places, which are critical resources of logistics system in Ligatne Natural Trails.

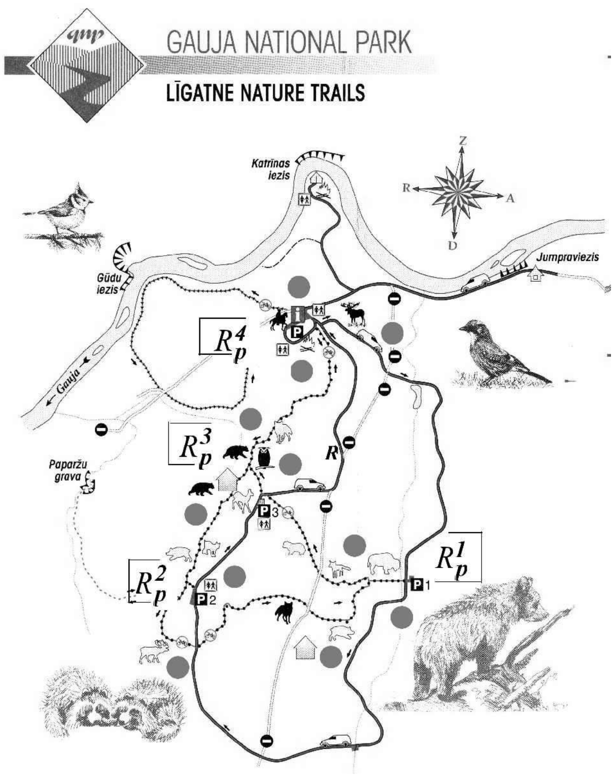


Fig. 3. Infological model of the Ligatne Nature Trails system

Further, this model can be transformed to EXTEND environment.
EXTEND is designed from the ground up to be a flexible, extendable discrete event system simulation tool. It can be used to model every aspect of an organization at all levels of expertise - from manager to engineer/scientist and from novice to professional modeller [5, 6].

EXTEND has an interactive and graphical architecture that is combined with a robust development environment. EXTEND's modern, advanced design and rich feature set reduces the amount of time for developing, validating, verifying, and analyzing the simulation models. Model builders can use EXTEND's pre-built modelling components to quickly build and analyze systems with little or no programming. Simulation tool developers can use EXTEND's built-in, compiled language, ModL, to develop new reusable modelling components.

In routing model of the Ligatne Natural Trails the following initial parameters are defined: total amount and capacity of each parking place, amount and capacity of sight places, average amount of visitors arrived by car, amount and throughput of pedestrian trails, amount and throughput of car routes and some statistics data mostly involving arrival and departure distributions of cars at tourism object (see Fig.4).

After detailed estimation of the collected data, it was supposed that intervals between visitor cars arriving are distributed exponentially. Of course, it is only statistical assumption with allowable probability and confidence, therefore simulation can gives only approximate results. If modeller would like to obtain results that are more precise, and especially for validation of the model, then it is possible to create their own generator based on some constant distribution generators federated in joint set. In such a set each constant generator will works in conformity with its own time schedule creating real flow of the visitors car interarrival distribution.

The Ligatne Natural Trails working hours are limited, because it would not be right-minded solution to attend wild animals during the night. Therefore, the Entrance and Booking must be closed at least one hour before closing the park giving possibilities for visitors to finish their travel. In EXTEND model it is achieved by queue with splitting outputs. Before deadline, the items arriving are transferred to the Entrance block. Otherwise, items are transferred out of the model.

The *Booking* is located at the parking place $\boldsymbol{R_P^4}$ (P4) and if total capacity of parking place is exceeded the visitor can decide to stay and wait a little bit or to go back. After booking, which takes some minutes the visitor have a chance to get to the Scenic route, but before the current situation in whole route must be estimated. Only four parking places are allowed on the one-way Scenic route with predefined total capacity. It is declared, while first parking place $\boldsymbol{R_p^1}$ (P1) is overloaded, the main gate must be closed, and car must wait in $\boldsymbol{R_P^4}$.

The buffering element (first-in-first-out queue with declared minimal waiting time, necessary for sightseeing) is used for simulation of sightseeing and parking, for instance *Sightseeing P1*. If queue is full then the signal about busy parking serves as feedback closing the main gate of the Scenic route.

After getting on the one-way Scenic route, the visitors have some scenarios: to go directly to the parking located close by some specific animals; to go to first free parking place on the route; to have slow some minutes driving to trail Exit. Any parking place R_P^i can be selected only if it has free places.

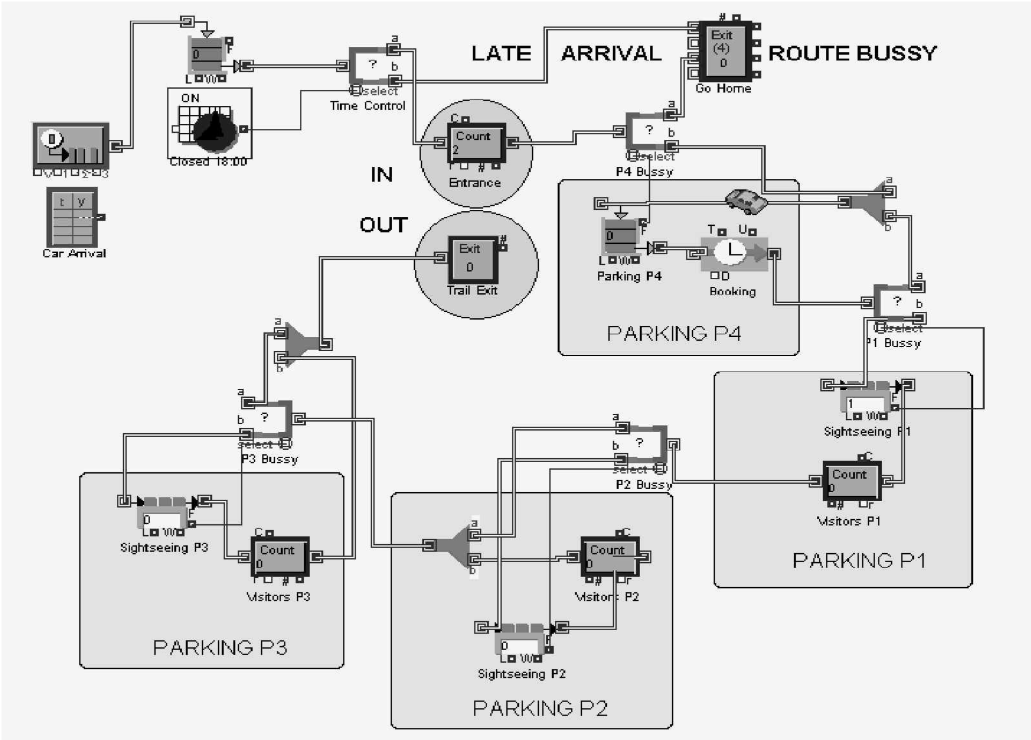


Fig. 4. Car routing model of Ligatne Natural Trails in EXTEND

Using the routing model (see Fig.4) the necessary capacity of the parking place R_P^i can be calculated, if intensity of the tourists flow (tourists per hour) is I^i. It can be recognized, how long the delay will be for waiting in the queue to get on the Scenic route with the car. The same manner is possible to plan the size of the sight places.

The current approach provides initial data for the second task formulated in previous chapters. It is the task related with forecasting of the regeneration capabilities of the nature resources of the sight place for instance. This simulation task is implemented at the micro process level. The intensity of the tourist flow I^j, which affected the environmental situation in sight place R_S^j, serves as initial data for agent-based simulation.

5 Agent-based simulation environment NetLogo for critical resources modelling

5.1 NetLogo Fundamentals

NetLogo is a programmable modelling environment for simulating natural and social phenomena. It was authored by Uri Wilensky in 1999 and is in continuous development at the Centre for Connected Learning and Computer-Based Modelling [7]. It is particularly well suited for modelling complex systems that are developing over time. Modellers can give instructions to hundreds or thousands of independent "agents" all operating concurrently. NetLogo is written in Java, so it can run on all major platforms as a standalone application. Individual models can be run as Java applets inside a web browser.

Some basic NetLogo terms are explained below to understand the model description.

The NetLogo "world" is made up of agents. Each agent can carry out its activity, all simultaneously. In NetLogo, there are three types of agents: turtles, patches, and the observer. Turtles are agents that move around in the "world". The world is two-dimensional and is divided up into a grid of patches. Each patch is a square piece of "ground" over which turtles can move. In NetLogo, commands and reporters tell agents what to do. A command is an action for an agent to carry out. A reporter computes a result and reports it. Variables can be a global, a turtle variable, or a patch variable. Buttons in the interface tab provide an easy way for interactive modelling. NetLogo plotting features let creating plots to promote understanding, what is going on during simulation. Set of primitives allows interacting with outside files, which is great advantage of NetLogo over StarLogo or Agentsheets for instance [8].

5.2 GRASS Logo – simulation model for decision-making support

An intensity of tourist flows I affected sight places R_S were determined as initial conditions for current modelling level during previous steps of the modelling.

How high will be the grass during selected period if meteorological conditions will be given and tourist flows intensity will be predefined? That is the problem for decision-maker, who must plan the tourism object operating and work load of the service staff.

Simulation model GRASS Logo utilizes three basic meteorological factors – temperature, humidity, and trample, which is the function of I .

All conditions are average values. The model notices the following basic relations:

- Higher temperature and lower humidity – grass height reduces fast;
- Higher temperature and higher humidity – grass grows fast;
- Temperature below zero – grass does not grow;
- The natural grass growth is affected by trample, which always reduces grass growth.

The GRASS Logo model starts with variables and "world" initialization clicking button "Setup" (see Fig.5). During initialization optimal grass height is given, the "world" is drawn and values to the basic variables are assigned.

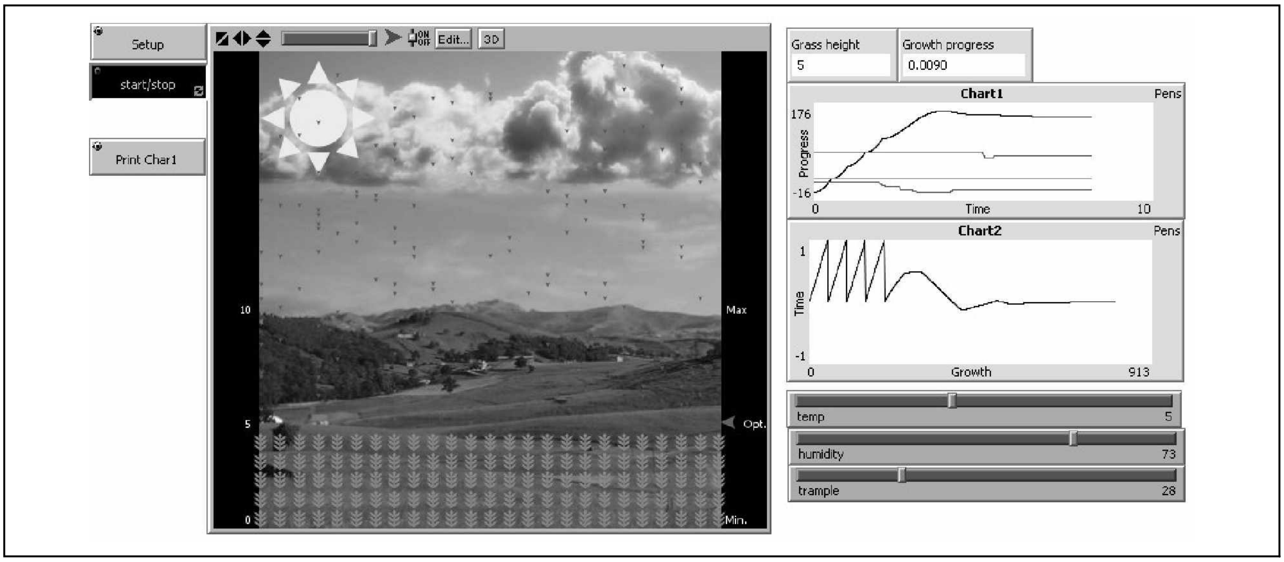


Fig. 5. Simulation desktop of the GRASS Logo model

Further, by clicking the "Start/Stop" button, is possible to launch or to stop the iteration of the simulation. The "Start/Stop" button calls the procedure "go". Procedure "go" describes main simulation process. During this process the speed and the density of the rain are calculated, which are related with given humidity. New drops are made on the modelling desktop by calling the procedure "new_drop". Then all "drop" agents are moved forward and the drops gained the ground are erased. The growth calculation is done and the colour of the sun, which depends of air temperature, is determined. Changes of all factors (temperature, humidity, trample, growth progress) are shown on the plots. To continue the simulation this procedure can be looped.

Button "Print Char1" exports "Chart 1" to the output file. Curve "Chart 1" shows values of "*temp*", "*humidity*", "*trample*", and all growing progress. Curve "Chart 2" – shows value of the variable "*factor*" representing growing.

Sliders "temp", "humidity", and "trample" ensure manual changes of the variables assigned during iteration.

The picture in the middle of simulation desktop is "world", where the agent's live.

The example shows, how the growth of the grass is calculated (see Fig.6).

```
1. set factor factor - 0.001 * trample
2. set factor factor + 0.0003 * humidity + 0.00125
   * temp
3. set progress progress + factor
4. if factor < -1 and statuss > -12 [set statuss
   statuss - 1 set factor 0 grow_down]
5. if factor > 1 and statuss < -1 [set statuss statuss
   + 1 set factor 0 grow_up]
6. if factor < -1 or factor > 1 [set factor 0]
```

Fig. 6. Extract of GRASS Logo model code

A new value to variable "*factor*" is assigned. It is calculated using formula: factor - 0.001 * trample. Then a new value to variable "*factor*" is assigned again. It is calculated using formula: factor + 0.0003 * humidity + 0.00125 * temp.
A new value to variable "*progress*" is assigned. It is calculated using formula: progress + factor. If "*factor*" is less than -1 and "*statuss*" is greater than -12 then it reduces variable "*statuss*" by 1, sets zero to the variable "*factor*" and calls procedure "*grow_down*".
If "*factor*" is greater than 1 and "*statuss*" is less than -1 then it increases the variable "*statuss*" by 1, sets zero to the variable "*factor*" and calls procedure "*grow_up*". If "factor" is less than -1 or greater than 1 then zero to variable "*factor*" is assigned.

Such a simple interactive simulation model is easy for operation, but sometimes very useful for decision-making.

During validation of the GRASS Logo model the chi-square and Kolmogorov-Smirnov tests given satisfactory results, which approves that the model of course is not absolute precise, but can be used for forecasting credible enough.

6 Conclusion

Sociotechnical systems are manageable systems respecting human and/or biologic and/or environmental factor role, mutual assistance, and influence on joint functioning of the both systems – technical and social.

Tourism systems are typical samples of the sociotechnical systems.

To select suitable modelling environment some requirements must be respected. Those are the transparency of the project designing and running for non-programmers and applicability for analysis not only classic logistics processes, but also social phenomena or tasks of environmental protection.

Selected tools LISTechnology, EXTEND and NetLogo correspond to the requirements mentioned above and allow elaborating the multi-level simulation model for tourism objects operation analysis at the Ligatne Natural Trails.

7 Acknowledgement

The presented activity is supported by the grant No.2006/11 of the Latvian Ministry of Education and Science "Simulation Tools EXTEND and NetLogo use for Ecosystems Analysis".

References:

[1] Gauja National Park, http://www.gnp.lv/eng/index.htm, 14.09.2006.

[2] Gauja National Park News, Tourism Newspaper, Cesis: A/S Preses Nams, No.1, 2003.

[3] E. Ginters, LISTechnology, in *Simulation and Information Systems Design: Applications in Latvian Ports*, /E.Bluemel and L.Novitsky, eds., ISBN 9984-606-73-2, Publishing House JUMI Ltd., 2000, pp. 39-41.

[4] R. J. A. Buhr, *Practical Visual Techniques in System Design: with Applications*. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1990.

[5] Extend Overview, http://www.imaginethatinc.com, 14.09.2006.

[6] D. Krahl, The EXTEND Simulation Environment, in *Proceedings of the 2002 Winter Simulation Conference*/ E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, eds., 2002.

[7] U. Wilensky, NetLogo, Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL, 1999 and http://ccl.northwestern.edu/netlogo/.

[8] Agentsheets, Inc., Agentsheets, http://agentsheets.com/, 06.08.2003.

2. Ginters, E., Silins, A., & Andrusaitis, J. (2007). Communication in Distributed Simulation Environment. In *ICOSSSE'07: Proceedings of the 6th WSEAS international conference on System science and simulation in engineering* (pp. 217-221). Stevens Point: World Scientific and Engineering Academy and Society. doi: 10.5555/1974442.1974475 Izgūts no http://www.wseas.us/e-library/conferences/2007venice/papers/600-139.pdf

Communication in distributed simulation environment

EGILS GINTERS, ARTIS SILINS
Sociotechnical Systems Engineering Institute
Vidzeme University College
4 Cesu Street, Valmiera LV-4200
LATVIA
http://www.va.lv

JANIS ANDRUSAITIS
Ligatne Natural Trail
Baznicas Street 3, Sigulda LV-2150
LATVIA
http://www.gnp.gov.lv

Abstract: - Growing complexity of the systems determines the important rule of the simulation, because specification possibilities of classical analytical methods are quite limited. The analysis and operation of the real sociotechnical systems ask for cooperation among experts of technical and social (environmental) systems. Serious amount of simulation tools and simulators are created for modelling technical as well as social systems, nevertheless designing of heterogeneous and distributed simulation models is still difficult. One of the reasons is lack of suitable techniques for communications and synchronization of the various models inside the joint systems model. The authors offer the CORBA solution for communication inside distributed simulation model of Ligatne Natural Trails - an integrated tourism object for recreation and ecosystems research.

Key-Words: - Agent-based simulation, Discrete event systems, Distributed simulation, Communication environment, Sociotechnical systems, EXTEND, NetLogo, CORBA, HLA

1 Introduction

Growing variety and complexity of systems determines the important rule of the simulation, because specification possibilities of classical analytical methods are quite limited. Moreover, solutions of designed models in real time are close to problematic.

Changes in thinking and understanding supported by the growing computational performance allow approximating the actors of engineering and social knowledge step by step. Although with displeasure, they must confess that analysis and operation of the real sociotechnical systems, especially stochastic, cannot be substantial without equal notice of both technical factors and social (environmental, biologic) phenomena.

The technical systems are mostly closed, but self-organization and changes in the structure of the model during the session are not typical for them. Unlike the social systems, whose are cognitive and the result of the activity can depend on the new knowledge obtained during the previous iteration and huge amount of important perturbations and inputs. Serious amount of simulation tools and simulators are created for modelling technical as well as social systems allowing to design continuous, discrete, time-driven, event-driven and other models, nevertheless designing of heterogeneous simulation models is still difficult. One of the reasons is lack of suitable techniques for communications and synchronization of the various models inside the joint systems model.

2 Simulation of the Ligatne Natural Trails

The Gauja National Park (Latvia) was founded in 1973 [1]. It focuses mainly on nature protection and educational and leisure tourism. The Ligatne Natural Trails was founded in 1975 as an integrated tourism object for recreation and ecosystems research giving possibility for introduction with flora and fauna, which is typical for Latvian regions. The Ligatne Natural Trails is a part of the Gauja National Park. There exist different routes ensuring movement the visitors either by foot or by car. The total amount of visitors and cars are permanently growing. The problem mentioned above asks for expanding of parking and traffic dispatching. Permanently growing amount of visitors makes pressure on functioning of ecosystem and regeneration possibilities of lawns in sightseeing places. Therefore, potential workload forecasting of the tourism object, which depending of tourists flows intensity, becomes important. In 2006 two separate simulation models were elaborated [2] (see Fig.1).

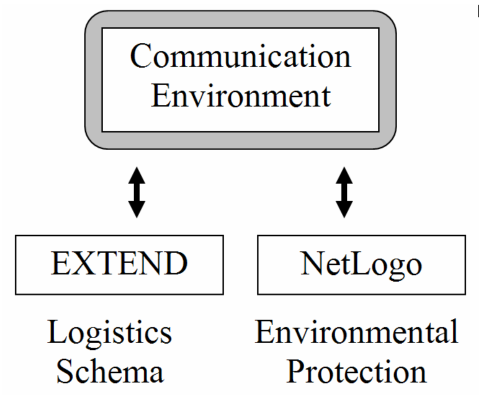


Fig. 1. Structural model of simulation system of the Ligatne Natural Trails

First - for throughput analysis of the trails and scenic routes, and planning the parking places. Functional model in this case was described as discrete event system and was simulated in EXTEND [3]. The second model was provided for estimation of other critical resource as capacity and quality of the sight place. Under the quality we understood the capabilities of regenerating of the nature resources (lawn, for instance) in given area influenced by the flow of the tourists with predefined intensity. In this case agent based approach was used and NetLogo [4] environment was selected. Only one problem still remained related with communication and synchronizing the separate models located on different places and operated in the real time.

EXTEND [3] is modelling environment supporting modelling, analysis and optimisation of the discrete-event and continuing processes. The environment is relatively open and well-linked with MS Office. It is possible to create user's component's using embedded ModL language, which is similar to C language. EXTEND allows the communications with ODBC, ActiveX and C classes.

NetLogo [4] is multi-platform multi-agent programming and modelling environment designed mostly for analysis of environmental systems and social phenomena. NetLogo is created in Java ensuring the platform independent environment, possibilities of object-oriented structuring and good programming options. NetLogo allows designing different extensions for communication with outer environment.

To ensure interoperability and information exchange among the models the convenient communication environment is necessary.

3 Communication environments

Some communication environments and algorithms of collaboration of the distributed systems and agents were analysed to find more convenient of them by the following criterions: performance of the operation, less-consuming contribution for connecting the separate models to the communication environment and maintenance possibilities, and support of the solution. The following environments have been selected:

Aggregate Level Simulation Protocol (ALSP) [5] – both software and a protocol, is used by the United States military to link analytic and training simulations. The main components of ALSP are Infrastructure Software for distributed runtime simulation support and management and Interface – the set of data exchange message protocols for interaction among objects represented in different simulations. Really ALSP is predecessor of the HLA (High Level Architecture) [6];

Common Object Request Broker Architecture (CORBA) [7] – the standard elaborated for joint communication the programmes written in different languages. To achieve this aim the interfaces of the modules must be specified in conformity with the requirements of the Interface Definition Language (IDL). IDL specifications are accessible for C, C++, Java, COBOL, Smalltalk, Ada, Lisp and Python, which made this mechanism well suited enough. Nevertheless, the CORBA environment asks for centralised synchronisation that made distributed simulation systems vulnerable, but it is important only in some specific cases;

The Foundation for Physical Intelligent Agents (FIPA) specifications [8] are a collection of standards which are intended to promote the interoperation of heterogeneous agents and the services that they can represent. Many of the ideas originated and developed in FIPA are now coming into sharp focus in new generations of Web/Internet technology and related specifications. FIPA standards for agents and multi-agent systems were officially accepted by the IEEE in 2005. It was decided to look on those specifications into the wider context of software development and integrate with non-agent technologies. In close future this approach would became one of more exciting ways for communication of the different systems.

High Level Architecture (HLA) [6] is concept of the architecture for distributed simulation systems. HLA ensure interoperability and reuse among simulations. It consists of Rules that simulations (federates) must follow to achieve proper interaction during a federation execution; Object Model Template (OMT) that defines the format for specifying the set of common objects used by a federation (federation object model), their attributes, and relationships among them; Interface Specification (IFSpec), which provides interface to the Run-Time Infrastructure (RTI), which can be distributed and ties together federates during model execution. The distributed time management can be done, because all federates' nodes directly undertake synchronization roles. Therefore, the total simulation is more quickly and the system is safer, unfortunately, implementation is more complex and laborious.

In addition, each of the environments and the protocols have their own lacks, nevertheless, respecting the criterions of the selection declared above the most convenient environment would be CORBA.

4 CORBA use for simulators collaboration

The CORBA architecture is shown in Fig. 2, where the following items are mentioned [9]:

Object – a programming entity that consists of an identity, an interface, and an implementation, which is known as a Servant;

Servant - an implementation entity that defines the operations supported by CORBA IDL interface. Servants can be written in a variety of languages, including C, C++, Java, Smalltalk, and Ada;

Client - the program that invokes an operation on an object implementation;

Object Request Broker (ORB) provides a mechanism for transparently communicating client requests to target object implementations. The ORB simplifies distributed programming by decoupling the client from the details of the method invocations. When a client invokes an operation, the ORB is responsible for finding the object implementation, transparently activating it if necessary, delivering the request to the object, and returning any response to the caller;

ORB Interface - a logical entity that may be implemented as one or more processes or a set of libraries. To decouple applications from implementation details, the CORBA specification defines an abstract interface for an ORB;

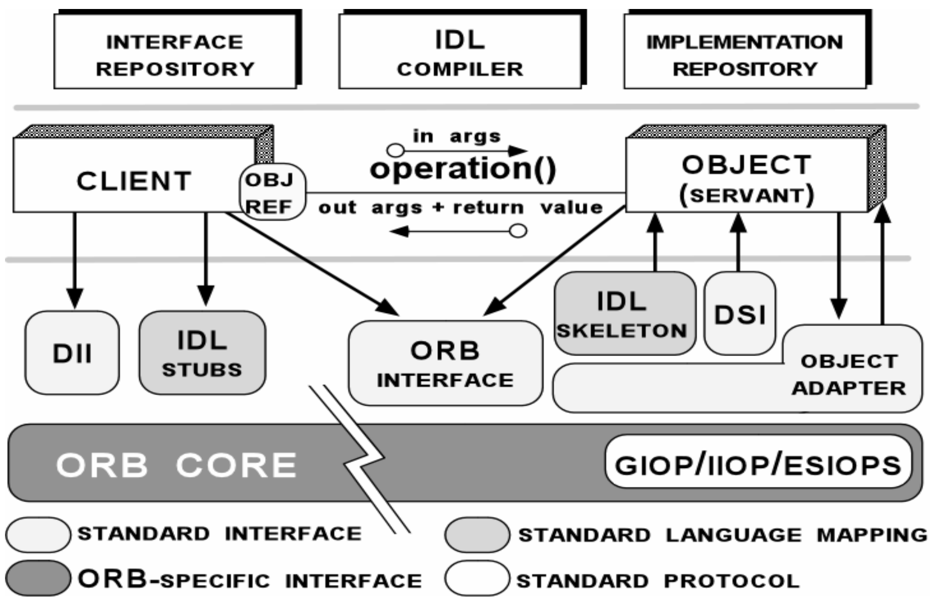


Fig. 2. CORBA Object Request Broker architecture [9]

CORBA IDL stubs and skeletons - the „glue" between the client and server applications, respectively, and the ORB. The transformation between CORBA IDL definitions and the target programming language is automated by a CORBA IDL compiler;

Dynamic Invocation Interface (DII) allows a client to directly access the underlying request mechanisms provided by an ORB. Applications use the DII to dynamically issue requests to objects without requiring IDL interface-specific stubs to be linked in;

Dynamic Skeleton Interface (DSI) allows an ORB to deliver requests to an object implementation that does not have compile-time knowledge of the type of the object it is implementing. DSI is the server side's analogue to the client side's DII;

Object Adapter - assists the ORB with delivering requests to the object and with activating the object. An object adapter associates object implementations with the ORB;

Internet Inter-ORB Protocol (IIOP) is protocol provided for interoperability between CORBA-compliant ORBs. IIOP is the TCP/IP transport mapping of a *General Inter-ORB Protocol* (GIOP). IIOP enables requests to be sent to networked objects. The CORBA interoperability architecture also accommodates communication using optional *Environment-Specific IOPs (ESIOPS).*

Designing of distributed simulation model of the Ligatne Natural Trails therefore involved three steps (see Fig. 3):

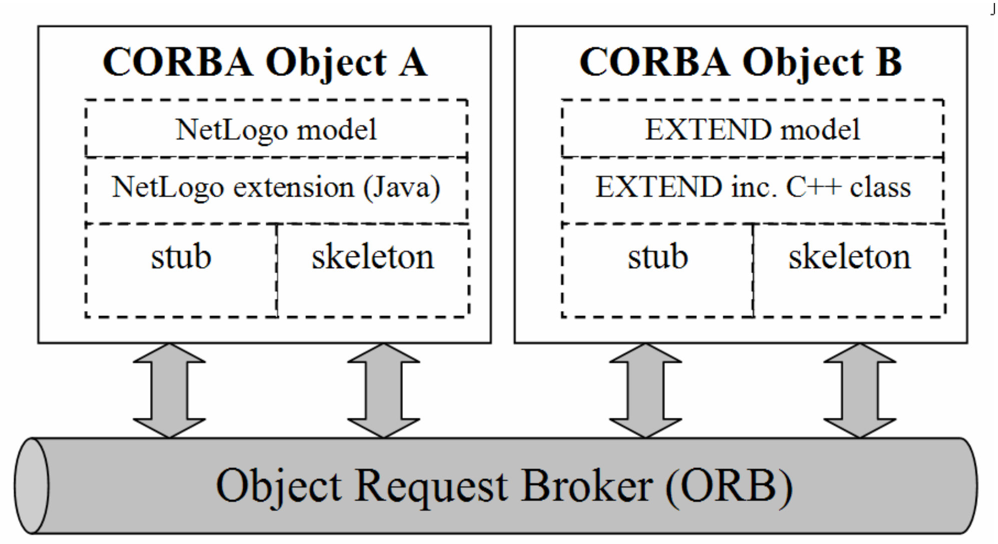


Fig. 3. NetLogo and EXTEND integration via CORBA

- The specific extension for NetLogo environment for communication between models and outer space was designed. It was done in JAVA language in conformity with NetLogo requirements;
- The similar extension was designed for discrete-event simulation environment Extend. It was done in C++, because EXTEND supports the implementation of C++ classes;
- Integration of the constructions designed above in CORBA architecture.

In Fig. 3 two CORBA objects are shown marked by A and B and involving NetLogo and EXTEND models discussed previously [2].

Each object has "stub" and "skeleton", where "stub" is used for client functions support giving possibilities for connecting to the server.

Otherwise, "skeleton" allows to the object the server's functions implementation granting answering to the requests. NetLogo extension in Java and EXTEND extension in C++ ensure collaboration of the simulators with CORBA ORB environment.

5 Conclusion

Sociotechnical systems are manageable systems respecting human and/or biologic and/or environmental factor role, mutual assistance, and influence on joint functioning of the both systems – technical and social [10].

Serious amount of simulation tools and simulators are created for modelling technical (EXTEND, ARENA, Witness, AutoMod, Simul8, STELLA etc.) [11] as well as social systems (NetLogo, AgentSheets, SWARM, RePast, MIMOSA etc.) [12], nevertheless designing of heterogeneous simulation models is still difficult. One of the reasons is lack of suitable techniques for communications and synchronization of the various models inside the joint systems model.

Some different protocols and methods (CORBA, HLA, FIPA, ALSP and other) [13, 14] exist for elaboration the communication environment, unfortunately any of it has their own disadvantages and right selection is still problematic.

Because no solutions are useable without programming knowledge then the authors offer the CORBA (Common Object Request Broker Architecture) [15] for integration of the separate models operating inside the goal model of the heterogeneous system. The authors consider that given approach, which is practically checked in the Ligatne Natural Trails simulation system, is more convenient by criterions of the performance of operating, possibilities for the maintenance and support, and labour-consuming as well.

6 Acknowledgement

The presented activity is supported by the agreement No. 2007/1-17/26 of the Latvian Ministry of Education and Science "COMMUNICATION ENVIRONMENT OF HYBRID SIMULATION SYSTEMS".

References:

[1] Gauja National Park, http://www.gnp.lv/eng/index.htm, 14.09.2006.

[2] E. Ginters and A. Silins, Multi-level approach for environmental systems modelling in the Ligatne Natural Trails, in *WSEAS Transactions on Systems*. ISSN 1109-2777. Issue 4, Volume 6, April 2007, 795-801.

[3] D. Krahl, The EXTEND Simulation Environment, in *Proceedings of the 2002 Winter Simulation Conference*/ E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, eds., 2002.

[4] U. Wilensky, NetLogo, Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL, 1999 and http://ccl.northwestern.edu/netlogo/, 11.09.2007.

[5] What is ALSP? http://ms.ie.org/alsp/questions/What%20is/index.html, 11.09.2007.

[6] K. Perumalla, Tutorial. Handling Time Management Under the High Level Architecture, 2006, http://www.ornl.gov/~2ip/doc/perumalla-tutorialslides-iitsec06.pdf, 12.09.2007.

[7] Welcome to the CORBA Directory. http://corba-directory.omg.org/, 11.09.2007.

[8] The Foundation for Physical Intelligent Agents. http://www.fipa.org, 12.09.2007.

[9] Overview of CORBA, http://www.cs.wustl.edu/~schmidt/corba-overview.html, 12.09.2007.

[10] E. Ginters and A. Silins, Modelling tools for environmental systems analysis in the Gauja National Park, in *Proceedings of 6th WSEAS International Conference on Applied Computer Science (ACS'06)*, ISSN 1790-5117, ISBN 960-8457-57-2, Tenerife, Puerto de la Cruz, Spain, December 16-18, 2006, 194-198.

[11] Verbraeck, Chapter 8. Simulation, in E. Ginters (ed). *Logistic Information Systems*, Part 2. Jumi, Riga, Latvia, ISBN 9948-30-021-8. 2002, p. 100-133.

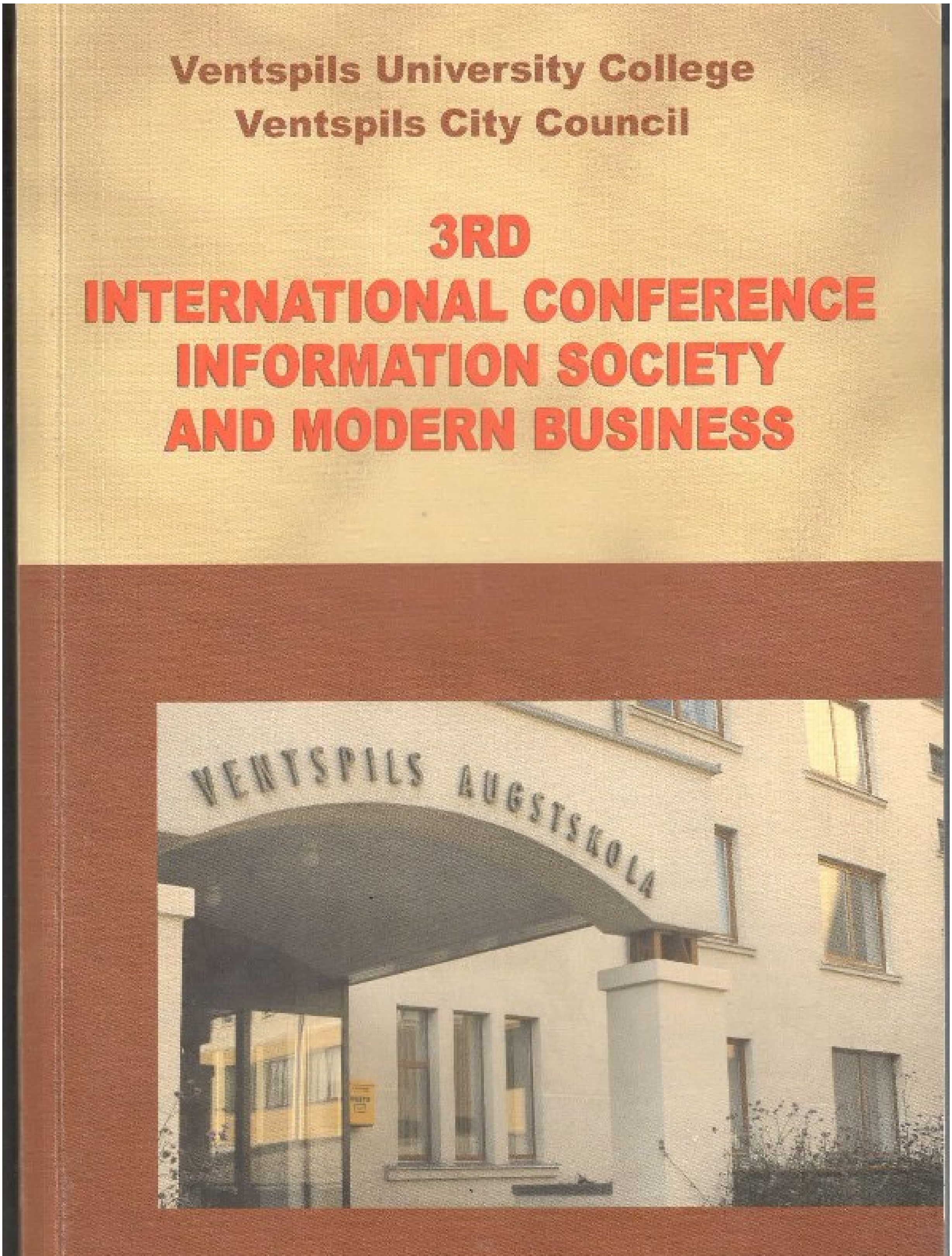
[12] N. Gilbert and K. G. Troitzsch, *Simulation for the Social Scientist*, Open University Press, 2 edition, ISBN 978-0335216017, 2006, P.312.

[13] S. Strassburger, The road to COTS-interoperability: from generic HLA-interfaces towards plug-and-play capabilities, in *Proceedings of the 37th conference on Winter simulation*, December 03-06, 2006, Monterey, California.

[14] A. Verbraeck, Component-based Distributed Simulations. The Way Forward?, in *Proceedings 18th Workshop on Parallel and Distributed Computer Simulation*. (Kufstein, Austria, 16-19 May 2004), IEEE Computer Society Press, Los Alamitos, CA, 2004. pp. 141-148.

[15] F. Bolton, *Pure CORBA*, Sams, ISBN 978-0672318122, 2001, P. 944.

3. Ginters, E., & Silins, A. (2008a). Exchange Mechanisms in Distributed Simulation of Sociotechnical Systems. In *Information Society and Modern Business. Proceedings of 3rd International conference, 21-22 September 2007, Ventspils* (p. 362). Ventspils: Ventspils University College. Izgūts no https://doi.org/10.5281/zenodo.13925129


Ventspils University College
Ventspils City Council
3RD
INTERNATIONAL CONFERENCE
INFORMATION SOCIETY
AND MODERN BUSINESS
VENTSPILS AUGSTSKOLA

Ventspils University College
Ventspils City Council

3[RD] INTERNATIONAL CONFERENCE

INFORMATION SOCIETY AND MODERN BUSINESS

PROCEEDINGS

21-22 September, 2007, Ventspils

LiePA

3rd International Conference

"Information Society and Modern Business"

dedicated to the 10th anniversary of Ventspils University College

21-22 September, 2007, Ventspils, Latvia

Ventspils University College
Ventspils City Council

Scientific Committee:

Head of Editorial Board:

2007 marks the 10th anniversary of Ventspils University College. A long time has passed since September 1, 1997, when 100 students and 15 teachers commenced their first academic year in the newly founded college. Since that memorable day Ventspils University College has grown about 10 times – in addition to the initial Faculty of Economics and Business Administration and Faculty of Translation Studies a new Faculty of Information Technologies has been established. Ventspils University College offers various bachelor and master programms, and even doctoral studies. Since 1997 a number of research centers have been founded – Ventpils International Radio Astronomy Center, Engeneering Research Center, and Center for Applied Linguistics. There are also other modern structures like Technology Transfer Center and Business Incubator. In the beginning, the main focus was on the study process, today Ventspils University College gives much greater prominence to reaserch. At the present moment research activities are carried out in humanities, social sciences, and natural sciences, including research of crosscultural communication, applied linguistics, economics and business administration, investment theory and financial mathematics, information technologies and telecomunications, applied space technologies and radioastronomy.

In order to celebrate its 10th anniversary, Ventspils University College organized the 3rd international research conference „Information Society and Modern Business" with the following sections:

1. Business Administration and Management Technologies
2. Investments and Financial Management
3. New Industrial Technologies and Innovative Business
4. Development of Information Technologies: Problems and Solutions
5. Language Contacts, Translation and Interpreting, Crosscultural Communications
6. Space Technologies and Radioastronomy.

Authors are completely responsible for the content and formulation of their proceedings.

ISBN 9984-648-98-2

EXCHANGE MECHANISMS IN DISTRIBUTED SIMULATION OF SOCIOTECHNICAL SYSTEMS

EGILS GINTERS, ARTIS SILINS

Sociotechnical Systems Engineering Institute
Vidzeme University College
Cesu Street 4, Valmiera LV-4200, Latvia
Phone: +371-29266909
E-mail: egils.ginters@va.lv

Abstract
Growing variety and complexity of systems determines the important rule of the simulation, because specification possibilities of classical analytical methods are quite limited. The designing of heterogeneous simulation models is still difficult. One of the reasons is lack of suitable techniques for communications and synchronization of the various models inside the joint systems model. Some different methods and technologies (HLA etc.) exist. The author's discuses the CORBA (Common Object Request Broker Architecture) solution use for integration of the different separate models operating inside the goal model of the heterogeneous system.

Keywords: Simulation, sociotechnical systems, heterogeneous models, CORBA, HLA, EXTEND, NetLogo.

Introduction

Changes in thinking and understanding supported by the growing computational performance allow approximating the actors of engineering and social knowledge step by step. Although with displeasure, they must confess that analysis and operation of the real sociotechnical systems, especially stochastic, cannot be substantial without equal notice of both technical factors and social (environmental, biologic) phenomena.

The technical systems are mostly closed, but self-organization and changes in the structure of the model during the session are not typical for them. Unlike the social systems, whose are cognitive and the result of the activity can depend on the new knowledge obtained during the previous iteration and huge amount of important perturbations and inputs. Serious amount of simulation tools and simulators are created for modelling technical as well as social systems allowing to design continuous, discrete, time-driven, event-driven and other models, nevertheless designing of heterogeneous simulation models is still difficult due to lack techniques for communications and synchronization.

As one of the typical samples existing HLA (High Level Architecture) can be mentioned. HLA serves as standard solution for distributed simulation, but if simulation environment have not been designed to be compatible with HLA *a priori*, then it application is problematic, therefore as one of suitable solutions the CORBA (Common Object Request Broker Architecture) solution for integration of the different separate models operating inside the goal model can be used.

1. THE HETEROGENEOUS SIMULATION MODEL OF THE TOURISM OBJECT

The Ligatne Natural Trails [1] was founded in 1975 as an integrated tourism object for recreation and ecosystems research giving possibility for introduction with flora and fauna, which is typical for Latvian regions. There exist different routes for the visitors either by foot or by car. The total amount of visitors and cars are permanently growing. This asks for expanding of parking and traffic dispatching. Permanently growing amount of visitors makes pressure on functioning of ecosystem and regeneration possibilities of lawns in sightseeing places. Therefore, potential workload forecasting of the tourism object, which depending of tourists flows intensity, becomes important. In 2006 two separate simulation models [2] were elaborated, which involves logistics model described as discrete-event system and written in EXTEND and the model for regenerating possibilities forecasting specified as agent-based systems and designed in NetLogo (see Fig. 1). The models are provided for advising the operator-manager of the logistics system of the Ligatne Natural Trails.

The logistics schema consists of the set of routes with places for parking the cars. The total amount of parking places is limited. Therefore, only limited amount of cars can enter the park. The tourists travelling by cars are going until free chosen parking place and then attending the sightseeing places for animals watching. Capacity of sightseeing places also is limited even more exceeding the regeneration possibilities of the natural resources (lawns, for instance) is restricted. The regeneration possibilities depend on meteorological factors (temperature, humidity) and intensity of tourist's flow. The logistics model allows planning the parking places and generates input for NetLogo model aimed for regeneration possibilities forecasting. The trail manager respecting the outputs of the models can decides about limiting the cars and tourists flow entering the tourism object.

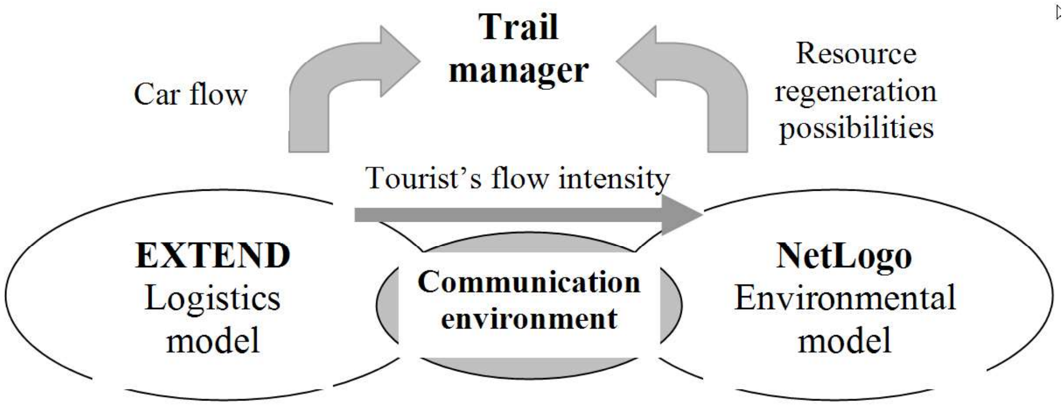


Fig. 1. Heterogeneous advising system of the trail manager

The task of the communication system is intensity data transfer from the logistics model to the environmental model of the object in real-time. There some different techniques can be used, but respecting some specific criterions as time, safety and cost of the solution. Safety and time are not critical because real-time demands in this case are flexible and safety of the data transfers is the same. Critical is solutions cost, because non-profit application is observed. It means that performance, of course, is important, while the price is critical, therefore HLA introduction would be unfortunately problematic [3]. Nevertheless, some other solutions can be used (ALSP, CORBA, FIPA etc.) [4, 5, 6].

2. CORBA - COMMON OBJECT REQUEST BROKER ARCHITECTURE FOR DISTRIBUTED COMMUNICATION IN HETEROGENEOUS MODELS

The CORBA [7] can be considered as one of most universal and wide spread application for information exchange in distributed software environment, but this solution seems to be used also for other objects communications support.

The main element in the CORBA architecture is Object Request Broker (ORB), which provides a mechanism for transparently communicating of the clients. The ORB simplifies distributed programming by decoupling the client from the details of the method invocations, because the ORB is responsible for finding the object implementation, transparently activating it, delivering the request to the object, and returning any response to the client (see Fig.2).

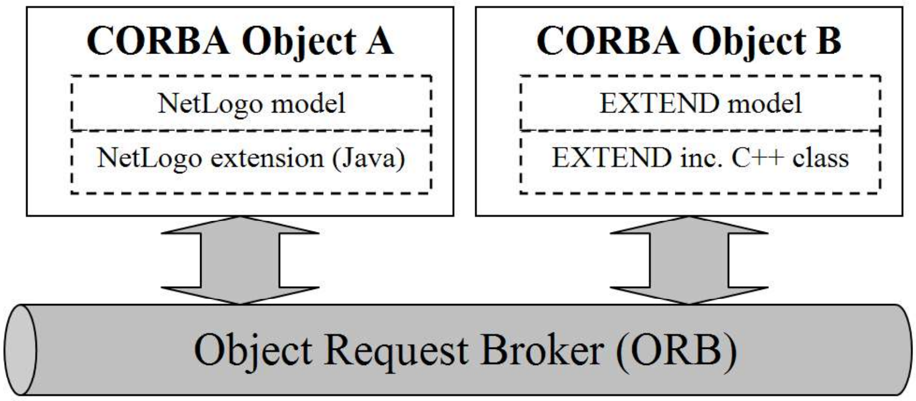


Fig. 2. ORB based communication in heterogeneous model

In this architecture two objects communicate, one of them is *servant*, but other is *client*. In Ligatne Trail model the servant is logistics model working in EXTEND environment, but client is environmental model operating in NetLogo. The EXTEND extension is written in C++, but NetLogo supplements – in Java (see Fig. 3).

The servant operates in following order:

- Creates and initializes the ORB object for interface needs;
- Receives the reference to Portable Object Adapter (POA) and activates POA. The POA issues references on the objects announced by servant. Client by these references can find the necessary object and uses the functions of the object;
- The servant is registered in the ORB object;
- Receives the reference to the servant registered;
- The main *Naming Service* is clarified. The *Naming Service* ensures converting the reference of the servant in textual form. Client can use the Naming Service for the reference resolving;

- Records the link and the name of servant in Naming Service;
- Waiting for client request.

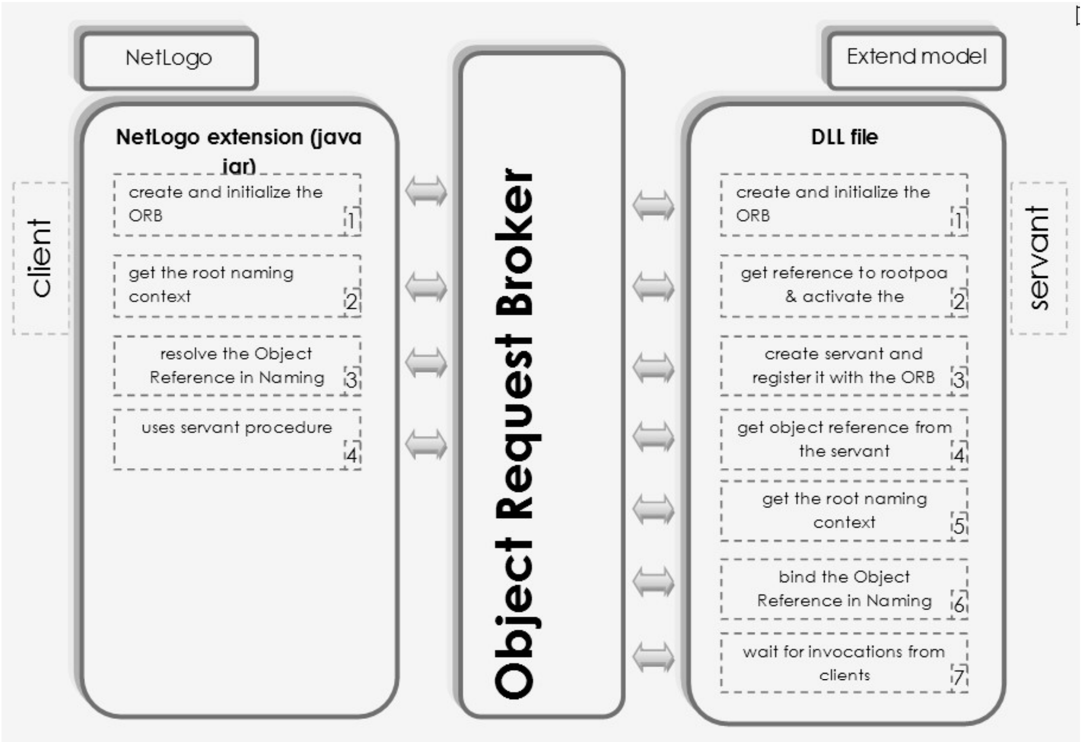


Fig.3. CORBA client's communication algorithm

The client operates in following order:

- Creates and initializes the ORB object for interface needs;
- The main *Naming Service* is clarified;
- Receives the reference from the *Naming Service* and necessary servant;
- By the reference can request the procedures of the servant.

The practical implementation approves that CORBA architecture can be used for the current distributed simulation model communication needs, but performance research and comparison with other tools (HLA etc.) is the next task.

Conclusions

The analysis of the sociotechnical systems determines the heterogeneous simulation models use. Because interdisciplinary knowledge and tools are necessary then modelling is inconvenient. Mainly the representatives of engineering direction vote for the technical systems modelling environment use, but great part of the representatives coming from social sciences possibilities of simulation and quantitative analysis raise the light surprise.

One of the main elements in heterogeneous system is communication subsystem, which support the data exchange and synchronising the processes. The functionality, performance parameters and safety are important. The different tools are exist HLA, CORBA etc, nevertheless estimation the availability each of the methods is important. The authors of the Ligatne Natural Trails tourism object management system recommends the CORBA solution use, because safety and performance parameters weight in total integrated criterion for estimation of the communication subsystem is not critical.

References

[1] Gauja National Park, http://www.gnp.lv/eng/index.htm, 14.09.2006.

[2] E. Ginters and A. Silins, Multi-level approach for environmental systems modelling in the Ligatne Natural Trails, in *WSEAS Transactions on Systems*. ISSN 1109-2777. Issue 4, Volume 6, April 2007, 795-801.

[3] Pitch. Make your systems work together. Pitch pRTI 1.3. http://www.pitch.se/products/hla-13-products/pitch-prti-13.html, 19.10.2007.

[4] What is ALSP? http://ms.ie.org/alsp/questions/What%20is/index.html, 11.09.2007.

[5] F. Bolton, *Pure CORBA*, Sams, ISBN 978-0672318122, 2001, P. 944.

[6] The Foundation for Physical Intelligent Agents. http://www.fipa.org, 12.09.2007.

[7] Overview of CORBA, http://www.cs.wustl.edu/~schmidt/corba-overview.html, 12.09.2007.

4. Ginters, E., & Silins, A. (2008b). Simulation Data Exchange in Distributed E-learning Environment. In *Proceedings of the 4th WSEAS/IASME International Conference on Education Technologies (EDUTE'08)* (pp. 138-143). Corfu: WSEAS. Retrieved from http://www.wseas.us/e-library/conferences/2008/corfu/edute/edute23.pdf

Simulation Data Exchange in Distributed E-learning Environment

ARTIS SILINS, EGILS GINTERS
Sociotechnical Systems Engineering Institute
Vidzeme University College
Cesu Street 4, Valmiera LV-4200
LATVIA

Abstract: - Nowadays in training systems the simulation applications frequently are used allowing economizing significant funding. Sometimes the use of distributed and non-homogeneous simulation environments is typical. To ensure the synchronizing and data interchange the adequate communication environment is necessary. As one of the most popular environments the HLA could be nominated, however significant costs of the commercial versions asking for other and more affordable environments. One of those is CORBA. The aim of the article is analysis and comparison of HLA and CORBA environments from efficiency view.

Key-Words: - Simulation, e-learning, CORBA, HLA, Distributed simulation

1 Introduction

Nowadays e-learning and consulting systems have multilevel structure involving different items provided for the training, decision making, managing and control, and other processes. The architecture of the e-learning environment often is distributed.

At the level of decision making the consolidation of all data is necessary. Therefore, the demand for the communication environment which ensures the information exchange with appropriate efficiency is important. During the last years the significant role in training systems designing have the simulation models, which serve for presentation of the physical objects and processes in virtual environment. Simulation allows reducing the training time and costs, and the same time increases the safety.

Frequently in such training systems the virtual and augmented reality (VR/AR) solutions and the different simulation environments are used. For instance, in decision making model of Ligatne Natural Trails Park for the natural resources use and workload research two separate simulation environments were used: NetLogo - for the analysis of the workload of the natural resources and Extend – for the logistics research. In GrassLogo model the flow of cars and tourists generated at Extend level serves as initial conditions for the decision making about further exploiting of the natural resources supported by NetLogo model [1]. NetLogo is agent-based simulation tool, but Extend is provided for discrete-event systems simulation [2, 3]. Important question in this case is information transfer from Extend model to NetLogo.

The similar task (see Fig.1) arises when training scenario involves different simulators because technological process in manufacturing consists of different operations.

During the training session switching among the scenarios happen to find better approach for the training [4]. Each of the technological operations involving in the scenario and simulated would consists of some parts.

Any operation on the scene can be simulated by different simulator. For instance, for the background generation one kind of the simulation tools, for the objects visualisation other kind of tools are used, but the decision making is based on the results of other simulator operation. And the requirement for the real-time support is important in this case. The similar problem would be related with the multilevel simulation tasks implementation where each separate micro-analytical model runs in separate territory, but joint results are gathered at the State level.

The requirement for efficient mechanism of the information interchange among the different simulation environments is typical in all these cases.

Many solutions exist, but HLA [5] can be nominated as more popular one.

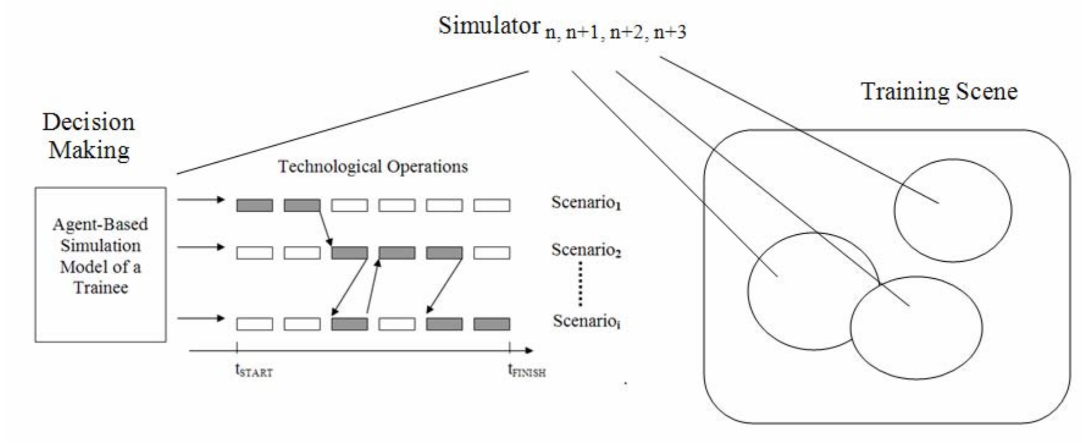


Fig. 1. Training in distributed simulation environment

However, the high costs of the commercial versions promote for new ways searching and using of the distributed environments provided for software modules communication, for instance CORBA [6]. The aim of the article is analysis and comparison of communication environments HLA and CORBA from the efficiency view.

2 CORBA vs. HLA

The Common Object Request Broker Architecture (CORBA) is provided as communication environment for collaboration and information interchange among different software objects. In CORBA applied software need the interface written in Interface Definition Language (IDL) to achieve successful communication. IDL specifications are elaborated for greatest part of today's programming languages as C, C++, Java, Python and other.

Two kinds of the objects exist in CORBA: Servants and Clients. The Servant is an implementation entity that defines the operations supported by CORBA IDL interface, but Client is the program that invokes an operation on an object implementation. The Object Request Broker (ORB) provides the mechanism for transparently transfer the client requests to target objects implementations. The ORB simplifies distributed programming by decoupling the client from the details of the method invocations. When a client invokes an operation, the ORB is responsible for finding the object implementation, transparently activating it if necessary, delivering the request to the object, and returning any response to the caller [7].

The basic principle of communication in CORBA prescribes possibilities for the communication with predefined server which parameters and location are well known for the client. If the server does not accessible then the communication will not be successful.

The High Level Architecture (HLA) is communication environment designed especially for the distributed simulation in conformity with the specifications elaborated by U.S. Department of Defence and IEEE [8]. Nowadays HLA cannot be considered as military technology, but as the communication standard for the distributed simulation models.

Communications activities in HLA are guided by HLA Rules, which ensure proper interaction of the federates in the federation and describe those responsibilities. To ensure the communication of the federates the interfaces also must be specified in conformity with the requirements of the Runtime Infrastructure (RTI). To create the federation three kind of the object model templates exist: federation (FOM), simulation (SOM) and management (MOM). The rules determine the mechanism by which from the separate federates (simulation programme in conformity with SOM rules) the federation (in conformity with FOM requirements) can be created, which will be managed under the joint principles (in conformity with MOM specifications). It means that each of federates must be supplied by the possibilities for RTI libraries access.

Unlike CORBA the communication protocol in HLA is different. The all federates can receive an information, but only federates which need it will use it further.

After short validation of both approaches is absolutely evident that without the specific skills in programming does not possible to create and affordable environment for the communications of different simulation models. And secondly, if CORBA solutions can be obtainable as shareware items then HLA cheapest ones exceed some thousands of dollars. Therefore, it is important to clarify the specific features of the communication mechanisms in CORBA and HLA to ensure the selection of the environment with appropriate parameters of performance and efficiency.

3 Distributed Communication Mechanism's Comparison

To compare the communication mechanisms the data transfer between two and more different simulation models is used. The test package consists of character type records. Each package can involve no more than 2000 records. The length of the record is 5000 UTF-8 characters. The length of the package varies with the step of 50 records. Each transmission session is repeated some times to achieve the statistic result approved by test of Kolmogorov-Smirnov.

In CORBA case the following communication scheme is used (see Fig. 2):

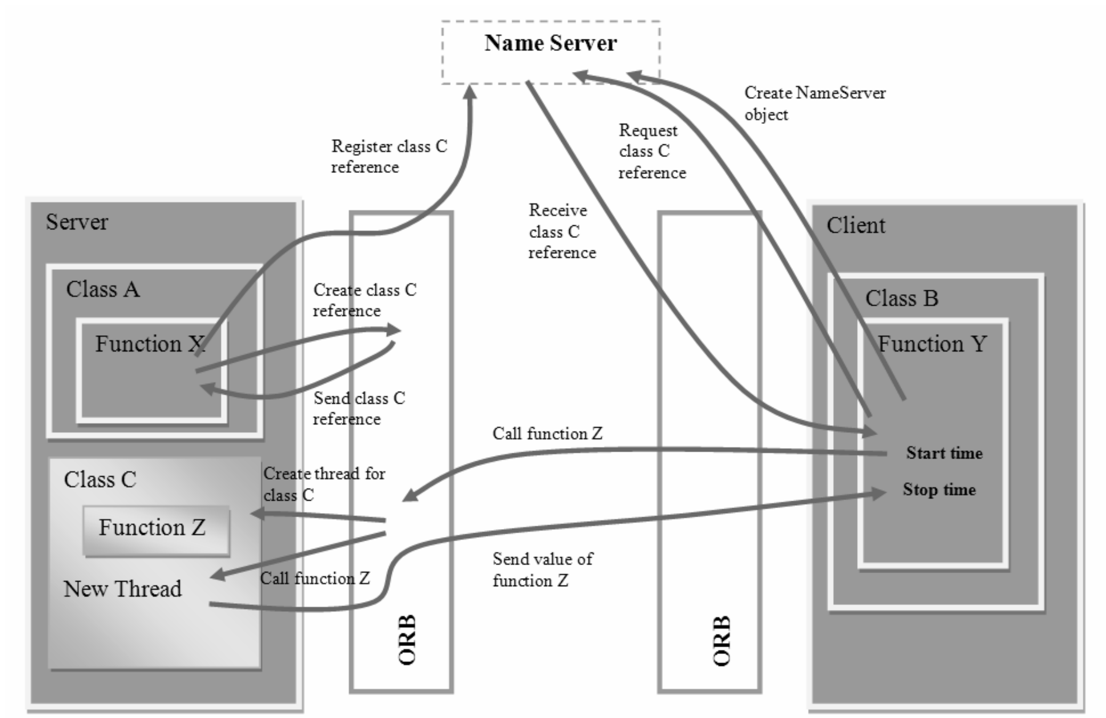


Fig. 2. Communication in CORBA environment

The essence of the session is calling (Start) the function Z deployed on the Server and receiving the reverted value (Stop). The session length is measured in milliseconds. Since the function Z belongs to the C class then Client must receive the reference of the C class before. Otherwise, the name of the C class must be registered on the NameServer. Then at the beginning of the session is possible to create the separate thread along which the information interchange is implemented. The communication in CORBA is based on the sharing of the resources, therefore in some client's case the separate threads are created for the parallel information interchange.

Unlike CORBA in HLA the information package is not marked for the predefined respondent. Some federates can receive the message, but only the federate which need it use it. In communication example (see Fig. 3) two federates exist – Server and Client. Each of federates must subscribe for further interactions before the transmission. The Server subscribes for the data receiving and sending the confirmation, but the Client – for the data sending and receiving the confirmation message. The DATA interaction consists of the information package and the name of the Client, but the interaction CONFIRM involves the name of the Client only. It means that after the sending of the information package the Client waiting for the interaction CONFIRM which approves that the information package is received. Interactions are prescribed by FOM which determines what kind of the interactions will be done.

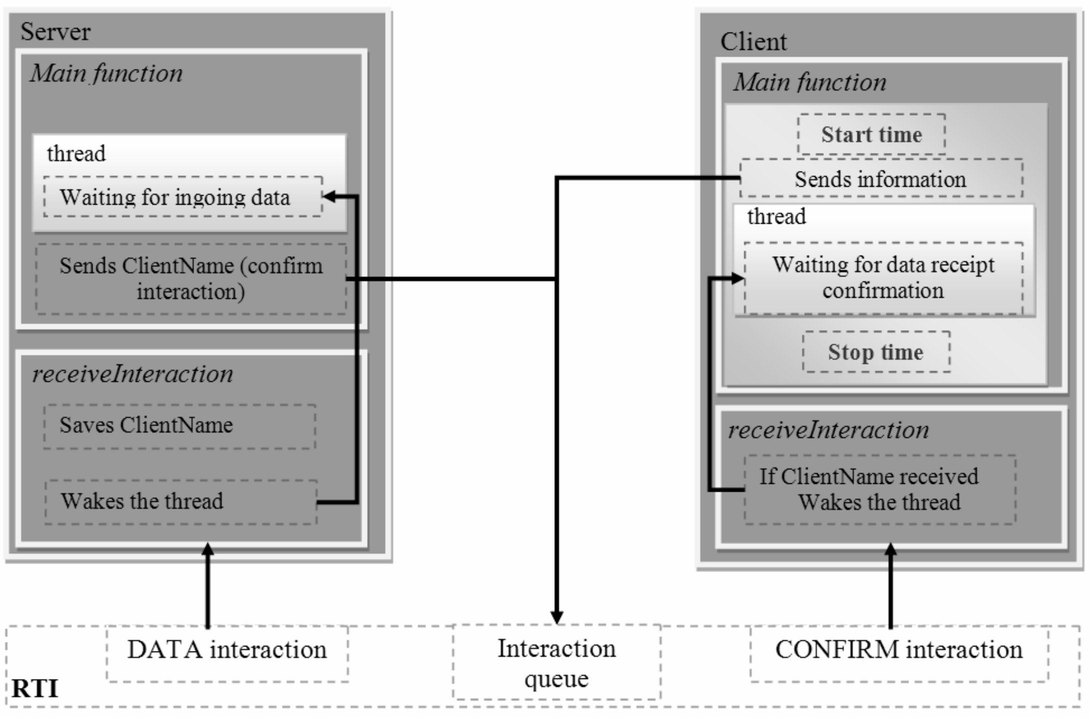


Fig. 3. Communication between HLA federates

All interactions are realised through RTI managing the interaction queue under FIFO rules. The collaboration in multi-client case is the same and the queues are used.

The results obtained (see Fig. 4) assure that in the case of two respondents no significant difference between CORBA and HLA communication environments.

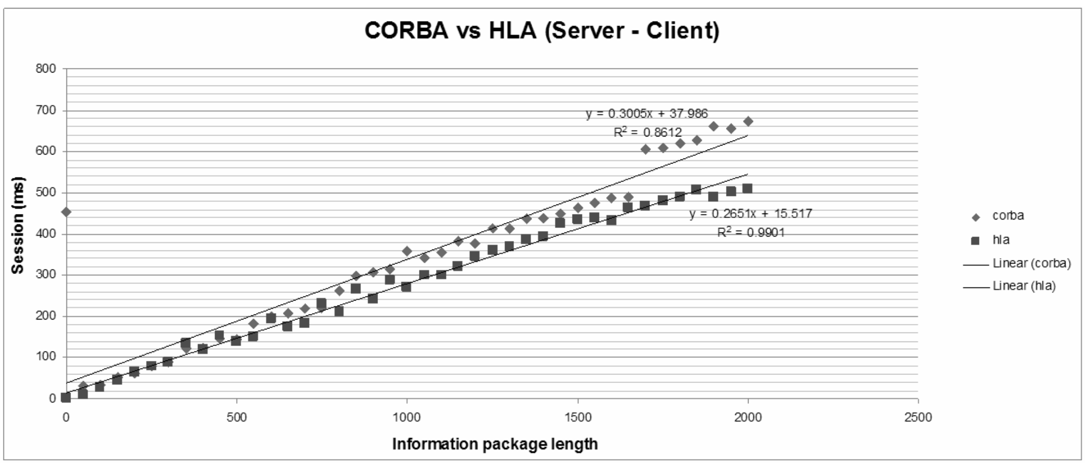


Fig. 4. HLA vs. CORBA comparison

The throughput is similar if the information amount is less than 500 x 5000 x UTF-8 characters.

However, increasing the volume of the information package, some differences appear and HLA wins in the competition with CORBA environment.

4 Conclusion

Nowadays in training systems the simulation applications frequently are used allowing economizing significant funding.

Sometimes the use of distributed and non-homogeneous simulation tools is very important. To ensure the synchronizing and data interchange the adequate communication environment is necessary.

As one of the most popular environments the HLA could be nominated, however significant costs of the commercial versions asking for other and more affordable environments. One of those is CORBA.

The research approves that in the case of transmitting the short information packages CORBA and HLA are close enough. However, in the case of audio-visual information interchange or in real-time applications HLA will be much better solution.

Further the research in multi-objects environment and under ad-hoc rules will be continued.

References:

[1] Ginters E., A. Silins and J. Andrusaitis, "Communication in distributed simulation environment", in *Proceedings of 6th WSEAS International Conference on Systems Science and Simulation in Engineering*, ISBN 978-960-6766-14-5, Venice, Italy, 21-23 November, 2007, pp. 217-222.

[2] Wilensky U., NetLogo, Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL, 1999 and http://ccl.northwestern.edu/netlogo/

[3] Krahl D., The EXTEND Simulation Environment, in *Proceedings of the 2002 Winter Simulation Conference*/ E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, eds., 2002.

[4] Ginters E., et al., VR-Solutions Use in Training of Glass Melting Oven Operators, in *10. IFF Wissenschaftstage: Virtual reality and Augmented Reality for Engineering, Testing and Operating Technical Systems*, Fraunhofer Institut Fabrikbetrieb und –automatisierung, ISBN 978-3-8167-7383-2, June 27-28, 2007

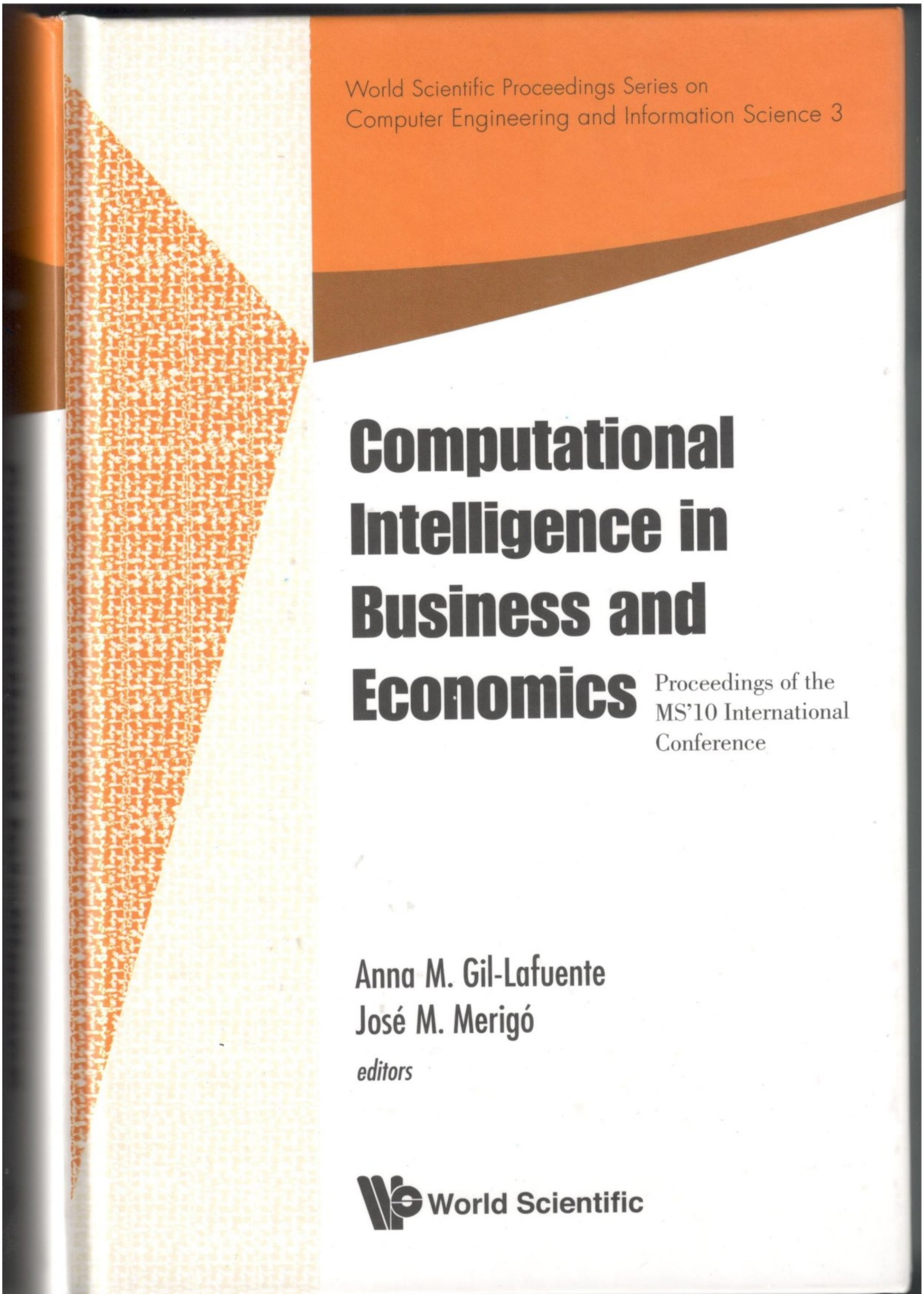
[5] Perumalla K., Tutorial. Handling Time Management Under the High Level Architecture, 2006, http://www.ornl.gov/~2ip/doc/perumalla-tutorialslides-iitsec06.pdf, 12.09.2007

[6] Bolton F., *Pure CORBA*, Sams, ISBN 978-0672318122, 2001, P. 944.

[7] Overview of CORBA, http://www.cs.wustl.edu/~schmidt/corba-overview.html, 05.09.2008

[8] Kuhl F., R. Weatherly and J. Dahmann, *Creating Computer Simulation Systems: An Introduction to the High Level Architecture*, ISBN 978-0130225115, Prentice Hall, 1999, p. 224

5. Silins, A., Ginters, E., & Aizstrauta, D. (2010). Easy Communication Environment for Distributed Simulation. In *Computational Intelligence in Business and Economics* (pp. 91-98). https://doi.org/10.1142/9789814324441_0014. Retrieved from https://www.researchgate.net/publication/282288675_Easy_Communication_Environment_for_Distributed_Simulation#fullTextFileContent


World Scientific Proceedings Series on
Computer Engineering and Information Science 3
Computational Intelligence in Business and Economics
Proceedings of the MS'10 International Conference
Anna M. Gil-Lafuente
José M. Merigó
editors
World Scientific

World Scientific Proceedings Series on Computer Engineering and Information Science contains proceedings from major, peer reviewed computer science, engineering, information technology and social science conferences with up-to-date coverage of new developments, methodologies and applications of new computer-information-processing methods such as artificial intelligence, soft computing, computational intelligence, as complementary to traditional mathematical and statistical methods in Computer Engineering and Information Science.

Computational Intelligence in Business and Economics

This book provides some of the most recent developments in Computational Intelligence applied to business and economics presented at the MS'10 International Conference, Barcelona, 15–17 July, 2010. It presents several new theoretical advancements and a wide range of applications in different business and economic areas including accounting, finance, management, marketing, sports, tourism, economics and politics, and also some applications related with engineering and modeling and simulation.

This book is very useful for researchers and graduate students interested in pursuing research in business and economics with an orientation to modern techniques for dealing with uncertainty such as those related with modeling and simulation and computational intelligence.

ISBN-13 978-981-4324-43-4
ISBN-10 981-4324-43-4

9 789814 324434

World Scientific
www.worldscientific.com
7925 hc

EASY COMMUNICATION ENVIRONMENT FOR DISTRIBUTED SIMULATION

ARTIS SILINS, EGILS GINTERS, DACE AIZSTRAUTA
Sociotechnical Systems Engineering Institute, Vidzeme University of Applied Sciences, Cesu Street 4, Valmiera LV-4200, Latvia

The development of simulation enables the researchers to explore more sophisticated problems at a more detailed level. This implies a necessity to use several models, or even different simulation tools. There are several solutions, but they are complicated, expensive and not suitable for people without specific knowledge. This paper describes a solution that would give the opportunity to ensure the communication between different parts of the joint model written by different simulation tools within HLA communication architecture at the same time being unsophisticated and appropriate for users without extensive technical skills. The easy communications system consists of three main elements simulation tool, communication adapter and HLA.

1. Introduction

The development of simulation enables the researchers to explore more sophisticated problems at a more detailed level. At the same time this implies a necessity to use several models, or even different modelling tools. The real systems often deal with complicated socio-technical systems that can not be explored at the sufficient level of quality with one modelling tool. Different simulation approaches (multi agent systems, discrete events simulation, system dynamics, etc.) usually envisage the use of different simulation tools (software). This makes the issue of their mutual communication. There are several solutions, but they are complicated, expensive and not suitable for people without specific knowledge. This is an important constraint that impedes the development of distributed simulation especially within social and behavioural sciences.

Hence this paper describes a solution that would give the opportunity to ensure the communication between simulation tools within HLA communication architecture at the same time being unsophisticated and appropriate for users without extensive technical skills.

2. Former Research in Field of Distributed Simulation – Lessons Learned

This paper is based on previous research made by authors. First of them examined multi level modelling of environmental systems in the Ligatne Natural Trails. Different routes ensure movement of visitors either by foot or by car. The total amount of visitors and cars is permanently growing and that leads to plans for expanding of parking and traffic dispatching. But at the same time permanently growing amount of visitors makes pressure on functioning of ecosystem and regeneration possibilities of lawns in sightseeing places. Therefore, potential workload forecasting of the tourism object, which depending of tourists flows intensity, becomes important. In 2006 two separate simulation models were elaborated [3].

First - for throughput analysis of the trails and scenic routes, and planning the parking places. Functional model in this case was described as discrete event system and was simulated in EXTEND [1]. The second model was provided for estimation of other critical resource as capacity and quality of the sight place. Under the quality we understood the capabilities of regenerating of the nature resources (lawn, for instance) in given area influenced by the flow of the tourists with predefined intensity. In this case agent based approach was used and NetLogo [10] environment was selected. Only one problem still remained related with communication and synchronizing the separate models located on different places and operated in the real time.

EXTEND [1] is modelling environment supporting modelling, analysis and optimisation of the discrete-event and continuing processes. The environment is relatively open and well-linked with MS Office. It is possible to create user's component's using embedded ModL language, which is similar to C language. EXTEND allows the communications with ODBC, ActiveX and C classes.

NetLogo [10] is multi-platform multi-agent programming and modelling environment designed mostly for analysis of environmental systems and social phenomena. NetLogo is created in Java ensuring the platform independent environment, possibilities of object-oriented structuring and good programming options. NetLogo allows designing different extensions for communication with outer environment.

To ensure interoperability and information exchange among the models the convenient communication environment was necessary.

In further research it was concluded that although serious amount of simulation tools and simulators are created for modelling technical (EXTEND, ARENA, Witness, AutoMod, Simul8, STELLA etc.) as well as social systems (NetLogo, AgentSheets, SWARM, RePast, MIMOSA etc.), nevertheless designing of heterogeneous simulation models is still difficult. One of the reasons is lack of suitable techniques for communications and synchronization of the various models inside the joint systems model. [2]

Thereafter test models were made to examine the possibilities to connect two different simulation tools using communication architecture. Two communication architecture environments were compared – CORBA and HLA – by experimenting with them and their capacity to work with certain load [2, 4]. Besides the technical comparison, it was concluded that one needs to have a good knowledge of programming otherwise it is impossible to organize and manage communication between simulation tools [2]. Therefore the aim of this paper is to analyze a new comprehensible or Easy Communication Concept and describe the mechanism for ensuring the communication of simulation tools for users without programming skills.

3. Communication Environment for Distributed Simulations

This section first describes the existent communication mechanisms and observed disadvantages. Afterwards the authors propose their communication system and describe its requirements in detail.

3.1. *Communication Mechanisms in Distributed Models*

One of them is HLA Blockset™ - a commercial product created by ForwardSim Inc. The HLA Blockset™ provides a library of Simulink® blocks specifically designed for the integration of HLA services into Simulink®. Using the HLA Blockset, you can connect your Simulink models to a distributed simulation using the High Level Architecture standard HLA 1.3 or IEEE 1516 [6]. But the product is supposed for the Simulink® modelling tool only.

Similarly in their research on distributed simulation systems Hibino *et.al* [5] uses similar approach. They propose a manufacturing adapter to connect manufacturing system simulators with HLA using a plug-in style and evaluate it using a hypothetical manufacturing system. To be more precise three major commercial based manufacturing system simulators as QUEST, SIMPLE++, and GAROPS are connected using the developed manufacturing adapter.

Both above mentioned examples share the central solution that permits to implement HLA functionality in the simulation tool. However to be able to use any of these communication systems, modelling expert needs to immerse oneself in HLA structure and principles of operation.

Therefore it is necessary to develop a communication adapter that would enable the co-operation between simulation tools without making the modelling expert to master the infrastructure of HLA in detail. The communication adapter should instead facilitate the work of modelling software user, so there is no need to think at the level of communication abstraction, but it is enough to define the name of communication channel and its properties (function and parameters).

3.2. *Requirements to Alternative Communication Mechanism*

The proposed communications system consists of three main elements: simulation tool, communication adapter and HLA. The main goal of this communication system is to provide a mechanism that does not demand from the user any special knowledge of HLA infrastructure nor the principles of its action nor programming skills. Further each of the elements and its role in the communication system is described.

3.2.1. *Simulation tool*

In order to be able to function in a communicative system, simulation tools have to be able to communicate in HLA and/or with different software. In this article strictly focus only one of these versions, namely, where simulation tools are not able to communicate in HLA, but can perform communication with other software (adapters) instead. In most cases this function is performed with some additional plug-ins, extensions or libraries. For example, NetLogo, that is a multi agent-modelling tool, uses different extensions [10], as well as discrete event modelling tool Extend that also allows using extensions [1].

But it is important that independent developers are allowed to create these extensions, thus promoting HLA communication. One should take into account that it is common for modelling tools to use extensions as functional libraries – i.e. they do not keep any live extension instances. That means there is no possibility to realise HLA communication in a direct way therefore an adapter is used. The adapter is carrying out the HLA communication and passing on the information to the extension of the simulation tool.

3.2.2. *HLA*

High Level Architecture (HLA) [8] is a concept of the architecture for distributed simulation systems. HLA ensures interoperability and reuses among simulations. It consists of rules that separate parts of distributed simulation model (federates) must follow to achieve proper interaction during a federation execution; Object Model Template that defines the format for specifying the set of common objects used by a federation (federation object model), their attributes, and relationships among them; Interface Specification, which provides interface to the Run-Time Infrastructure, which can be distributed and ties together federates during model execution. The distributed time management can be done, because all federates' nodes directly undertake synchronization roles. Therefore, the total simulation takes less time and the system is safer, unfortunately, implementation is more complex and laborious.

For this reason adapter that will ease the communication of simulation tools using HLA needs to be developed.

3.2.3. *Communication adapter*

Adapter is the central element from the perspective of this paper. It embraces communication from both sides – communication with simulation tool and communication with HLA. Adapter is like an interpreter that interprets HLA data to the format that is comprehensible for the simulation tool.

The communication among different simulation tools will be accomplished using HLA communication architecture. For the model to be able to perform its tasks in HLA environment, one needs an adapter. The communication adapter would provide the data exchange between the modelling tool and communication architecture. At the same time also the adapter needs to be easily manageable, so that the adjustment of simulation model would be possible for users without programming skills. The user has to be able to define understandable model communication parameters with the help of GUI, and another model has to be able to receive information according to pre-defined parameters.

3.2.4. *Description of communication systems operation using adapter*

The adapter is software module that is executed on the same machine as the simulation tool. The adapter will ensure the functions of a model within HLA architecture. HLA itself is used to make the functionality of these models and the results universal, so that they can be used by anyone that is able to connect to this federation.

The adapter (*Communication adapter* in Figure 1) may receive the data at any moment. This article does not refer to situations, where HLA Time Management ensures federation time advancing, as described by Ke Pan *et al.*[7], including the time advancing of the modelling tools.

The adapter should be able to store data and be ready to pass those to the simulation tool, whenever it requests for them. For this reason the data storage is used (*Data storage* in Figure 1). This data storage will ensure that the model receives all data in a time, even if it falls behind from other models. Additional reason for this solution would be the inability to inject data into running model. Therefore the receipt of data is organised by requests. See Figure 1 for a description of one model communication using adapter. The *Model* in Figure 1 represents the model that is created with a certain simulation tool or environment.

The communication between the model's library and the adapter is carried out using XML that ensures extensive and universal opportunities for data and communication description.

To ensure the communication with the model in this manner, the adapter has to look for communication requests at certain port at local host. Simulation tool will connect to the adapter with a request once in a certain period of time. This request is specified as XML-based document and hidden for end-user.

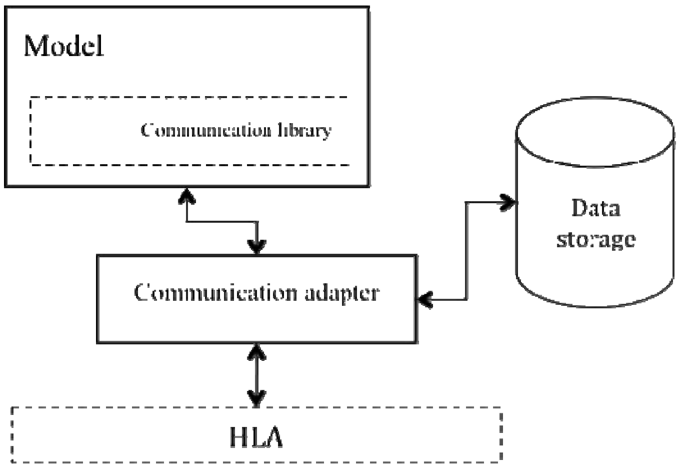


Figure 1. Model communication with adapter and data storage.

After the adapter returns data from data storage, which corresponds to the interaction for which the adapter has subscribed to in the HLA federation.

With the help of the adapter, the user can define information that he wants to interchange. The HLA sign-up configuration is accomplished with the help of adapter and has to be user-friendly. The adapter has to let the user to view all interactions of federation and all their properties, as well as the user needs to be enabled to subscribe to the interactions he is interested in with GUI.

When the user has signed up to an interaction, it has access to modelling tool. Further the user calls out the interaction in the modelling environment and receives data (the reception of data depends on the realization of extension). At the same time the user can define his interactions itself. It should be taken into account that the user has to be able to do this only with the input of the interactions name and parameters. In this case we assume that all parameters are defined with string data type, as this is very important when generating Federation Object Model (FOM). When the user has defined all parameters, then the adapter generates FOM and signs up to the federation (*adapter federer*). Every user-defined interaction is transmitted to a custom made federer – the configuration federer. This federer is located in another federation and it is responsible for FOM file. That means that configuration federer maintains the FOM file, according to which the modelling adapters create their federation.

Based on FOM the adapter connects to the federation (if there is no federation, then one is created), that functions according to the "rules" described in FOM. All interactions and their parameters are described in FOM. And also in conformity with FOM the adapters offer user functionality (published by adapters of other models) that can be used in simulation tools.

Configuration federer operates on the same machine where HLA. This federer operates independently from modelling federers, that means, modelling users are not responsible for its operations and do not have any direct impact on it.

4. Conclusions

Finally some important conclusions can be drawn regarding the distributed simulations and communication between models. First, it can be observed that the developers of simulation tools do not pay attention to the opportunities of models to communicate with other models, even of the same modelling environment. The paper outlines the structure of a comprehensible communication system and points out to its advantages.

Second, the existent mechanisms do not offer time synchronization option; therefore there is a need to invent data storage. Further research should be aimed at searching for such time synchronising options.

Third, future research should also seek for possibilities to ensure the transmission of more complicated data types.

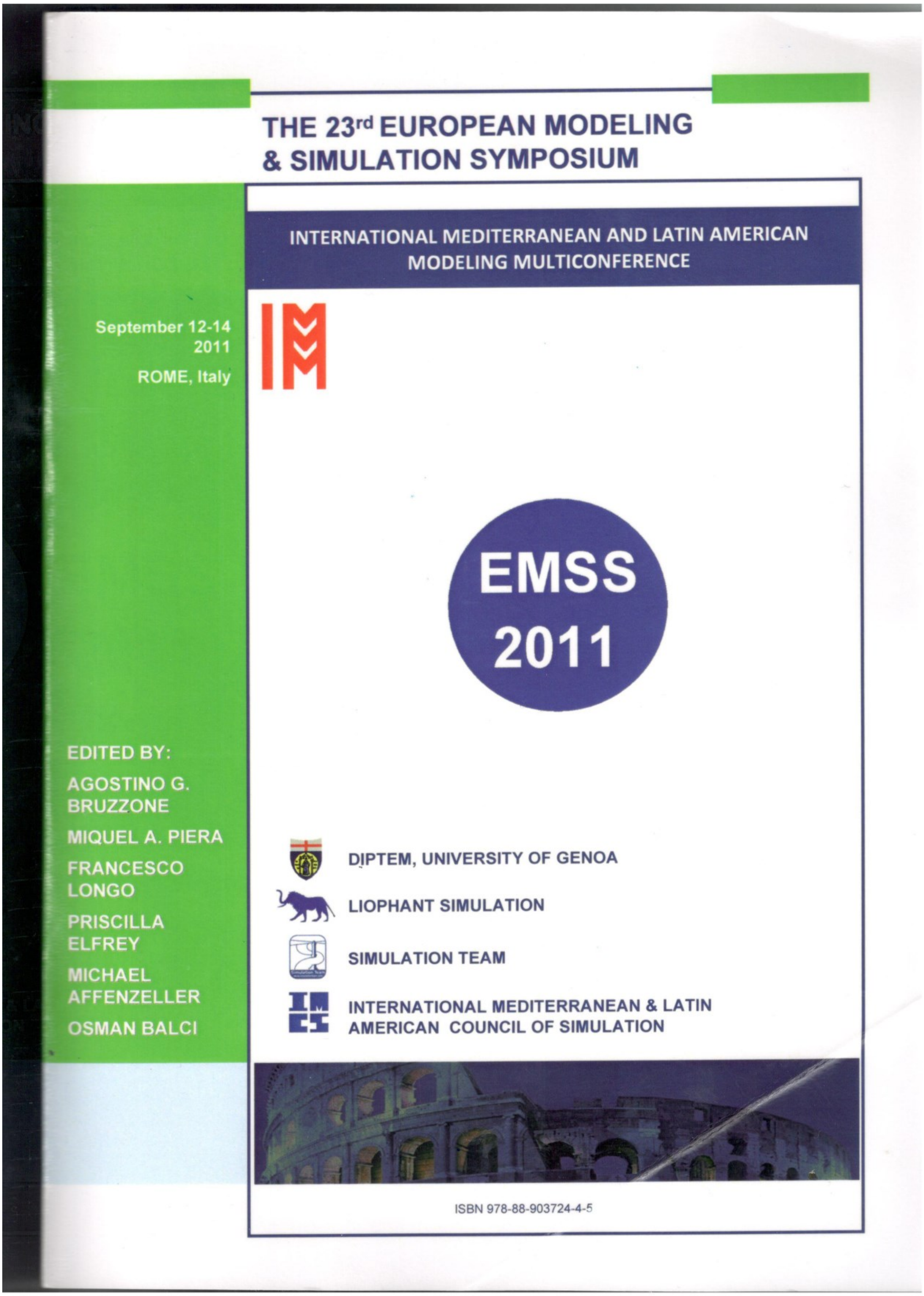
At the same time it should be noted that not all simulation tools offer the creation of extensions, therefore the comprehensible communication system described in this paper would not be applicable for such tools.

Therefore if it is observed that computational modelling is revolutionizing the social and organizational sciences [9], then there are new challenges for engineering scientists in terms of enabling this process and empowering the researchers from other disciplines.

References

1. D. Krahl, The EXTEND Simulation Environment, in Proceedings of the 2002 Winter Simulation Conference/ E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, eds.,(2002).
2. E. Ginters, A. Silins, J. Andrusaitis, Communication in distributed simulation environment, In Proceedings of 6th WSEAS International Conference on Systems Science and Simulation in Engineering, Venice, Italy, (2007), 217-221.
3. E. Ginters, A. Silins, Multi-level Approach for Environmental Systems Modelling in the Ligatne Natural Trails. WSEAS Transactions on Systems, Issue 4, Volume 6, (2007), 795-801.
4. E. Ginters, A. Silins, Simulation Data Exchange in Distributed E-learning Environment, Proceedings of the 4th WSEAS/IASME International Conference on EDUCATIONAL TECHNOLOGIES (EDUTE'08), Corfu, Greece, (2008),138-143.
5. H. Hibino, Y. Fukuda, Y. Yura, K. Mitsuyuki, K. Kaneda, Manufacturing adapter of distributed simulation systems using HLA, Simulation Conference, Proceedings of the Winter conf.,vol 2,(2002) 099-1107.
6. HLA Blockset™, ForwardSim Inc, High Tech Park of Québec, http://www.forwardsim.com/en/prod_hlab.php, (2009).
7. K. Pan, S.J. Turner, W. Cai, Z. Li, A Hybrid HLA Time Management Algorithm Based on Both Conditional and Unconditional Information, Parallel & Distrib. Comput. Center, Nanyang Technol. Univ., Singapore, (2008), 203-211.
8. K. Perumalla, Tutorial. Handling Time Management under the High Level Architecture, http://www.ornl.gov/~2ip/doc/perumalla-tutorialslides-iitsec06.pdf, (2009).
9. K.M. Carley, Computational organizational science and organizational engineering, Simulation Modelling Practice and Theory 10 (2002) 253–269.
10. U. Wilensky, NetLogo, Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL, http://ccl.northwestern.edu/netlogo/, (2009).

6. Ginters, E., Sakne, I., Lauberte, I., Aizstrauts, A., Dreija, G., Aquilar Chinea, R.-M., Merkuryev, Y., Novitsky, L., & Grundspenkis, J. (2011). Simulation Highway – Direct Access Intelligent Cloud Simulator. In *Proceedings of 23rd European Modelling & Simulation Symposium (EMSS 2011)* (pp. 62-72). Izgūts no https://doi.org/10.5281/zenodo.13925109


THE 23rd EUROPEAN MODELING & SIMULATION SYMPOSIUM
INTERNATIONAL MEDITERRANEAN AND LATIN AMERICAN MODELING MULTICONFERENCE
September 12-14 2011
ROME, Italy
EMSS 2011
EDITED BY:
AGOSTINO G. BRUZZONE
MIQUEL A. PIERA
FRANCESCO LONGO
PRISCILLA ELFREY
MICHAEL AFFENZELLER
OSMAN BALCI
DIPTEM, UNIVERSITY OF GENOA
LIOPHANT SIMULATION
SIMULATION TEAM
INTERNATIONAL MEDITERRANEAN & LATIN AMERICAN COUNCIL OF SIMULATION
ISBN 978-88-903724-4-5

Index

SIMULATION HIGHWAY – DIRECT ACCESS INTELLIGENT CLOUD SIMULATOR

Egils Ginters[(a)], Inita Sakne[(a)], Ieva Lauberte[(a)], Artis Aizstrauts[(a)], Girts Dreija[(a)], Rosa Maria Aguilar Chinea[(b)], Yuri Merkuryev[(c)], Leonid Novitsky[(c)], Janis Grundspenkis[(c)]

[(a)]Sociotechnical Systems Engineering Institute
Vidzeme University of Applied Sciences
Cesu Street 4-2A, Valmiera LV-4200, Latvia
[(b)]Universidad de La Laguna
Pabellón de Gobierno, C/ Molinos de Agua s/n., 38200 La Laguna, Spain
[(c)]Faculty of Computer Science and Information Technology
Riga Technical University
Kalku Street 1, Riga LV-1050

[(a)]egils.ginters@va.lv, [(b)]rosi@isaatc.ull.es, [(c)]merkur@itl.rtu.lv

ABSTRACT

Simulation nowadays plays an increasing role in the assessment of possible solutions and situation analysis. However, the tools used and previously created models are often incompatible and territorially distributed. Domain specialists lack the expertise and specific programming skills to use them effectively. The development of the Future Internet creates new challenges for simulation engineering by offering extended access. However, primarily a conception is needed, which would create unified access to the simulation architecture on the Future Internet. The paper discusses development of a new conceptual approach to simulation engineering, aiming to support domain experts in performing simulation of complicated socio-technical systems on the Cloud.

Keywords: simulation engineering, simulation highway, Future Internet, Cloud simulation

1. INTRODUCTION

The requirements for situations forecasting and transparency of the scenarios in manufacturing and politics before decision making are topical due to high losses of possible faults and impact on the society and environment.

One of the most effective methods to verify possible solutions while saving financial resources and minimizing security risks is simulation. Simulation is a critically vital component of decision making.

The world is varied, as are the methods of specification that correspond to core systems: continuous, discrete, determined, stochastic etc. Formal languages and analytical methods are used to describe these systems. However, the mentioned specifications can be done by an information technology (IT) professional or systems analyst rather than by a domain expert. All of these applied systems can be simulated by using a predefined set of instruments: discrete-event tools (DEVS), system dynamics (SD), an agent-based approach (ABM), multilevel models, micro analytical models etc. Naturally, all of these instruments have corresponding popular and concrete simulation tools which have their own modeling languages, formats and rules. Real socio-technical systems are not homogenous, therefore, to make a decision, multiple distinctive simulation models have to be combined in a unified environment. There are exist distributed communication environments like HLA that provide such functionality, however, that is not possible without significant financial investments, even more each case is a custom engineering solution.

The development of Future Internet, Internet of Things, Service oriented Architecture and Cloud computing creates a new challenge and possibilities for simulation engineering. The aim of this article is to discuss the project findings related to developing the new conceptual approach to simulation engineering on the Future Internet giving various domain experts an immediate possibility to specify different simulation cases as well as translate, distribute and implement these specifications on the Cloud through the Simulation Highway.

2. SIMULATION AND TOOLS

A system is a set of entities, real or abstract, comprising a whole where each component interacts with or is related to at least one other component and they all serve a common objective. In a system we can always find different types of organization in it, and such organization can be described by concepts and principles which are independent from the specific domain at which we are looking. Socio-technical systems are open to, and interact with, their environments, and that they can acquire qualitatively new properties through emergence, resulting in continual evolution (Von Bertalanffy 1976). A system would be defined as group of objects that are joined together in some regular interaction or interdependence

towards the accomplishment of some purpose (Banks 1996). Otherwise, a system can be defined as a collection of interacting components that receives input and provides output for some purpose (Chang 2004).
The socio-technical systems are tended to self-organization, cognition and continual evolution. The systems can be classified as physical and conceptual or abstract systems, open or closed, continuous or discrete systems, static or dynamic systems, linear or non-linear and deterministic or stochastic systems.
A model of the goal systems is an external and explicit representation of part of reality as seen by the people who wish to use that model to understand, to change, to manage and to control that part of reality (Pidd 1996). Model can be defined as a representation of a system for the purpose of studying the system (Banks 1996).

Simulation is the imitation of the operation of a real-world process or system over time (Banks 1998). Simulation modeling and analysis is the process of creating and experimenting with a computerized mathematical model of a physical system (Chang 2004). Simulation is the way how to research the model.

Nowadays different simulation technologies and mostly non-compatible set of simulation software tools are used. For example, some groups can be mentioned (Gilbert and Troitzsch 2006; A.Bruzzone, A.Verbraeck, E.Ginters et. al. 2002): System dynamics and world (large systems) (DYNAMO, VenSim, PowerSim, STELLA, iThink etc.); Queuing models (discrete-event systems) simulation software (DELSI 1.1., OMNET++, QSIM, QPR Process, EXTEND, SIMPLE++, ARENA, SIMUL8, AutoMod, WITNESS, AnyLogic etc.); Micro analytical simulation solutions (MICSIM, UMDBS, STINMOD, DYNAMOD, CORSIM etc.); Multilevel simulation tools (HLM, MIMOSE, MLwiN, aML etc.); Cellular automata CAFUN, LCAU, CASim, Trilife, JCASim, CellLab, CAGE, SITSIM etc.); Agent-based tools (AgentSheet, NetLogo, SWARM, RePast etc.); Learning and evolutionary models simulation tools (Artificial Neural Nets, Evolutionary models, Reinforcement Learning (Research simulators - Stuttgart Neural Network Simulator (SNNS), PDP++, JavaNNS, XNBC and the BNN Toolbox for MATLAB etc.; Data analysis simulators - Alyuda NeuroIntelligence, BrainMaker, EasyNN-plus, MATLAB Neural Network Toolbox, NeuralTools, Netlab, Palisade etc.; Component based development environments - JOONE, Peltarion Synapse and NeuroDimension, NeuroSolutions etc.). The simulation software tools (HLA, DIS etc.) ensure elaboration of distributed models. Sometimes also adjacent approaches like CORBA are applied and new distributed simulation environments are elaborated (Aizstrauts et.al 2010).

Unfortunately approaches are very different and simulation platforms and alphabets do not compatible. Up today selection of the simulation software tools are intuitive. Most part of the simulators cannot be used by domain decision makers in real-time and immediate way due to complexity, heavy architecture and special knowledge on programming and mathematics.

3. COMMUNICATION IN SIMULATION

The development of simulation enables the researchers to explore more sophisticated problems at a more detailed level. At the same time this implies a necessity to use several models, or even different modelling tools. Different modelling approaches (multi agent systems, discrete events simulation, system dynamics, etc.) usually envisage the use of different modelling tools (software). This makes the issue of their mutual communication a central concern to the researcher. In Figure 1 development of simulation tools architecture is shown spreading from stand-alone simulators to Web solutions and homogeneous or heterogeneous distributed simulators.

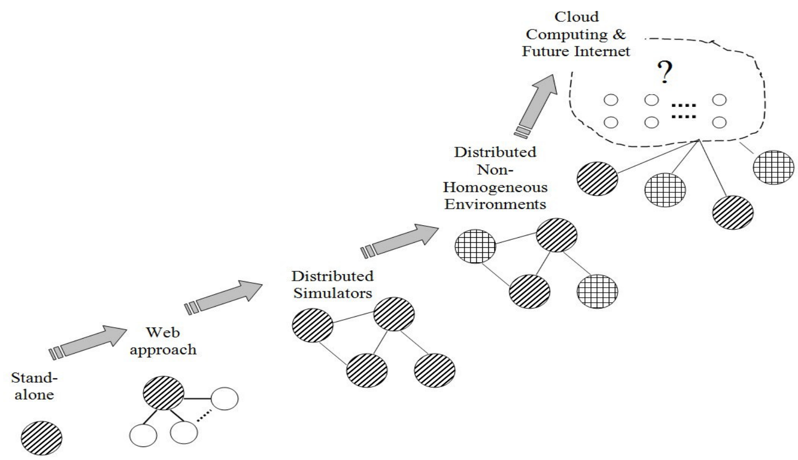


Figure 1: Development of simulation tools architecture

At present, it is very difficult to combine different models made by different simulators because there are no unified descriptions (specifications) for simulation models and a common and well suitable approach for joining simulation tools has not been developed. Therefore, designers cannot co-operate easily when using different simulator models.
Of course, there are several solutions, but they are complicated and not suitable for people without specific knowledge.
This is an important constraint that impedes the development of distributed simulation especially within social and behavioural sciences. These sciences often deal with complicated socio-technical systems that cannot be explored at the sufficient level of quality with one modelling tool.

High Level Architecture (HLA) (Carley 2002) is a concept of the architecture for distributed simulation systems. HLA ensures interoperability and reuses among simulations. It consists of rules that separate parts of distributed simulation model (federates) must follow to achieve proper interaction during a federation execution (see Figure 2); Object Model Template that defines the format for specifying the set of common objects used by a federation (federation object model), their attributes, and relationships among them; Interface Specification, which provides interface to the Run-Time Infrastructure, which can be distributed and ties together federates during model execution.

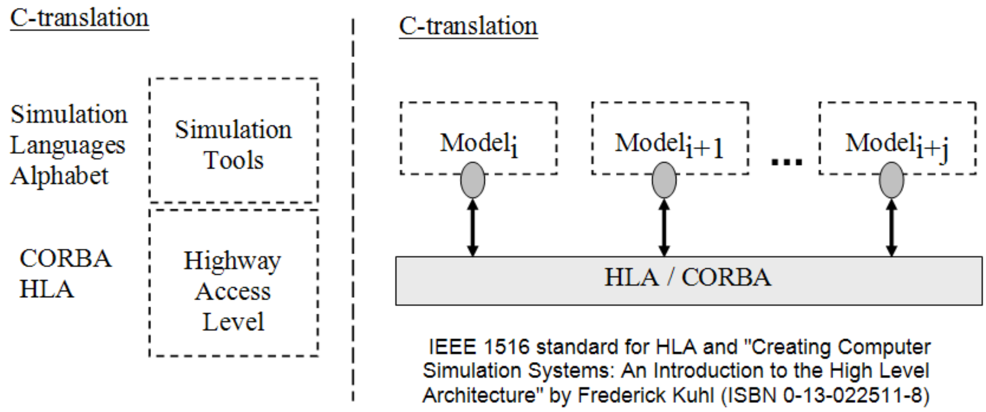


Figure 2: Communication in distributed simulation model

The distributed time management can be done, because all federates' nodes directly undertake synchronization roles. Therefore, the total simulation takes less time and the system is safer, unfortunately, implementation is more complex and laborious.
HLA was created for certain purposes and it was developed according to the needs of industry, where HLA was used. Accordingly innovations of HLA were encapsulated and less attention was paid to the developments within the field of simulations. HLA is a powerful tool for complex simulation systems. But as the technologies develop, the scope and functionality of simulations becomes wider. Inevitably some spheres emerge where HLA does not meet the needs anymore, it is not always convenient. Among one of shortcomings of HLA is its complexity. The wide variety of HLA functionality is rarely needed, besides expenses of HLA and its implementation are rather high (Aizstrauts et.al 2010). Some different protocols and methods (CORBA, HLA, FIPA, ALSP, DIS and other) (Strassburger 2006; Verbraeck 2004; Bolton 2001) exist for elaboration the communication environments, unfortunately any of it has their own disadvantages and right selection is still problematic.

One of the most important problems of the existing communication tools (environments) is compatibility with the architectural solutions of the Future Internet, because environments are non-flexible, and with low adaptability to the requirements of the SoA and the Future Internet of services.

4. THE FUTURE INTERNET AND THE CLOUD

Over the last decade, most Europeans and more than two billion people worldwide have become Internet users. More than 67.6% in the European Union and almost 26% people worldwide use the Internet. In 2010 the number of Internet users in China reached 400 million, which is more than the population of the entire United States (Blackman et.al 2010). With over 1 trillion pages and billions of users the Web is one of the most successful engineering phenomena ever created. At the end of 2009 there were 234 million websites of which 47 million were added in the year. The Web is now a rich media repository, the current upload to Flickr is equivalent to 30 billion new photos per year and YouTube now serves over 1 billion videos per day (Pedrinaci 2010).

The rapid development of mobile technology, providing Internet access to individual remote devices (Internet of Things), has accelerated the pace of development; as a result it is expected that the number of users in 2020 will reach seven billion. The Internet moved from the technical to the social category, where development is more driven by the efforts of interested Internet users than the pressure of technological achievements (Pedrinaci 2010). Some of these users see it as a business environment, others as a provider for social networking opportunities. An inevitable fusion of technology and the social environment can be observed, where the Future Internet is actually a representation of the public, a complex, but integral part of the social system. Future research must conceptualize the Internet as the global social machine. Research shows that (Blackman et.al 2010) technology development is not the key driver of Internet development. The vital part is social factors, which determine that no revolutionary technical changes are possible in the Internet development at the moment. That is why the implementation of new methodologies, algorithms and services gains a special status. They change and improve the Internet step by step by improving the quality of service a user receives i.e. performance, security and intelligence. Simulation is not an exception as it can become one of such services in the Future Internet environment.

Mobile technology has long been offering personal simulation tools such as AgentSheets and others. As a result, it is expected that simulation will become one of the services in the future network Internet of Services, which, in turn, will be promoted by the Internet of Things, which will provide access to remote and mobile equipment.

The Semantic Web is an extension of the current human – readable Web, adding formal knowledge representation so that intelligent software can reason with the information in an automatic and flexible way (Pedrinaci 2010). Semantic Web research has therefore largely focussed on defining languages and tools for representing knowledge in a way that can be shared, reused, combined, and processed over the Web. This research has led to a plethora of standards such as RDF(S), OWL, as well as corresponding tools such as ontology editors, RDF(S) storage and querying systems. The semantic approach can be one of the basic elements in the development of a unified approach for the specification of simulation models and the translation of further constructions to execute them in the Future Internet environment.

It is noted that the human-machine interface plays a special role in the development of the Internet where the emphasis on browser-type access and search-engines gradually moves to Facebook, Twitter and national social network analogues. However, this is just the beginning as 3D immersion and virtual and augmented reality (VR/AR) applications development and introduction (Future Internet 3D) is expected. The development of this direction could have a direct impact on the progress of simulation engineering by providing the visualization of simulation results, which could gradually replace built-in visualization tools, contributing to the unification of a simulation approach.

There are different types and practices of standardization: with ISO-styled standardization, the process may be heavy-handed; or IETF types – global and motivated by desire to keep the internet running effectively, a place where some consensus will be found, and based between ISO and IEEE; or IEEE types of standards – in some ways the opposite of ISO in process and being purely technical; finally we have various Web consortia (e.g. OASIS) becoming even more important since they are high level, including the various open source standards for interfaces and whole applications, which may or may not be normalized (Blackman et.al 2010).

Currently, the most popular methods to describe data and semantic information are considered XML, RDF or OWL, SOAP or REST notations are used for protocol description, but BPEL or BPMN are used to specify orchestration mechanisms. Efforts are being made to make these languages the standard tools for describing Future Internet services, although their possibilities to describe the functioning of a socio-technical system raise serious doubts. At least stochastic process specification could lead to the solving of a series of difficult problems. Opposed to the orchestration approach, the use of choreographies CHOReOS (http://www.choreos.eu) seems more actual and promising, because there is no single monitoring and synchronization service, but all members of the service network work independently and within the extent of their competence to reach the determined goal. Such architecture is more viable, as there are no central administration resources, the disruption of which equals the doom of the service.

To describe the essence of Future Internet, the following concept set is offered: Internet of Contents (IoC), which is provided by the Internet of Services (IoS), while the remote access to specific devices is provided by the Internet of Things (IoT), but it is possible that the user network is part of a larger network, and then it is considered the Internet of Networks (IoN). There is a need for both researchers and practitioners to develop platforms made up of adaptive Future Internet applications. In this sense, the emergence and consolidation of Service-Oriented Architectures (SOA), Cloud Computing and Wireless Sensor Networks (WSN) rise benefits, such as flexibility, scalability, security, interoperability, and adaptability, for building applications.

As one of important parts of the Internet of Services the Cloud computing could be mentioned. Cloud computing ensure scalable storage, computation facilities, application hosting, or even the provisioning of entire applications accessed remotely through the Internet. Some well-known commercial Cloud solutions are for instance Google's AppEngine, Gmail, and Docs, Amazon's Elastic Computing, or Salesforce (Pedrinaci 2010). The European Commission supports approximately 140 Future Internet projects to a greater or lesser extent and some research of them are related with Cloud (http://www.future-internet.eu/activities/fp7-projects.html), for instance, Cloud4SOA (http://www.cloud4soa.eu/) focuses on the semantic interoperability and on introducing a user-centric approach for applications which are built upon and deployed using Cloud resources; Cloud-TM (http://www.cloudtm.eu/) working on the development and administration of Cloud applications; CONTRAIL (http://contrail-project.eu/) designs an open source system for integration of heterogeneous resources into a single homogeneous Federated Cloud; CumuloNimbo (http://www.cumulonimbo.eu/) provides consistency, availability, and simpler programming abstractions, such as transactions; Morfeo 4CaaST (http://4caast.morfeo-project.org/) - platform for elastic hosting of Internet-scale multi-tier applications; mOSAIC (http://www.mosaic-cloud.eu/) develops an platform allowing to the users to tune Cloud services; OPTIMIS (http://www.optimis-project.eu) establishes an open Cloud Service Ecosystem for adaptable and reliable IT resources support; VISION Cloud (http://www.visioncloud.eu/) introduces an infrastructure for reliable delivery of data-intensive storage services and many other projects launched during previous calls.

Cloud-based services because are expected to become one of the main IT market niches and as a consequence many large companies are working on the creation of their own solution to retain a competitive position (The Economist 2009). Due to the reason mentioned above it would be rational to provide Cloud-based simulation service as integral component of the Internet of Services.

5. SIMULATION LAYOUT DESIGN AND VISUALIZATION OF THE RESULTS

One of the essential actualities is the display of simulation results in a form that is understandable to domain specialists and as close as possible to the specifics of the business sector.

Currently, built-in tools with limited functionality are used to display simulation results. Furthermore, it is not clear how to visualize the results of a non-homogenous and distributed simulation.

There are several different definitions for Virtual Reality (VR), but one of the formulations determines that it is the simulation of the goal system using computer graphics and providing the user with the ability to interact with the researchable system by using three and more levels of freedom (Burdea 2003). It all depends on the research object. If reality is the object of research, the virtual constructions complement the visible object. In the present case, there is talk of an Augmented Reality (AR) solution. If the research object is a virtual object, where a real life existing installation or component is added, then it can be considered an Augmented Virtuality (AV) situation. Currently, virtual reality questions relate to a scientific sub-sector, which combines various fields such as computers, robotics, graphics, engineering and cognition. VR worlds are 3D environments, created by computer graphics techniques, where one or more users are immersed totally or partially to interact with virtual elements. Mainly, special devices stimulate the sight, hearing and touch. Higher immersion level can be achieved with output devices, which are mainly directed to humans visual sense, for example, head mounted displays (HMD), stereoscopic monitors, special glasses, projection walls, CAVE systems, etc. Multiple sound sources positioned provides 3D sound, and touch can be simulated by the use of haptic devices (Moraos and Machado 2009).

The basic element of a VR system is the authoring platform which provides the import of models from other 3D graphics tools (AutoCAD, Maya, 3D Max etc.), the generating and rendering of scenes, and the building and operation of scenarios. Although the construction of VR/AR systems is still expensive and time consuming, which is caused by the incompatibility of hardware and authoring platforms, gradual development is taking place (Ginters et.al 2007). At least VRLM supports most authoring platforms. In recent years VR platform providers (Bluemel 2011) have been working on the creation of tools for simulation layout planning. Initially, VR/AR might be a good supplement to any of the simulation environments to improve the clearness of simulation, but in the near future the agreement between simulation and VR professionals could lead to the development of a unified VR-simulation interface concept. Another useful VR application could be the visualization of simulation results by adapting them to the perception and industry specifics of domains experts. In any case, it is clear that the fusion of VR with simulation environments and tools is a matter of the nearest future.

There are more than twenty EC FP7 projects, which currently do research in the field of virtual reality, however, there are few with a connection to the development of Future Internet, for instance, IRMOS (http://www.irmosproject.eu) will design, develop, integrate and validate a Service Oriented Infrastructure that enables a broad range of interactive real-time applications. It will support the development and deployment of real time applications in a distributed way. The infrastructure will be demonstrated by focusing on virtual and augmented reality; VirtualLife (http://www.ict-virtuallife.eu) aims to combine a high quality immersive 3D virtual experience with the trustworthiness of a secure communication infrastructure, focusing on the creation of secure and ruled places within the virtual world where important transactions can occur; 2020 3D Media

(www.20203dmedia.eu) is aimed to the development of new technologies to support the acquisition, coding, editing, networked distribution, and display of stereoscopic and immersive audiovisual media, capable of providing novel and more compelling forms of entertainment both for home and for public grounds. The users of the resulting technologies will be both media industry professionals across the current film, TV and 'new media' sectors producing programme material as well as the general public.
For the time being, virtual reality experts are busy with their internal problems and, it seems, are not ready to deploy and adapt their systems for the Future Internet environment. A close cooperation between Future Internet architects and VR/AR ideologists has not been formed because both sides are not ready for serious negotiations, although opinions can be heard, that Web 3.0 will be in 3D. However, it is clear that sooner or later it will happen, and simulation experts should participate in the development of this unified concept.

6. SIMULATION HIGHWAY – THE CONCEPT

Simulation Highway - common approach and rules to deploy, access, join and exploit the different and heterogeneous simulation models in distributed environment on the Future Internet and Cloud.

The Simulation Highway (Ginters and Vorslovs 2008; Ginters and Aquilar 2008) ensures translation and distribution the simulation requests in the Cloud. These simulation requests address a set of simulation cells organizing implementation highway during the simulation session of defined task.

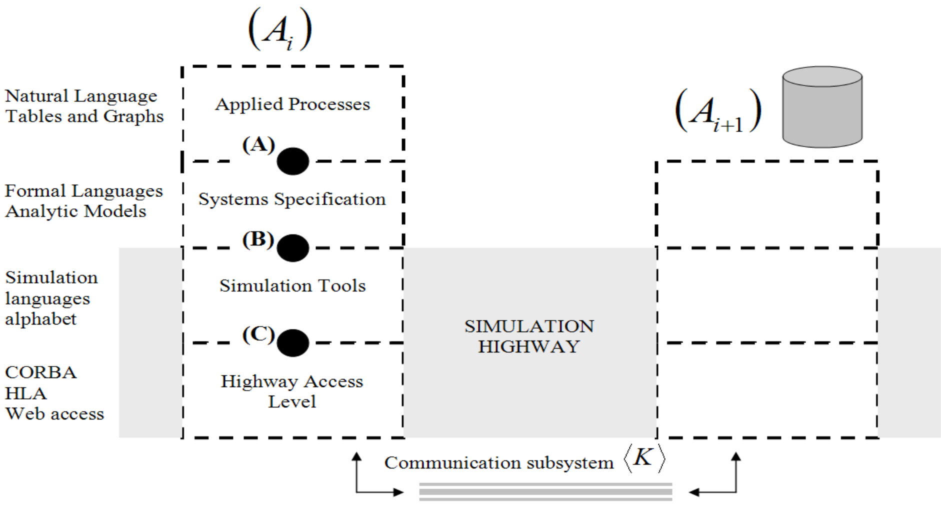


Figure 3: Multilevel Requests Translation and Distribution on Future Internet (Ginters and Aquilar 2008)

Each simulation cell serves as a server and simultaneously as a switch so that various and previously existing simulation models that are registered to the cell and participates in the decision making task in real-time can be connected to the simulation highway of the task (see Figure 4). Cognition is integral attribute of each simulation cell.

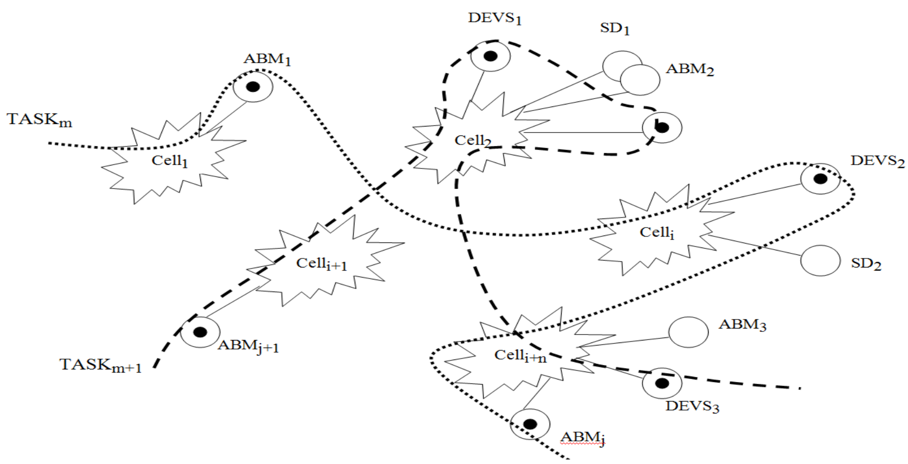


Figure 4: Simulation Highway – Distributed Multiple Access Simulation Environment

To each cell different models: discrete-event (DEVS), system dynamics (SD), agent-based (ABM) can be registered. Each task, formulated by the domain expert, generates a heterogeneous chain of interacting models, or highway. The results are visualized in a manner that is understandable and demonstrative to the client using VR facilities provided by the Future Internet 3D.

The important problem for distributed simulation is placidity of the Cloud platforms. For example, Amazon's EC2 supports the Message Passing Interface (MPI), the message-based communications protocol used by parallel programs that run on clusters (Fujimoto et.al 2010). This provides theoretically a variant to making parallel and distributed simulations.

Nevertheless this approach is too general to be successfully used in such a specific field as simulation, because Cloud platforms are better oriented at providing high bandwidth communications among applications for longer sessions than to interchange of many small messages requiring quick delivery. Theoretically it is possible that simulation societies would arrange common Cloud platform suitable for implementation of the simulation tasks, but such approach would be the step away from the aim, because the benefit is use of the common Cloud solutions by all the tasks. Therefore, selection of the right Cloud platform would be the challenging task.

7. ONTOLOGY BASED ADAPTABLE UNIVERSAL SIMULATION SPECIFICATION LANGUAGE

One of the major problems is a different process specification and the performance of further transmissions, ensuring cooperation with the Simulation Highway.

Existing simulation languages are different and problem-oriented; the abstraction level is low enough. On the other hand, the languages used by software engineers are not suitable to describe real and complex processes. There have been several attempts to achieve universal solutions (see Figure 5).

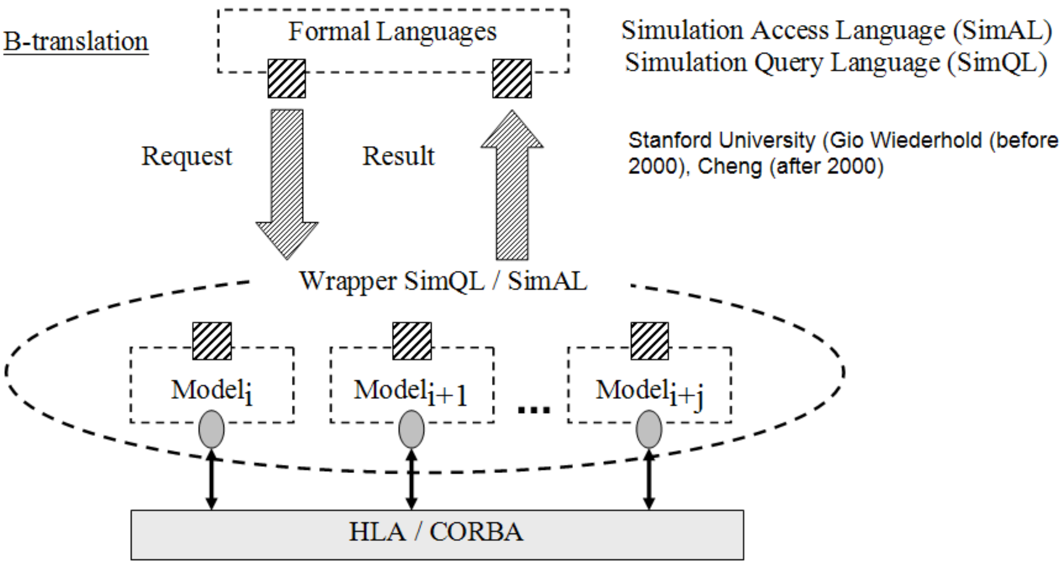


Figure 5: Client-Server Access Model SimAL/SimQL

In the beginning of the 21st century client-server access was offered for simulation models by integrating SQL query analogues, but the proposed solution did not gain wide acceptance, because it was not enough, i.e. access to the simulation environment did not improved for domain experts, because the query still had to be made in what a person without programming skills would consider an unfriendly language.

Ontologies provide formal methods for describing the concepts, categories, and relationships within a domain. Domain ontologies may be particularly helpful to simulation modellers since they can be used to communicate domain information to simulation and modelling tools with limited human intervention.

Ontology driven simulation (ODS) takes advantage of this feature by using software tools to align knowledge resident in domain ontologies with knowledge resident in a modelling ontology in order to facilitate the creation of simulation models. In ODS, a tool is used to map concepts from domain ontologies to concepts in a modelling ontology and then create instances of modelling ontology classes to represent a model.

Ontology driven simulation uses ontologies to drive the creation of simulation models and in doing so makes use of an agreed upon set of terms and relationships that are shared by domain experts, modelers, and model development tools. These terms and relationships provide a semantic grounding and structure for the executable model. The domain ontology exists for a specific application area, and its classes and instances determine the types of components that will make up the model.

The modeling ontology is developed independently of any specific domain ontology but may rely on general upper ontologies. Modellers receive the benefit of having a stable set of terms for the domain available through the design tools that they are using.

Domain experts and others who make use of the simulation models benefit by having simulations use a common set of terms with which they are familiar.

Under the framework of the Simulation Highway project is intended to create Ontology Based Adaptable Universal Simulation Specification Language based on a knowledgeable, ontological and semantic approach that would give various domain experts an immediate possibility to specify different simulation cases as well as translate, distribute and implement these specifications in the Cloud through the Simulation Highway.

The language would be based on UML and its followers BPMN and BPEL, and supplemented with the possibilities to describe stochastic and heterogeneous processes, as well as additions that would provide domain experts with more friendly access. The use of BPMN and BPEL would facilitate the introduction of the new simulation specification language in the Future Internet environment.

8. SIMULATION HIGHWAY AND VISUALIZATION OF THE RESULTS

It is clear that the presentation form of the simulation results should be demonstrative and close to the perceptual characteristics of domain experts. The higher is the level of immersion, the better the quality of the gained knowledge.

The important reason for VR/AR use is the requirements for the quality and the performance of simulation layout design. Of course, in industrial tasks it does not a critical factor like military applications (Smith 2010), but in any case intelligent application of VR facilities would substantially reduce the potential errors and time for the layout design.

The development of virtual and augmented reality (VR/AR) solutions and evolution of simulation environments creates a convergence of both sides where VR/AR will become an integral part of simulation tools. However, the next step is the demand for universalism and sufficiently open access, which could be provided by the Future Internet of Services and Cloud facilities.

This means, that requirements have to be defined and developed for an interface between the Simulation Highway and the set of virtualization tools on the Future Internet 3D (see Figure 6).

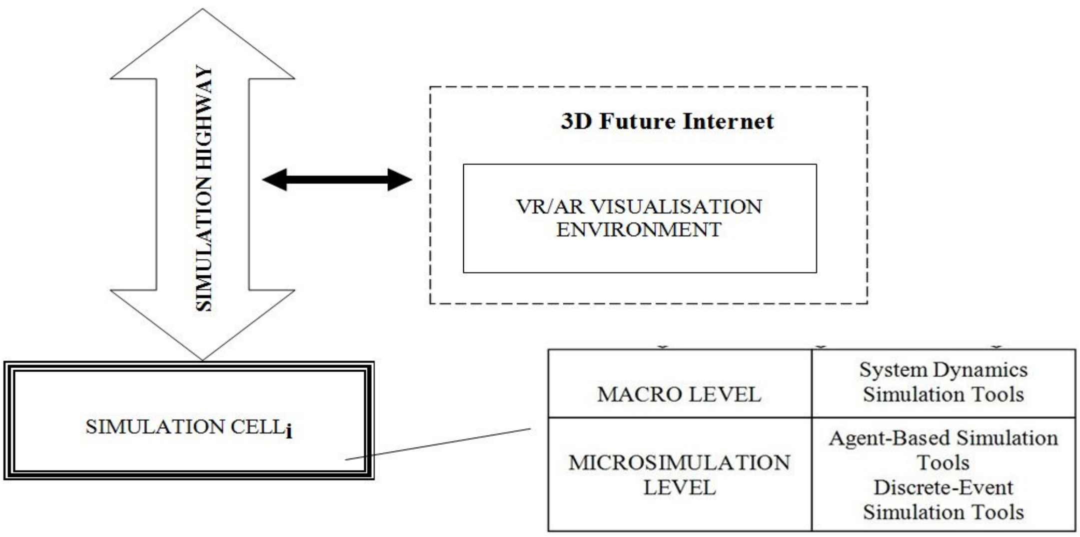


Figure 6: Simulation Highway - part of the Internet of Networks

The Future Internet of Devices (Things) ensuring access to the net for the mobile users dramatically increases the number of potential modellers. However, in the case of simulation, it must be remembered that the resource capacity of mobile devices (screen, memory, battery) is limited and cannot be compared to stationary users.

9. CONCLUSIONS

Simulation, in its half a century long existence, has achieved structurally diverse and heterogeneous solutions. New possibilities have emerged, which are analogous to software design, such as prototyping, automated model generation and others, and that is why simulation can be defined as simulation engineering.

The diversity and time consuming creation of models claim for requirements by unification and standardization. This can slow down the development of simulation technologies while expanding its scope of application, which could create the possibility of convenient decision making for domain experts.

Unification is doubtful without the creation of a unified approach to the distributed simulation and Ontology Based Adaptable Universal Simulation Specification Language based on a knowledgeable, ontological and semantic approach, which should preferably be based on habitual, well supported and with the Future Internet key persons understandings compatible constructions such as UML, BPEL, BPMN, etc.

The development of the Future Internet solutions is one of the priorities of European research. Therefore, service providers, including simulation professionals, should appropriately focus on solution development for the Future Internet of Services on the Cloud. A major advantage of simulation in the Cloud is that it is scalable. As the number of modellers increases, simulation servers can be added to increase the computational and storage capacity. Serious challenge is selection of the right platform for the Cloud due to real-time requirements of simulation.

The visualization of simulation results has to be done in an understandable and acceptable form, which VR/AR solutions can provide. These have to be integrated into the Future Internet 3D, but the Simulation Highway, as a part of Internet of Networks, has to ensure an interface between VR/AR and simulation models.

The proposed Simulation Highway solution is an attempt to unify the access to heterogeneous simulation models, to provide domain experts with the immediate specification possibilities and to ensure the economy of computer resources by deploying the simulation on the Future Internet of Services. The solution promotes the development tendencies of the Internet of Things, because it intends to deploy simulation tools to mobile communication devices.

Project partners will realize a demonstration tasks which will relate to a decision making simulation on port logistics and policy modeling. Expected results would allow for the consolidation of varied and different previously developed simulation models into a unified environment and ensure their direct usage by miscellaneous industry domain experts and specialists.

ACKNOWLEDGMENTS

The current article was prepared in the framework of ERAF project No.2DP/2.1.1.2.0/10/APIA/VIAA/001 “Support for preparation of IST FP7 STREP project “Simulation Highway””.

REFERENCES

Von Bertalanffy, Ludwig, 1976. General System Theory: Foundations, Development, Applications, Rev. ed., ISBN-978-0807604533.

Banks, J., Carson, S. II, and Nelson, B.L., 1996. Discrete-Event System Simulation, 2nd ed., Prentice Hall, Upper Saddle River, N.J.

Chang, Chris, 2004. Simulation modeling handbook: a practical approach, CRC Press LLC, ISBN 0-8493-1241-8.

Pidd, Michael, 1996. Tools for Thinking: Modelling in Management Science, ISBN 978-0471964551, 1996, 360p.

Banks, Jerry, 1998. Handbook of simulation, John Wiley & Sons, ISBN 0-471-13403-1.

Gilbert, Nigel, and Troitzsch, Klaus G., 2006. Simulation for the Social Scientist. Second Edition. ISBN 139780335221493, Open University Press.

Bruzzone, A., Verbraeck, A., Ginters, E., et. al., 2002. Logistics Information Systems, Part 1-2, Jumi, ISBN 9984-30-021-8, Riga, 2002, 682 p.

Carley, K.M., 2002. Computational organizational science and organizational engineering, Simulation Modelling Practice and Theory 10 (2002) 253–269.

Aizstrauts, A., Ginters, E., Aizstrauta, D., 2010. Step by Step to Easy Communication Environment for Distributed Simulation. //In: *Annual Proceedings of Vidzeme University College „ICTE in Regional Development”*, ISBN ISBN 978-9984-633-20-6, Valmiera: Vidzeme University of Applied Sciences, Sociotechnical systems engineering institute, 2010, pp.11-13.

Silins, A., Ginters, E., Aizstrauta, D., 2010. Easy Communication Environment for Distributed Simulation. //In: *World Scientific Proceedings Series on Computer Engineering and Information Science 3 “Computational Intelligence in Business and Economics”*/ Proceedings of the MS’10 International Conference, Barcelona, Spain, July 15-17, 2010, ISBN 978-981-4324-43-4, pp.91-98

Strassburger, S., 2006. The road to COTS-interoperability: from generic HLA-interfaces towards plug-and-play capabilities, in *Proceedings of the 37th conference on Winter simulation*, December 03-06, 2006, Monterey, California.

Verbraeck, A., 2004. Component-based Distributed Simulations. The Way Forward?, in *Proceedings 18th Workshop on Parallel and Distributed Computer Simulation*. (Kufstein, Austria, 16-19

May 2004), IEEE Computer Society Press, Los Alamitos, CA, 2004. pp. 141-148.

Bolton, F., 2001. *Pure CORBA*, Sams, ISBN 978-0672318122, 2001, P. 944.

Blackman, C., Brown, I., Cave, J., Forge, S., Guevara, K., Srivastava, L., Tsuchiya, M., Popper, R., 2010. Towards a Future Internet. Interrelation between Technological, Social and Economic Trends. Final Report for DG Information Society and Media, European Commission DG INFSO, Project SMART, 2008/0049 November 2010

Pedrinaci, C., and Domingue, J., 2010. Toward the Next Wave of Services: Linked Services for the Web of Data, Journal of Universal Computer Science.

The Economist. 2009. Clash of the clouds. The Economist. 392, 80-82, 2009

Burdea, G., Coiffet, P., 2003. Virtual Reality Technology. 2nd ed., Wiley Interscience, 2003.

Marcos de Morales, R., Dos Santos Machado, L., 2009. Online Training Evaluation in Virtual Reality Simulators Using Possibilistic Networks.//In: Proceedings SHEWC 2009 Safety, Health and Environmental World Congress, July 26 - 29, 2009, Mongaguá, Brazil.

Ginters, E., Vorslovs, I., 2008. Simulation Highway for Food Quality Management. *Proceedings of the International Scientific Conference „Aplied Information and Communication Technologies"*, Latvia University of Agriculture, Jelgava, Latvia, April 10-12, ISBN 978-9984-784-68-7, 2008, pp.86-95.

Ginters, E., Aguilar Chinea, M. R., 2008. Simulation Highway for Applied Systems Management. *Proceedings of the 7th WSEAS Int. Conf. on CIRCUITS, SYSTEMS, ELECTRONICS, CONTROL & SIGNAL PROCESSING (CSECS '08)*, Puerto de La Cruz, Tenerife, Spain, December 15-17, 2008, ISBN 978-960-474-035-2, 2008, pp.180-186

Towards a Future Internet, 2010. Interrelation between Technological, Social and Economic Trends. *Final Report for DG Information Society and Media. European Commission DG INFSO Project SMART 2008/0049*, November 2010, Available from: http://www.internetfutures.eu/wp-content/uploads/2010/11/TAFI-Final-Report.pdf, [accessed 01.04.2011]

Fujimoto, R.M, Malik, A.W., Park, A.J., 2010. Parallel and Distributed Simulation in the Cloud, SCS M&S Magazine – 2010 / n3

Smith, R., 2010. Simulation in the Cloud, ITEC 2010, London, 18-20 May.

AUTHORS BIOGRAPHY

Egils GINTERS is director of Socio-technical Systems Engineering Institute. He is full time Professor of Information Technologies in the Systems Modelling Department at the Vidzeme University of Applied Sciences. He is a member of the Institute of Electrical and Electronics Engineers (IEEE), European Social Simulation Association (ESSA) and Latvian Simulation Society. He participated and/or coordinated some of EC funded research and academic projects: FP7 FUPOL project No. 287119 (2011-2014), FP7-ICT-2009-5 CHOREOS project No. 257178 (2010-2013), e-LOGMAR-M No.511285 (2004-2006), SocSimNet LV/B/F/PP-172.000 (2004-2006), LOGIS MOBILE LV/B/F/PP-172.001 (2004-2006), IST BALTPORTS-IT (2000-2003), LOGIS LV-PP-138.003 (2000-2002), European INCO Copernicus DAMAC-HPPL976012 (1998-2000), INCO Copernicus Project AMCAI 0312 (1994-1997). His main field of interests involves: systems simulation, logistics information systems and technology acceptance and sustainability assessment.

Yuri MERKURYEV is Professor and Head of the Department of Modelling and Simulation at Riga Technical University in Riga, Latvia. His professional interests include methodology of discrete-event simulation, supply chain simulation and management, as well as education in the areas of simulation and logistics management. Prof. Merkuryev is a corresponding member of the Latvian Academy of Sciences, President of Latvian Simulation Society, Board Member of the Federation of European Simulation Societies (EUROSIM), SCS Senior Member and Director of the Latvian Center of the McLeod Institute of Simulation Sciences, and Chartered IT Professional Fellow of the British Computer Society. He authors about 300 scientific publications, including 6 books, and is a co-editor (with Galina Merkuryeva, Miquel Angel Piera and Antoni Guasch) of a recently published by Springer-Verlag book "Simulation-Based Case Studies in Logistics: Education and Applied Research". He is editorial board member of several journals, including "Simulation: Transactions of the Society for Modeling and Simulation International," and "International Journal of Simulation and Process Modelling". Prof. Merkuryev regularly participates in organising international conferences in the area of modelling and simulation. In particular, he has served as General Chair of the International Conference "European Conference on Modelling and Simulation", ECMS'2005. He is permanently involved into organising of the HMS (The International Conference on Harbour, Maritime & Multimodal Logistics Modelling and Simulation) series of conferences within the annual International Mediterranean and Latin American Modelling Multiconference, I3M.

Rosa Maria AQUILAR CHINEA is Professor and Vice-Rector at Universidad de la Laguna in Santa Cruz de Tenerife, Spain. His professional interests include ontologies of discrete-event simulation environments. Prof. Aquilar Chinea regularly participates in organising international conferences in the area of modelling and simulation. In particular, he has served as Chair of the International Mediterranean Modelling Multiconferences EMSS 2006 and other.

7. Aizstrauts, A., Ginters, E. & Aizstrauta, D. (2011). Easy Communication Approach for Data Exchange in Distributed Simulation Environment. In *Proceedings of the 13th WSEAS international conference on Automatic control, modelling & simulation (ACMOS'11)* (pp. 34-38). Stevens Point: World Scientific and Engineering Academy and Society (WSEAS). Retrieved from http://www.wseas.us/e-library/conferences/2011/Lanzarote/ACMOS/ACMOS-04.pdf

Easy Communication Approach for Data Exchange in Distributed Simulation Environment

ARTIS AIZSTRAUTS, EGILS GINTERS, DACE AIZSTRAUTA
Sociotechnical Systems Engineering Institute
Vidzeme University of Applied Sciences
Cesu Street 4, Valmiera LV-4200
LATVIA
{artis.silins, egils.ginters}@va.lv http://www.socsimnet.com

Abstract: - The development of simulation enables the researchers to explore more sophisticated problems at a more detailed level. At the same time this implies a necessity to use several models, or even different modelling tools. Different modelling approaches (multi agent systems, discrete events simulation, system dynamics, etc.) usually envisage the use of different modelling tools (software). This makes the issue of their mutual communication a central concern to the researcher. There are several solutions, but they are complicated and not suitable for people without specific knowledge. The aim of the article is introduction to new and easy communication environment which would be used to substitute HLA and other more expensive tools.

Key-Words: - Simulation, HLA, Distributed simulation, Easy communication

1 Introduction

At the level of decision making the consolidation of all data is necessary. Therefore, the demand for the communication environment which ensures the information exchange with appropriate efficiency is important. Many solutions exist, but HLA [10] can be nominated as more popular one. However, the high costs of the commercial versions promote for new ways searching and using of the distributed environments provided for software modules communication.

Meanwhile the variety of simulation tools, including the development of different user-friendly simulation tools, enables professionals of various fields to use them in their everyday routine. Therefore mechanisms for simulation models communication should be useful for these professionals, they should be unsophisticated and user-friendly, that do not require any specific knowledge or skills.

Hence this paper describes a solution that would give the opportunity to ensure the communication between modelling tools or distributed models without HLA communication architecture at the same time being unsophisticated and appropriate for users without extensive technical skills. It is primarily based on the “Easy communication environment for distributed simulation” [1]. Instead of HLA this paper concentrates on a newly developed inter-adapter communication mechanism.

2 How to Access the High Level Architecture (HLA)

High Level Architecture (HLA) [11] is a concept of the architecture for distributed simulation systems. HLA ensures interoperability and reuses among simulations. It consists of rules that separate parts of distributed simulation model (federates) must follow to achieve proper interaction during a federation execution.

Specification, which provides interface to the Run-Time Infrastructure, which can be distributed and ties together federates during model execution. The distributed time management can be done, because all federates’ nodes directly undertake synchronization roles. Therefore, the total simulation is takes less time and the system is safer, unfortunately, implementation is very expensive, complex and laborious.

The wide variety of HLA functionality is rarely needed and only applicable to specific simulations, therefore one might ask, do we really need to use HLA? After investigating different possible options for data exchange among different models and simulation tools, the authors elaborated a pilot-solution – easy communication environment that facilitates access to the HLA environment [1]. Communication was possible locally (within one computer) as well as globally (among several computers). For local communication this solution used communication adapter – independent software

– that was on the same computer as the simulation tool. In case of global communication, the solution used HLA architecture as well as the adapter(s).
For this solution to become real, the simulation tools need to have a feature to create extensions. These extensions provide data exchange between the model and the "outside". As one cannot be sure, that the model is able to ensure live instance communication channel, therefore data storage for data storing was created. Simulation tool with the help of extension received data from the data storage. Also the transfer of data was ensured by the extension – the data were transferred to communication adapter, and later on were transferred to other communication adapter with the help of HLA. It should be noted that this mechanism did not use time synchronization that is realised by HLA to ensure that in whole federation there would be one time and separate federer would not hasten. Basically HLA was used as a communication adapter switch that allows them to communicate with each other as well as it provided a list with possible interactions. Considering the abovementioned, there is no need to "force" the usage of HLA as it is rather powerful and capable communication architecture, nonetheless it is not worth wile to use it for comparatively easy tasks.

3 New Approach to Easy Communication

3.1 Fundamentals
The basic principles of the proposed mechanism is similar to one described in "Easy communication environment for distributed simulation pub."[1], only instead of HLA another communication mechanism is used for data transfer to other communication adapters. Changes have been made also to the structure of data that is being transmitted between simulation tool and communication adapter, as well as among several communication adapters. The main elements of this mechanism are the following: *Models (Simulation tools)*, *Communication library, Communication adapter, Data storage* and *Communication gateway.*

In order to be able to function in a communicative system, *Simulation tools* have to be able to communicate with itself or different simulation tool. In this article strictly focus only one of these versions, namely, where simulation tools can perform communication with other software (adapter). In most cases this function is performed with some additional plug-ins, extensions or libraries. For example, NetLogo, that is a multi agent-modelling tool, uses different extensions [12], as well as discrete event modelling tool Extend that also allows using extensions [2].

Communication library is an extension of simulation tool that enables to make connections with the adapter for sending and receiving data. Extensions can be made in different programming languages that depend on the features of simulation tool. For example, NetLogo extension was made using Java, but the extension for Extend is a DLL file, that was made in C programming language. The exchange of information between *communication library* and *communication adapter* is organised as simple as possible, as one can not predict in what programming language the extensions have to be made and what are the available features.

Communication Adapter is the element that communicates with models and *communication gateway,* and transmits them further to other communication adapters. Communication is carried out using a pre-defined data format, i.e. XML structured format, so that the data can be easily constructed by communication library. With communication adapter the user can define what information it will share with other members of communication network. Defining this information is a substantial step in communications process, as other users can observe what kind of information is available in within the network. When the user has defined this information to be shared, then communication adapters of other users will display a notification about newly available information that can be used in their models. For example, the user defines that he will share the data from the crop-harvesting model, and other users see in their communication adapters, that there is a possibility to receive data from this model.

Data storage is an element of communication adapter that keeps incoming data, transmitted through adapter. When the model is ready to use these data, then it demands them from communication adapter, this data that is being kept in data storage. *Data storage* can store also older data, if the model requires.

Communication gateway is the component that allows exchanging information between several computers. Without communication gateway information exchange is possible only within one computer. For the communication to be successful, the *communication gateway* has to be within a network infrastructure, so that communication adapters would be able to connect wit them. Communication adapters themselves do not have to necessarily be accessible from outside, because communication is performed through the gateway, not directly. *Communication gateway* maintains descriptive information about data available among models (not the data from models), it is registered in the communication gateway by the communication adapter. This adapter regularly gets into contact with the gateway and receives the up-to-date information about data available among models.

3.2 One Node Communication

Figure 1 shows a diagram of *communication node* that ensures communication at local level – within one computer. *Communication node* consists of *communication adapter* and models generated by simulation tool.

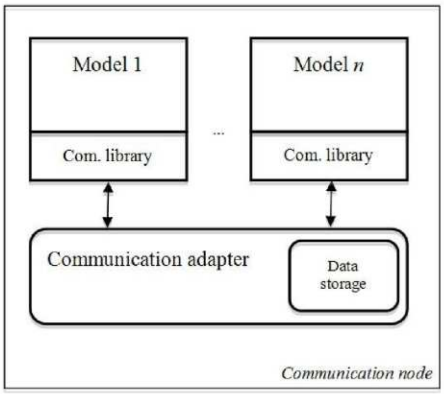


Fig. 1. Communication node

Between communication library and adapter within communication node two way communications is possible – receiving and transmission of data. Communication uses XML format, and that indicates also, which way of communication that is. If the model demands data, then the demanded data is returned, but this returned data is not XML anymore. Therefore it is easier for communication library to process the received data.

3.3 Multi Node Communications

Figure 2 shows the architecture for communication among several communication adapters in the framework of distributed simulation model, where nodes that can be located anywhere geographically. It is important that communication gateway can be seen by all adapters.

Also for communication between adapter and the gateway one uses XML messages. *Communication gateway* understands these kinds of action ("a") tags:

- Request for the list of available kinds of information (simulation data types, for example power of wind, direction of wind, etc);
- Registering of kinds of information;
- Subscription for simulation data;
- Sending of data.

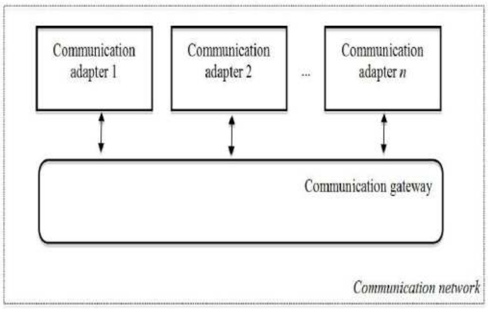


Fig. 2. Distributed communication network

Figure 3 shows the XML message for requesting the list of available kinds of information from simulations run within models. Adapter sends this kind of request frequently to *communication gateway*, so that the user can follow the up-to date information about available simulation results.

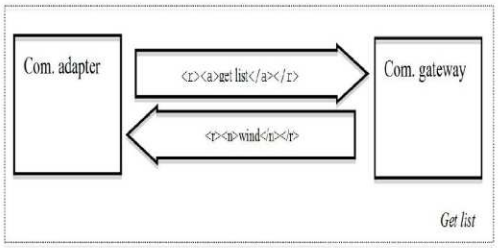


Fig. 3. Get list communication example

Figure 4 shows, how this information about available simulation information is being registered. Message has to indicate the type of action ("register"), as well as kind of information from the simulation ("wind"). Communication adapter after the registration receives a confirmation message, where "s" tag indicates status. If the status is "ok", then the registration was successful, if it shows "error", then no registration was performed. In this case "m" tag contains description of this error.

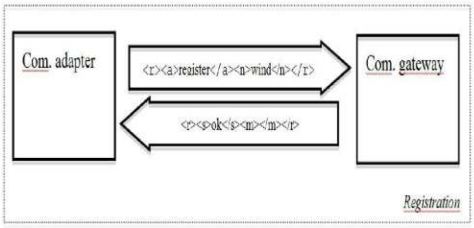


Fig. 4. Registration example

Figure 5 shows, how one subscribes for the simulation data. The example shows subscription for information about "wind". *Communication gateway* receives a confirmation message, where "s" tag indicates status. If the status is "ok", then the registration was successful, if it shows "error", then no registration was performed. In this case "m" tag contains description of this error.

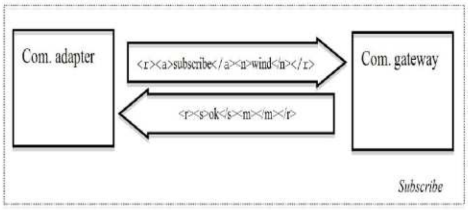


Fig. 5. Subscribe example

Figure 6 shows a situation, where communication adapter sends a message with type of action ("send"), kind of information ("wind") and value ("5"). When *communication gateway* receives the message with this information, it returns the status message to the adapter, and transmits the received information to all adapters that have subscribed for the certain simulation data ("wind" in this example).

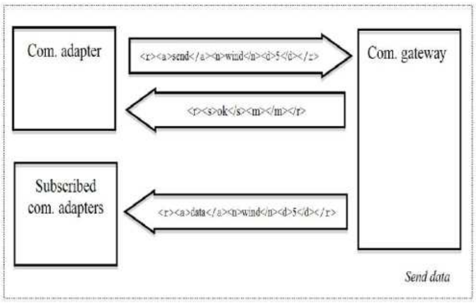


Fig. 6. Send/receive data example

Figure 7 illustrates the communication among adapters and activities of each adapter. Communication here is viewed from the side of adapters, without showing communication within the gateways.

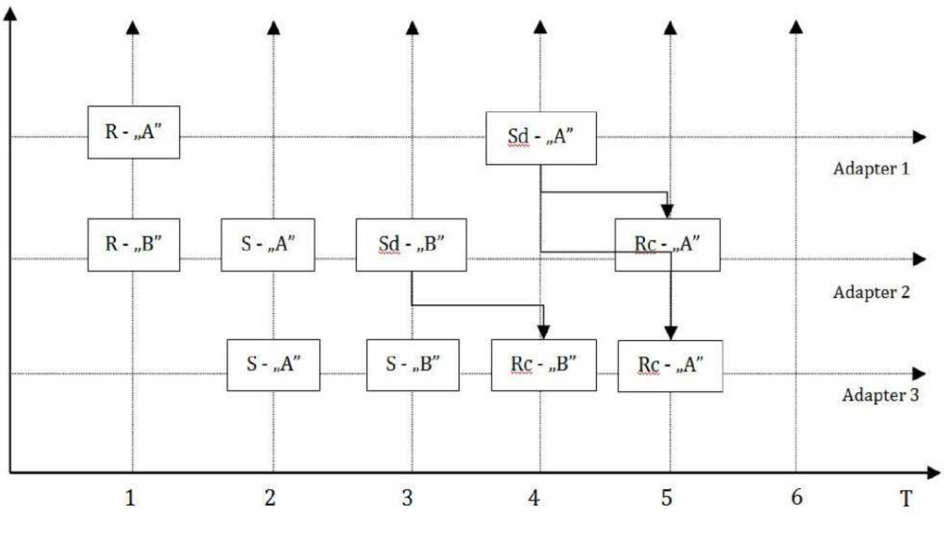


Fig. 7. Communications time diagram

The figure (see Fig. 7) shows communications system with three adapters. Adapter 1 performs only two activities in this example - registers interaction “A” and sends its data. Adapter 2 registers interaction “B” and subscribes for interaction “A” data. In the period of time 3 Adapter 2 sends data of interaction “B” and it is received by Adapter 3. Adapter 2 has subscribed for interaction “A”, and during period of time 5 it receives the data from Adapter1. During the period of time 2 Adapter 3 subscribes for interaction “A” and period of time 3 for interaction “B”. During the period of time 4 it receives data of interaction “B” that is being sent by Adapter 2, but during period of time 5 it receives data of interaction “A” that is sent by Adapter1.

4 Conclusion

Finally some important conclusions can be drawn regarding the distributed simulations and communication between models. First, it can be observed that the developers of simulation tools do not pay attention to the opportunities of models to communicate with other models, even of the same modelling environment. The paper outlines the structure of a comprehensible communication system and points out to its advantages. Second, the existent mechanisms do not offer time synchronization option; therefore there is a need to invent data storage. Further research should be aimed at searching for such time synchronising options. Third, future research should also seek for possibilities to ensure the transmission of more complicated data types. At the same time it should be noted that not all simulation tools offer the creation of extensions, therefore the comprehensible communication system described in this paper would not be applicable for such tools. Therefore if it is observed that simulation is revolutionizing the social and organizational sciences [11], then there are new challenges for engineering scientists in terms of enabling this process and empowering the researchers from other disciplines.

5 Acknowledgements

The current article was prepared in the framework of ERAF project No.2DP/2.1.1.2.0/10/APIA/VIAA /001 “Support for preparation of IST FP7 STREP project “Simulation Highway””.

References:

[1] A. Silins, E. Ginters, D. Aizstrauta, Easy communication environment for distributed simulation, Computational intellegence in business and economics, World scientific publishing Co. Pte. Ltd, 2010, 91-98.

[2] D. Krahl, The EXTEND Simulation Environment, in Proceedings of the 2002 Winter Simulation Conference/ E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, eds.,(2002).

[3] E. Ginters, A. Silins, J. Andrusaitis, Communication in distributed simulation environment, In Proceedings of 6th WSEAS International Conference on Systems Science and Simulation in Engineering, Venice, Italy, (2007), 217-221.

[4] E. Ginters, A. Silins, Multi-level Approach for Environmental Systems Modelling in the Ligatne Natural Trails. WSEAS Transactions on Systems, Issue 4, Volume 6, (2007), 795-801.

[5] E. Ginters, A. Silins, Simulation Data Exchange in Distributed E-learning Environment, Proceedings of the 4th WSEAS/IASME International Conference on EDUCATIONAL TECHNOLOGIES (EDUTE'08), Corfu, Greece, (2008),138-143.

[6] H. Hibino, Y. Fukuda , A user support system for manufacturing system design using distributed simulation, Production Planning & Control, Vol. 17, No. 2,(2006), 128–142.

[7] H. Hibino, Y. Fukuda, Y. Yura, K. Mitsuyuki, K. Kaneda, Manufacturing adapter of distributed simulation systems using HLA, Simulation Conference, Proceedings of the Winter conf.,vol 2,(2002) 099-1107.

[8] HLA Blockset™, ForwardSim Inc, High Tech Park of Québec, http://www.forwardsim.com/en/prod_hlab.php, (2009).

[9] K. Pan, S.J. Turner, W. Cai, Z. Li, A Hybrid HLA Time Management Algorithm Based on Both Conditional and Unconditional Information, Parallel & Distrib. Comput. Center, Nanyang Technol. Univ., Singapore, (2008), 203-211.

[10] K. Perumalla, Tutorial. Handling Time Management Under the High Level Architecture, http://www.ornl.gov/~2ip/doc/perumalla-tutorialslides-iitsec06.pdf, (2009).

[11] K.M. Carley, Computational organizational science and organizational engineering, Simulation Modelling Practice and Theory 10 (2002) 253–269.

[12] U. Wilensky, NetLogo, Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL, http://ccl.northwestern.edu/netlogo/, (2009).

8. Aizstrauts, A., Ginters, E., Aizstrauta, D., & Sonntagbauer, P. (2012). Easy Communication Environment on the Cloud as Distributed Simulation Infrastructure. In *Proceedings of the 5th WSEAS congress on Applied Computing conference, and Proceedings of the 1st international conference on Biologically Inspired Computation (BICA'12)* (pp. 173-178). Stevens Point, Wisconsin: World Scientific and Engineering Academy and Society (WSEAS). Retrieved from http://www.wseas.us/e-library/conferences/2012/Algarve/BICA/BICA-29.pdf

Easy Communication Environment on the Cloud as Distributed Simulation Infrastructure

ARTIS AIZSTRAUTS, EGILS GINTERS, DACE AIZSTRAUTA
Sociotechnical Systems Engineering Institute
Vidzeme University of Applied Sciences
Cesu Street 4, Valmiera LV-4200
artis.aizstrauts@va.lv, egils.ginters@va.lv
LATVIA

PETER SONNTAGBAUER
Cellent AG
Ziegelofengasse 33
1050 Vienna, Austria
Peter.Sonntagbauer@cellent.at
AUSTRIA

Abstract: - The development of Future Internet and Cloud computing creates new challenges for simulation scientists in terms of enhancing the accessibility to distributed simulation and empowering the researchers from other disciplines. The article tests the operation of simulation communication environment using Cloud infrastructure. The results of the experiment within Amazon EC2 Cloud infrastructure show no significant benefits for Easy Communication Environment users.

Key-Words: - HLA, CORBA, DIS, Cloud, Easy Communication Environment (ECE)

1 Introduction

Simulation nowadays plays an increasing role in situation analysis in research and the assessment of possible solutions in business. Sociotechnical systems deal with technological control as well as the impact of social or natural factors. EC FP7 project No.287119 FUPOL „Future Policy Modelling" (http://www.fupol.eu) aims at a completely new approach to traditional politics. Major innovations like multichannel social computing and crowd sourcing will change the way politicians communicate with citizens and enterprises and take decisions. This will enable governments to gain a better understanding of the needs of citizens. Likewise the FUPOL simulator will have the capabilities to simulate the effects of policies and laws and to assist governments in the whole policy design process. The simulation models and tools of the sociotechnical systems are often incompatible, heterogeneous and territorially distributed and it is especially problematic if domain specialists lack the expertise and specific programming skills to use them effectively. To achieve this, several heterogeneous and territorially distributed models have to communicate and co-operate. The development of Future Internet and Cloud creates new possibilities and challenges for simulation engineering by offering extended access.

This article discusses distributed simulation communication environments as the possible solution aiming at deployment of the simulation services on the Future Internet and the Cloud. Easy communication environment usage is tested within cloud computing infrastructure.

2 Distributed communication environments

2.1 High Level Architecture (HLA)

High Level Architecture (HLA) is a concept of the architecture for distributed simulation models. HLA is a powerful, however complex tool, and ensures interoperability and reuses among simulations. It consists of a)rules that determine federation; b)Object Model Template that defines the format for specifying the set of common objects used by a federation (federation object model); c)their attributes, and relationships among them; d)interface specification, which provides interface to the Run-Time Infrastructure, that can be distributed and ties together federates during model execution [16]. HLA is well standardized (IEEE Std 1516-2000, IEEE Std 1516.1-2000 and IEEE Std 1516.2-2000) [17] and could be used for heterogeneous and distributed simulation systems development.

2.2 Distributed Interactive Simulation (DIS)

Distributed Interactive Simulation (DIS) is based on distributed interactive simulation protocol – SIMNET. DIS is an open network protocol standard for linking real-time platform-level war gaming simulations. DIS standard specifies the layout of data to be transferred in network level. Communication packets/data PDUs are laid out exactly as defined in the DIS specifications. DIS Protocol Data Units (PDUs) define syntax (data format) and semantics (rules), for network data exchange and simulation interoperability. Designed to be an environment for real-time data exchange among distributed, computationally autonomous units. DIS has no central control unit for the entire simulation exercise and therefore may be used respecting choreographic interoperability provided in some Future Internet platforms. The development of DIS decreased following the proposal and promulgation of its successor – HLA [2].

2.3 Common Object Request Broker Architecture (CORBA)

The first Common Object Request Broker Architecture (CORBA) includes CORBA Object model, Interface Definition Language (IDL), and the core set of application programming interfaces (APIs) for dynamic request management and invocation (DII) and Interface Repository as well. Also some several features like interfaces for the Basic Object Adapter, memory management and more IDL language mappings exist [3]. Two kinds of the objects exist in CORBA: Servants and Clients. The Servant is an implementation entity that defines the operations supported by CORBA IDL interface. Client invokes operation on an object implementation from Servant [1].

CORBA standard was elaborated for joint communication of the programs written in different languages. To achieve this aim the interfaces of the modules must be specified in conformity with the requirements of the IDL. IDL specifications are accessible for C, C++, Java, COBOL, Smalltalk, language, which made this mechanism well suited enough. Nevertheless, the CORBA environment asks for centralized synchronization that made distributed simulation systems vulnerable, but it is important only in some specific cases [3].

2.4 Web Service

Web Service (WS) is a communications mechanism that allows the client to address server with a request. Web Service communication interface is defined as Web Services Description Language (WSDL), but communication occurs using Simple Object Access Protocol (SOAP), that is meant for structured information exchange. There are two types of WS use: Remote procedure calls and Service-oriented architecture. The drawback of WS is comparably slow operation and relatively greater amount of data that is transmitted. Web Services differ from the distributed object technologies in that they have reverted to an earlier “remote service” model. There is no concept of an object reference; instead a service is defined simply by an end-point that supports various operations [4].

2.5 Some drawbacks of existing communication environments

However to create heterogeneous and distributed simulation models researchers mostly use one of abovementioned communication environments. One of the most influential drawbacks is the lack of ready-made solutions that would allow creating multi-model systems. Unfortunately each case has to be dealt separately, by designing new communication environment among models. Integration of communication environments is not trivial, it is complex and time-consuming, and therefore a software engineer is needed as well as additional funding for the research project. In fact, the accessibility of modelling tools is very limited for domain experts, and this very often leads to mistakes in decision process.

3 Easy Communication Environment – way to the inclusion

To deal with the abovementioned problems concerning heterogeneous simulation modelling systems, a new communication concept was made – “Easy communication environment“. The essence of this concept lies in its simplicity – the domain expert is able to “connect” modelling tools without the help of software engineers. Easy Communication Environment has to be reusable and universal at the same time in order to be able to connect as much various modelling tools as possible. For more information see Aizstrauts A. et al [5].

The main principles were taken from HLA – communication occurs through broadcasting. The user uses special plugin/extension of simulation tool that allows send/receive data, indicates what data he wants to send/receive. The communication adapter operates on the same computer and it can either register the specified data or sign up for already registered data. So, similarly to HLA, the data is only sent to those, who have signed up for it.

Communication node consists of Communication adapter with embedded data storage, Communication library and simulation Models. The data storage allows storing the incoming data. It is necessary because models operate differently in time, and it may be possible that one model is not able to receive the amount of data sent from another model within a certain period of time. Easy Communication Environment uses XML data format to transfer simulation data requests and responses.

Figure 1 shows the architecture for communication among several Communication adapters in the framework of distributed simulation model, where nodes can be located anywhere geographically.

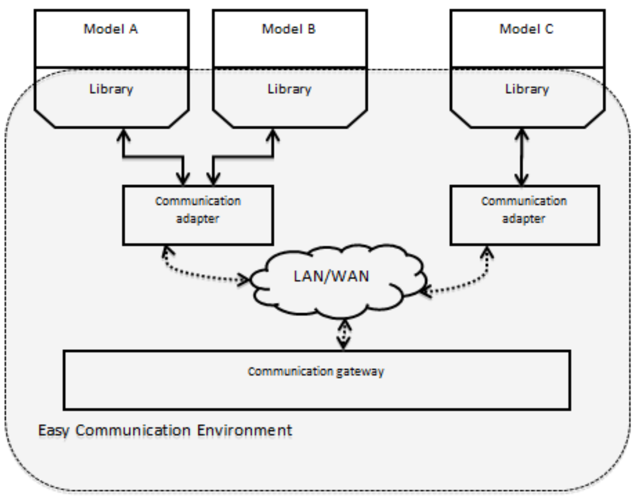


Fig.1. Easy Communication Environment [5].

Communication gateway ensures communication among adapters and it is important that all adapters can be seen by communication gateway. Communication gateway maintains the simulation data registry and data "subscriber" registry. It ensures that data is received by only those adapters that have subscribed to certain data [5].

4 Simulation – One of the Services on the Future Internet and the Cloud

4.1 Future Internet context

It seems that nowadays it is hard to over-emphasize the importance of the Internet and its presence in our everyday routines. Tselentis et al. states that the current Internet designed 40 years ago is today the most important information, service and networking infrastructure providing the mechanisms for the digital society at large to function as an integrated entity [6].

At the same time the increasing dependence on the Internet as a daily tool has tremendous social and economic impact [14]. Within this context a European initiative called "The Future Internet Assembly" (FIA) has been established with the goal to shape the Internet of the future. This initiative, which is backed by a number of European research projects under the EU Seventh Framework Programme (FP7), follows similar activities in the US, Japan, and Korea [8]. The European Commission supports approximately 140 Future Internet projects to a greater or lesser extent and some research of them are related with Cloud (http://www.future-internet.eu/activities/fp7-projects.html). There is a need for both researchers and practitioners to develop platforms made up of adaptive Future Internet applications. Simulation is not an exception, as it can become one of such services in the Future Internet environment [9].

4.2 Cloud-based simulation

Cloud computing is emerging as a novel computing paradigm. A specific feature of it is that computation and storage resources are provided as services. In this way, applications/software can be executed or maintained on the Cloud without the necessity of operating an own local infrastructure. Such a computing model significantly reduces the cost for resource and software management, which is clearly an attractive benefit for small business and research groups. In addition to cost-efficiency, Cloud computing shows other advantages such as on-demand resource provision, supporting legacy codes, service reliability, easy management, and so on [10]. Some well-known commercial Cloud solutions are for instance Google's AppEngine, Microsoft Azure, Sun Grid, Aneka, Amazon's Elastic Computing, Salesforce etc. [7, 11].

Cloud computing offers the promise of outsourcing the task of providing and managing execution platform to users while hiding many of the complicated details of parallel and distributed simulation execution. As such, it offers the possibility of making parallel and distributed simulation technology much more readily accessible to the simulation community. As cloud-computing environments become more common, their use for parallel and distributed simulations becomes more attractive [12].

The potential benefits and technical challenges that arise in utilizing cloud platforms for parallel and distributed simulations and a potential solution approach has been studied by Fujimoto, Malik and Park [12]. It has been observed, they argue, that parallel scientific codes executed over Amazon EC2 ran significantly slower compared to execution on dedicated nodes of a cluster. They stress that Cloud environments are often better at providing high bandwidth communications among applications than in providing low latency and this is problematic for many simulation applications that are accustomed to sending many small messages requiring quick delivery rather than fewer large messages requiring high bandwidth alone. Many of cloud environments in general, according to their research, are not specific to parallel or distributed simulation. For example, users must be assured that proprietary and confidential data will be secure in a cloud environment before they will consider such a move [12].

Another research to be mentioned is the one done by Yamazaki et.al. [15] Their on-going work aims at development of cloud-based web service called "Simulation Platform" for multi-scale and multi-modal neural modelling, and they stress that Cloud-based technology simplifies the work of researchers. They also conclude that such a simulation platform has tremendous applications also for more general use in various academic fields, including the peer-reviewing process and adding functionality to online articles. [15]

But one can outline also another major problem, e.g. while the development of different Future Internet mechanisms involves software engineers; they might lack the perception of possible kinds of services and their specific operation in the Future Internet. The end-result depends on many experts from different fields and their collaboration. These experts define what and how should be developed by software engineers from the users perspective. For example, in logistics some specific aspects of RFID, biometrics and GPS data exchange must be respected.

5 Easy Communication Environment on the Cloud

The use of Cloud computing in various fields of research still has to be tested. Also Buyya et al. indicates the need of Cloud computing and other related paradigms to converge so as to produce unified and interoperable platforms for delivering IT services as the 5th utility to individuals, organizations, and corporations [11].

Therefore the aim of this research is to test the Easy Communication Environment architecture on one of available Cloud computing platforms and consider the possibilities for future operations.

To choose among the Cloud platforms, one had to make sure what the requirements regarding the Easy Communication Environment were. Easy Communication Environment is JAVA based, that means that the platform for running Easy Communication Environment must to have JVM (Java Virtual Machine). Another criteria is related with simulation software. The experiment described in this paper deals with two modelling environments – Extend and NetLogo [5]. Extend operates only on Windows and Mac OS X, and NetLogo also on Linux. Accordingly, the chosen cloud solution had to offer appropriate infrastructure rather than a platform. Therefore Amazon Elastic Compute Cloud (EC2) was chosen for this experiment.

Amazon EC2 provides a virtual computing environment that enables a user to run Linux-based applications. The user can either create a new Amazon Machine Image (AMI) containing the applications, libraries, data and associated configuration settings, or select from a library of globally available AMIs. Amazon EC2 charges the user for the time when the instance is alive, while Amazon S3 charges for any data transfer (both upload and download) [11]. EC2 supports RedHat Linux, Windows Server, openSuSE Linux, Fedora, Debian, OpenSolaris, Cent OS, Gentoo Linux, and Oracle Linux OS server instances. Server instances are administered using we-based interface (AWS Management Console) or using Amazon E2 API.

To integrate the Easy Communication Environment with Amazon EC2 three server instances are needed. Two instances operate with simulation models and the third is reserved for the Communication gateway. Additional three Elastic IP addresses for each instance are needed. (Elastic IP addresses are static IP addresses designed for dynamic cloud computing [13]). These IP addresses are used by Communication adapter to establish a connection with Communication gateway. All server instances use Windows Server OS, respectively one runs Extend and Communication adapter, the second - NetLogo and Communication adapter, and the third – Communication gateway. Figure 2 shows Easy Communication Environment on Amazon EC2. To use simulation tools, user must have direct access to model. Therefore the Remote Desktop Protocol (RDP) is used to control simulation models.

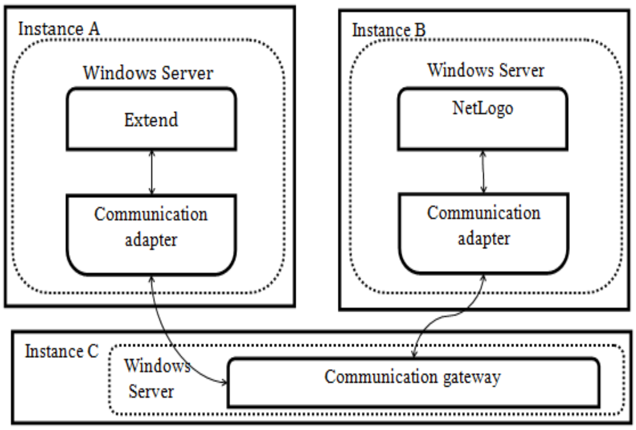


Fig.2. Easy Communication Environment on Amazon EC2.

Amazon EC2 offers „unlimited" server resources to be used for intensive load. Unfortunately Easy Communication Environment architecture in its current stage is not able to use this advantage. It is not possible to make GRID, as each simulation model operates on separate server and this action cannot be divided among several servers.

Easy Communication Environment allows communication among different types of simulation tools, and many of them support a dynamic modeling process with active participation of the user (for example, NetLogo). In this case the use of Amazon EC2 is not beneficial, but is rather inconvenient, because the user has to use RDP. Finally, if the model is deployed on Cloud, then the fact that only one user can work with the model, could also be perceived as a disadvantage, otherwise the great potential of Cloud computing is being neglected.

6 Conclusions

This article discusses the challenges of distributed simulation within the context of different communication environments and the Future Internet development. It proposes and tests the operation of communication environment using Cloud infrastructure. The development of the Internet of Services and Cloud computing creates new challenges for engineering scientists in terms of enhancing the accessibility to distributed simulation and empowering the researchers from other disciplines.

To ensure the communication among different models the user has to apply one of communication environments (HLA, CORBA, WS, etc,) but there are no ready-made solutions that would allow creating multi-model systems. By dealing with communication among models in case-based manner, huge resources are spent on technical issues, not on research.

The concept of "Easy Communication Environment" provides a simple solution where the domain expert is able to "connect" models without the help of software engineers. Easy Communication Environment is reusable and universal at the same time in order to be able to connect as much various models as possible.

The new paradigm of Cloud computing in its way corresponds to the needs of distributed simulation society and is very promising, although recent studies report also some drawbacks as many Cloud environments are not specific to such simulation tasks.

Within experimental running of Easy Communication Environment on Amazon EC2 infrastructure, it was concluded that Cloud computing has no significant benefits for Easy Communication Environment at the moment.

Acknowledgment
The current article was prepared in the framework of EC FP7 project No.287119 FUPOL „Future Policy Modelling".

References:

[1] A. Silins and E. Ginters. "Simulation Data Exchange in Distributed E-learning Environment", in: Proceedings of the 4th WSEAS/IASME International Conference on Educational Technologies (EDUTE 08), Corfu, Greece, October 26-28, 2008, ISBN 978-960-474-013-0, 2008, pp.138-143

[2] R. Murray, DIS Overview and Version 7 Information, Simulation Interoperability Standards Organization, 2010 http://www.sisostds.org/DesktopModules/Bring2mind/DMX/Download.aspx?Command=Core_Download&EntryId=29289&PortalId=0&TabId=105 10.11.2011.

[3] "Welcome to the CORBA Directory.", Object Management Group, Inc, http://corbadirectory.omg.org/, [accessed 07.11.2011]

[4] Gray N.A.B. 2004. "Comparison of Web Services Java-RMI, and CORBA service implementations", in Proceedings of Fifth Australasian Worshop on Software and System Architectures, Melbourne, Australia, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.63.1265&rep=rep1&type=pdf#page=60

[5] A. Aizstrauts, E. Ginters, D. Aizstrauta, "Easy Communication Approach for Data Exchange in Distributed Simulation Environment" in Proceedings of the13th WSEAS International Conference on Automatic control, modelling &simulation (ACMOS 11), May 27-29, 2011, Lanzarote, Canary Islands, Spain, ISBN: 978-1-61804-004-6, ISSN: 2223-2907, pp. 34-39.

[6] G. Tselentis et al, Towards the Future Internet, IOS Press, 2009 ISBN 978-1-60750-007-0 p.359

[7] C. Pedrinaci, J. Domingue "Toward the Next Wave of Services: Linked Services for the Web of Data," in Journal of Universal Computer Science, 2010

[8] D.Hausheer et al. "Future Internet Socio-Economics – Challenges and Perspectives," in Towards the Future Internet. G.Tselentis et al.(Eds.) IOS Press, 2009 ISBN 978-1-60750-007-0 (pp. 1-11)

[9] E. Ginters, I. Sakne, I. Lauberte, A. Aizstrauts, G. Dreija, R.Maria Aquilar Chinea, Y.Merkuryev, L.Novitsky, J.Grundspenkis, "Simulation Highway – Direct Access Intelligent Cloud Simulator," in Proceedings of 23th Europen Modelling & Simulation Symposium (EMSS 2011), ISBN 978-88-903724-4-5, 12-14 September, 2011, Rome, Italy, pp.62-72.

[10] J.Tao, H.Marten, D. Kramer, W.Karl, "An intuitive framework for accessing computing clouds," in Procedia Computer Science 4 (2011) 2049-2057

[11] R.Buyya,Y.C Shin, S.Venugopal, J.Broberg, I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," in Future Generation Computer Systems 25 (2009) 599-616

[12] M.R.Fujimoto, A.W.Malik, J.A.Park, "Parallel and Distributed Simulation in the Cloud,“ in SCS M&S Magazine – 2010/n3

[13] Feature Guide: Amazon EC2 Elastic IP ddresses, http://aws.amazon.com/articles/1346 [accessed 12.11.2011]

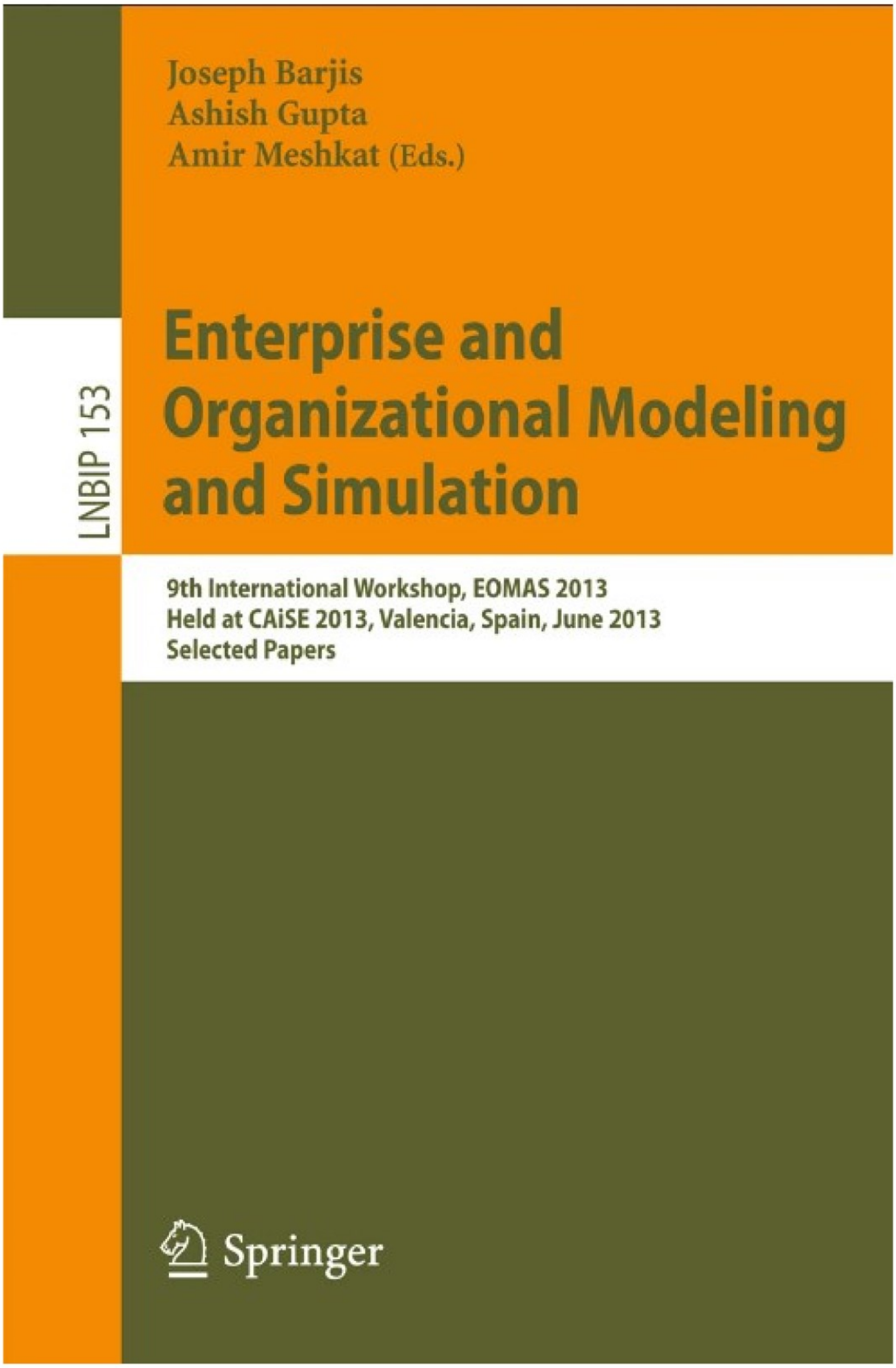
[14] C. Blackman, I.Brown et al. Towards a Future Internet, 2010. "Interrelation between Technological, Social and Economic Trends. Final Report for DG Information Society and Media". European Commission DG INFSO Project SMART 2008/0049, November 2010, Available from: http://www.internetfutures.eu/wp-content/uploads/2010/11/TAFI-Final-Report.pdf, [accessed 13.11.2011]

[15] T. Yamazaki, H.Ikeno et al. "Simulation Platform: A cloud-based online simulation environment," in Neural Networks Vol 24, Issue 7, September 2011, pp 693-698

[16] J.S.Dahman, R.M.Fujimot, R.M. Weatherly, "The Department of defense high level architecture," in Proceedings 1997 Winter Simulation Conference, 1997

[17] VTMÄK, http://www.mak.com/resources/industry-standards/hla--high-level-architecture.html [accessed 10.11.2011]

9. Aizstrauts, A., Ginters, E., Lauberte, I., & Piera Eroles, M.A. (2013). Multi-level Architecture on Web Services Based Policy Domain Use Cases Simulator. In *Lecture Notes in Business Information Processing* (vol 153, pp.130-146). Berlin: Springer. doi:10.1007/978-3-642-41638-5_9. Retrieved from https://www.researchgate.net/publication/282289690_Multi-level_Architecture_on_Web_Services_Based_Policy_Domain_Use_Cases_Simulator#fullTextFileContent


Joseph Barjis
Ashish Gupta
Amir Meshkat (Eds.)
Enterprise and Organizational Modeling and Simulation
LNBIP 153
9th International Workshop, EOMAS 2013
Held at CAiSE 2013, Valencia, Spain, June 2013
Selected Papers
Springer

Multi-level Architecture on Web Services Based Policy Domain Use Cases Simulator

Artis Aizstrauts[1], Egils Ginters[1], Ieva Lauberte[1]

[1] Sociotechnical Systems Engineering Institute
Vidzeme University of Applied Sciences
Valmiera, Latvia
e-mail: {artis.aizstrauts, egils.ginters, ieva.lauberte}@va.lv

Miquel Angel Piera[2]

[2] Dept. de Telecomunicació i Enginyeria de Sistemes
Universitat Autònoma de Barcelona
Barcelona, Spain
e-mail: miquelangel.piera@uab.es

Abstract. The FP7 FUPOL project aims at a completely new approach to traditional policy modeling providing complex domain use case verification with the FUPOL Simulator and visualisation of the results in a form suitable for beneficiaries. Policy domain use case models are diverse and versatile, therefore architecture of the simulator must fit complexity and usability requirements that determine involving heterogeneous agent-based and system dynamics simulation technologies and an appropriate implementation comprising on SOA based distributed models joined by Easy Communication Environment (ECE). The article deals with designing the FUPOL Simulator architecture, and the benefits, and drawbacks of the approach used.

Keywords: Agent-based simulation (ABM/MAS), FUPOL, Policy simulation, Distributed simulation, System dynamics, Service oriented architecture (SOA), Land Use Category Change (LUCC), Easy Communication Environment (ECE).

1 Introduction

Mathematically justified planning of economic policy, including regional policy, is extremely important because every wrong decision can lead to serious and even irreversible consequences. The economic crisis, which still significantly affects all countries, would be easier to overcome if all decisions would be based on right analytical solutions [1]. It is no secret that mathematical modelling and other advanced analytical methods are rarely used for policy planning and decision impact forecasting because they require specific knowledge. As a result, public and private sectors as well as institutions make wrong decisions on economic governance, taxation, social and economic development, monetary policy etc. Global economy planners such as International Monetary Fund (IMF) chief economists Olivier Blanchard and Daniel Leigh are no exception [2]. They have admitted mistakes in predicting the impact of austerity on European economics. The planners had underestimated the rise in unemployment and the decline in domestic demand related to fiscal consolidation, as well as under-predicted the likely impact of reforms on economic growth. As a result, Greece, Spain and Portugal were required to make swift budget cuts, which could lead to social tensions and disrupt national economies. The errors were caused by choosing incorrect action impact assessment methods and avoiding to take into account a considerable number of crucial factors that are typical for policy modeling. This is a situation where simulation technologies can be applied.

Using simulation solutions, policy planners can examine a variety of real-time solutions before their implementation as well as forecast the impact of potential decisions on the attainable goal in general [3].

Policy modeling as the object of research is versatile and complex. FP7 FUPOL project (see www.fupol.eu) aims at a new approach to traditional policy modeling. Major innovations like multichannel social computing and crowd sourcing used for data gathering change the way politicians communicate with citizens and enterprises, and make decisions. The simulation and visualisation tools assist governments and policy makers in the whole policy life cycle and allow avoiding voluntary and wrong decisions. In FP7 FUPOL the set of policy domains as the object for analysis and simulation [4, 5] involves community facilities (area design, open space), urban segregation and economics, edge land industrialization, sustainable tourism and other. Each comprises some use cases.

Policy use case models, depending on the domain, could be described as discrete or continuous, and determined or stochastic systems. For simulation of the above-mentioned models different and heterogenic simulation tools could be used. Simulation scenarios can ask for collaboration between separate simulation models for the implementation of the task or policy domain use case, i.e. distributed and/or multi-level simulation must be realised.

A wide set of different policy simulation tools exist in the world [6]. However their use is limited by incompatibility and orientation mainly to IT professional's bothering elaboration of wide spread reliable and sustainable stimulators, which could fit for deployment on the Future Internet and/or Cloud to ensure their accessibility [7].

Under the framework of the FUPOL project more than 60 different generic simulation tools, domain oriented packages and project applications were reviewed [6]. Unfortunately only a few fit the requirements set defined for the design of the FUPOL Simulator.

The objective of this article is to give insight into FUPOL Simulator architecture design, as well as benefits and drawbacks of the provided solution.

2 Methodology, Requirements and Findings

The FUPOL project policy modeling platform consists of different, however joined work packages, which collaboration is aimed to creating new knowledge and providing new possibilities to policy domain specialists related to forecasting of potential decision effect making results through adding modern supporting tools as semantic search, simulation and visualisation. Policy domain and use case model identification is carried out in collaboration with beneficiaries [8]. The policy modelling work package is responsible for policy use case model design and verification, which is done using Coloured Petri Networks (CPN) and Fuzzy Cognitive Maps (FCM) [9]. The FCMs, CPNs and ABM/MASs approaches are combined to achieve a certain level of transparency by describing the causal models as a set of behaviour rules also documented in a natural language for beneficiaries' needs. Analytical equations are used if system dynamics simulation is requested for a predefined domain use case implementation.

The methodology defined to generate ABM/MAS based policy use case causal models consists of different stages. FCMs, CPNs and ABM/MAS approaches are combined in order to achieve a certain level of transparency by describing the causal models as a set of behaviour rules (they could also be described in a natural language). A set of preliminary rules are defined using the information and data obtained from beneficiaries (field work), together with a review of scientific literature and also after some face to face meetings between modellers and pilot experts. CPNs and FCMs are internally used to define the agents' behaviour and, in the case of CPNs, also to verify the rules by studying their state space reachability.

CPN formalism allows specification rule based system dynamics as a formal language in which it will be possible to determine if the rules are consistent with the observed system dynamics, which dynamics have been properly formulated, which system states can be reached using the rules and check which rules should be added to reach certain final system states. Rules can be seen as a relationship between precedent conditions and a consequent body. This form of rules can be interpreted in CPN formalism as a set of pre-conditions, which must be satisfied in order to fire an event, and a set of post-conditions, which represent the new state of the system reached after firing the event. Each rule can be formulated as a transition, in which the pre-conditions will be formulated by means of input arc expressions of the place nodes connected to the transition, and the post-conditions will be computed by means of output arc expressions connected at the output place nodes. This one-to-one representation between rules and CPN transitions is a positive feature of FUPOL models to improve simulation transparency. One of the main advantages of analysing the rule based model using CPN formalism is that the state space of the system can be computed without considering particular time constraints (time events) and particular stochastic factor constraints as well. Thus, the full state space of the system can be computed providing all event sequences that could occur in the real system together with the evolution of the system (state variables) from an initial state to the different final states. In case a feasible final state is never achieved, it is possible to check why the conditions for an event are not satisfied and modify the rules (i.e. the transitions) or add new rules (i.e. new transitions) to achieve an acceptable representation of the system.

The constructing of an FCM model requires the identification of concepts to be included in the model. This is mainly carried out jointly by the decision-maker and the domain experts. Once all concepts have been agreed upon, the fuzzification procedure is carried out. Fuzzification essentially involves employing a membership function to break down a concept into a number of fuzzy (overlapping) sets in the range [-1, 1] and assigning a linguistic value that best describes the state of a concept within the boundaries of each fuzzy set. Next, each node is initialized with a numeric value (known as its "activation level") in the range of [-1, 1] to qualitatively signify its current presence or state in the problem. In general a value closer to -1 indicates that the concept has a strong negative presence, leading to inhibiting effects in the problem, while a value closer to 1 indicates a strong positive presence, leading to promoting effects in the problem. After consulting various policies domain use case experts the FCM will be finalized and all descriptions of the concepts and their corresponding activation levels, the causal relationships and their normalized weights are denoted, and an ABM/MAS simulation policy use case model can be designed.

The set of rules validated using CPN models are the base to define the agents' behaviour. The ABM/MAS models will be composed of basic agents (citizens, industry etc.) and their interaction, the observer agent and multi criteria objective function. The model will be evolving over time and agents will make decisions and modify behaviour depending on their interactions with other agents and the state of the system. The observer agent will be in charge of finding a trade-off between all the indicators of the multi criteria objective function.

Verified policy model use case specifications in transparent, understandable and formalized form (agent specification, agent interaction specification, time specification, flow model specification, system dynamics model, interoperability and data exchange specification, drivers, boundary conditions, quantitative parameters, input data sources and links, and other notes) [9] with explanations in natural language are transferred to design FUPOL Simulator software and run use case simulation models. As a result of running the simulation software, the generated simulation data generated is analysed by the Policy modelling work package using the data mining approach ANFIS [10] to refine the final FCM relations and their weights. Beneficiaries can use visualised FCM to better understand the causal effects.

Validation of the developed simulation model is performed at two different levels:

- Academic Validation is in charge of modeling, not simulation, because not all real, involved agents are represented in the model. The full State Space of the CPN model will be analysed to check if the different reachable states that could be obtained in a real context negotiation for the deployment of an urban policy can be generated by the rules codified in the ABM/MAS environment;
- Beneficiaries Acceptance: Once the simulation code is available, the different experts of the beneficiaries participate in simulation results analysis and consider different scenarios and boundary conditions to accept/reject the results obtained.

The methodology described above determines two groups of requirements to the FUPOL Simulator software design tools and environments:

- Policy domain requirements to simulation software;

- Technical requirements related to the FUPOL software platform.

The summary of the requirements for the FUPOL Simulator software design platform is given in Table 1.

The FUPOL application domains are formulated in [4]. However, to diminish potential mistakes technical and domain requirements were validated. Statistical assessment using Kolmogorov-Smirnov tests with confidence alpha 0.95 approved the credibility of obtained results. The analysis confirmed that user requirements are versatile and distinctive. That determines the design of the two-level FUPOL Simulator, which supports simulation on micro and macro levels in order to avoid bottlenecks in further use case model simulation. The ABM/MAS model simulation will be realized at a micro level. The macro level will be applied to long-term change forecasting simulations and be based on SD tools as optional (if necessary). In the last years similar ideas have been realised in different products, for example, Insight Maker [11].

The data exchange and communication between ABM/MAS and SD simulation models is ensured by some communication environment and PostgreSQL/PostGIS DB. If the simulation scenario must be implemented as a queue of some simulation models then distributed simulation and models routing must be used. The FUPOL Simulator must be the joint part of FUPOL software and ensure necessary interoperability. The FUPOL Simulator must collaborate with GIS GeoServer as a basic platform for spatial data processing. The FUPOL Simulator is required by WFS/WMS (and other web services), XML data use and SQL requests implementation.

The FUPOL Simulator must have a convenient GUI and functionally understandable simulation desktop not only for the implementation of input-output operations, but also for simulation initialization, calibration and control. It is reasonable that Java is used as a tool for add-on programming.

It is strongly recommended that FUPOL Simulator is an open source freeware solution ensuring web access only through a browse. The FUPOL Simulator must be open and adaptable to different operating environments including deployment on the Future Internet and/or the Cloud.

In order to select the FUPOL Simulator design software platform, generic simulation tools and domain oriented simulation packages were analysed in conformity with requirements mentioned in Table 1.

More than 32 generic simulation software tools have been assessed paying attention to the software, which is more probable to being used for FUPOL Simulator software design [6]. The review consisted of 3 SD tools (Stella, Vensim, Powersim), discrete-event (DEVS) simulation packages, 8 agent-based (ABM/MAS) software packages (AgentSheets, AntLogic, Ascape, NetLogo, StarLogo, MadKit, RePast Symphony, Mason), and cellular automatons (CA), micro analytic solution tools and tools pertaining to multi-level simulation and learning, and evolutionary simulation.

Table 1. Requirements for FUPOL Simulator software design tools.

Criterion	Values	Threshold
Simulation technology	system dynamics (SD), discrete-event systems (DEVS), agent-based approach (ABM/MAS), cellular automatons (CA), learning and evolutionary simulation etc.	ABM/MAS must be ensured and SD as optional
Licensing	commercial, freeware, shareware, and open-source, price, rules of usage	must be open source and freeware
Spreading and sustainability	how widespread and how long existing, technical support	widespread with stable technical support
Resolution and scalability	model dimensions and resolution	The highest model dimension and/or scalability possible, no less than 1000x1000 or 50m per cell (ABM/MAS)
Documentation	Quality of software documentation (manuals, examples etc.)	well documented
Architecture	desktop, client-server, web services, distributed models	web services and SOA design possibilities, and distributed models implemented as optional
Interoperability	with other platforms, environments and tools, web services	The platform must be open
Environment	operating system, specific environment, programming languages, compiler or interpreter	wide deployment environment including virtual machines and deployment possibilities on the cloud
Applications	scratches, libraries, Geographical Information Systems (GIS), different domains	existence of GIS applications (WFS/WMS services) and possibilities for access to PostgreSQL/ PostGIS DB
Graphic User Interface (GUI)	possibilities aimed at convenient simulation desktop design	compatibility with Java extensions

In addition, LUCC tools, combined domain simulation packages, different approaches and descriptions for more than 20 applications, int.al European level modeling projects were reviewed. A detailed review was done on six products – UrbanSim, Metronamica, WhatIf?, FEARLUS, SLEUTH, LandUse Scanner.

The aforementioned analysis confirmed that domain oriented simulation packages practically do not correspond to the FUPOL Simulator software design requirements, but generic simulation tool Repast Symphony can be used as a basis for further work. The tool will be used for the design of ABM/MAS based policy use case simulation models. The SD simulation software must also be realized using the same simulation tool ensuring the functionality is optionally similar to the STELLA [12]. The FUPOL Simulator GUI and other necessary software must be implemented in Java.

Further chapters are aimed at discussing of the FUPOL Simulator architecture elaborated in conformity with requirements defined above.

3 Structural Model of FUPOL Simulator

The FUPOL Simulator is responsible for implementing policy use case model simulation elaborated under framework of the FUPOL project.

The simulation is ensured at two levels: micro and macro simulation (see Fig.1). At the micro level agent-based simulation (ABM/MAS) operations related to versatile and small basic components interaction and forecasting of the interaction results are performed. ABM/MAS could be used for forecasting of continuing changes, however the use of SD simulation is more reasonable. Therefore the FUPOL Simulator is designed as a two level system. At the macro level (if it is necessary) the SD simulation is implemented. In this case the micro level is the data source for the macro simulation model. If the scenario requests simultaneous interoperability between the set of models, distributed simulation is performed using the Easy Communication Environment (ECE) [13, 14, 15].

ECE is communication mechanism provided for data exchange among simulation tools and models. The basis is a reduced High Level Architecture (HLA) [16] exchange mechanism based on broadcasting principles. One of the main statements of ECE is simplicity of use to promote the design of distributed simulation models for domain specialists who have no specific knowledge in programming.

The FUPOL Simulator software (see Fig.2) management part consists of a Simulation Initialization and Control block (GUI) and Multilevel and Distributed Simulation Models Management unit.

The Simulation Initialization and Control block (GUI) is responsible for:

- Restrictions and limitations (red lines, red polygons);
- Drivers and Categories (selection, creation, weighting);

- Simulation Algorithm Parameterization (probability, distance or surrounding, switching on-off);
- Initial Simulation Window Marking, Zooming and Selection;
- Data Input (CORINE, INSPIRE, EUROSTAT, User Data);
- Data Output (CORINE, INSPIRE, User Data);
- Data Visualisation (map, tables, graphs, set of pictures, movie);
- Launching (Start, Stop, Back);
- Timing (time periods);
- Interaction with FUPOL Software Core platform (authentication, authorization, GIS, DB and Data access).

Multilevel and Distributed Simulation Models Management unit ensures:

- Simulation Platform selection (ABM or SD platforms);
- Simulation Models Repository Management;
- Simulation Models Connection, Routing and Multi-level management (collaboration with ECE and the Future Internet, and/or the Cloud).

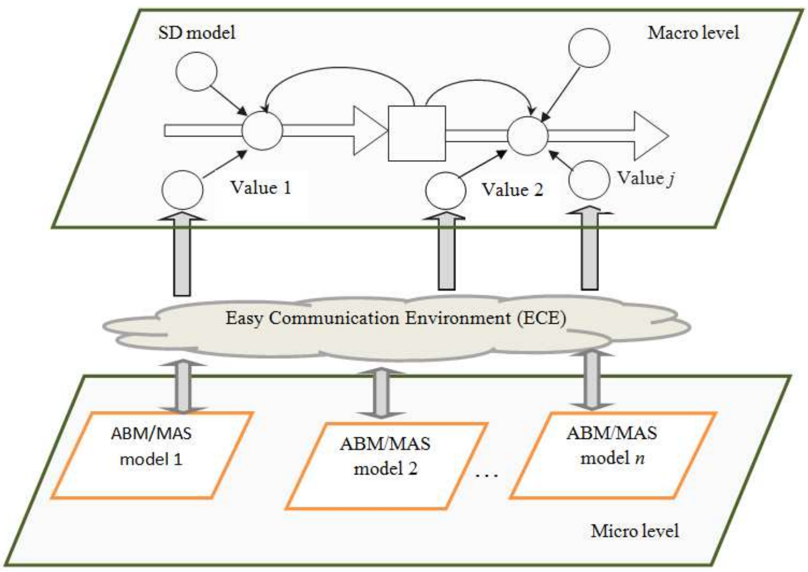


Fig.1. ABM and SD simulation model integration in the preliminary architecture of the FUPOL Simulator.

The Agent-Based Micro Model Repository comprises ABM/MAS models that are aimed at FUPOL policy use case simulation. Perhaps, for one scenario separate model interaction is necessary or some data must be sent to the macro level for SD simulation. The operations mentioned above are realized by the Multi-level and Distributed Simulation Models Management block, which is responsible for selecting the necessary Simulation Platform to run distributed models.

The Simulation Initialization and Control block (GUI) is responsible not only for simulation process control, but also for the interaction with the FUPOL Core platform Enterprise Service Bus (ESB) [17] to access spatial data and the database (FUPOL GIS DB). The Controls block also manages the simulation desktop to control the simulation process and ensure visualization of the intermediate and final results. The FUPOL GIS DB is used for storing intermediate and final simulation session.

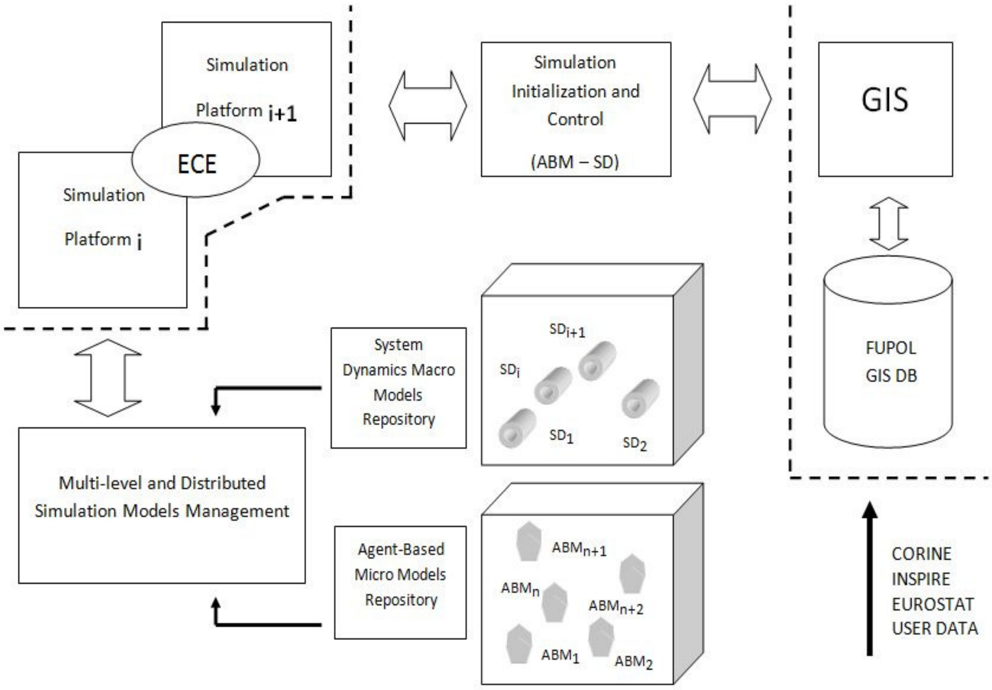


Fig.2. FUPOL Simulator preliminary conceptual architecture.

The FUPOL Simulator is part of the FUPOL software system and therefore must comply (with some constraints) with the interoperability requirements accepted in the FUPOL software environment.

4 Functional Model of FUPOL Simulator Architecture

The FUPOL Simulator is one of the FUPOL project software services. The FUPOL software services use Enterprise Service Bus (ESB) for communication.

The Visualization service, which is the main recipient of FUPOL Simulator output, receives data directly from PostgreSQL/PostGIS DB and GeoServer or uses ESB to access intermediate and final simulation session results.

The FUPOL Simulator uses PostgreSQL/PostGIS DB, GeoServer and other FUPOL Core Platform services (see Fig.3).

Fig.3 shows that every policy domain use case simulation model has two main elements – Model (simulation model) and Web service. Model does all the modeling, but Web service allows other FUPOL software services to exploit the particular policy domain use case simulation model. Other core platform services access policy domain uses cases simulation models through Simulation Initialization and Control (GUI) (see Fig.2) blocks.

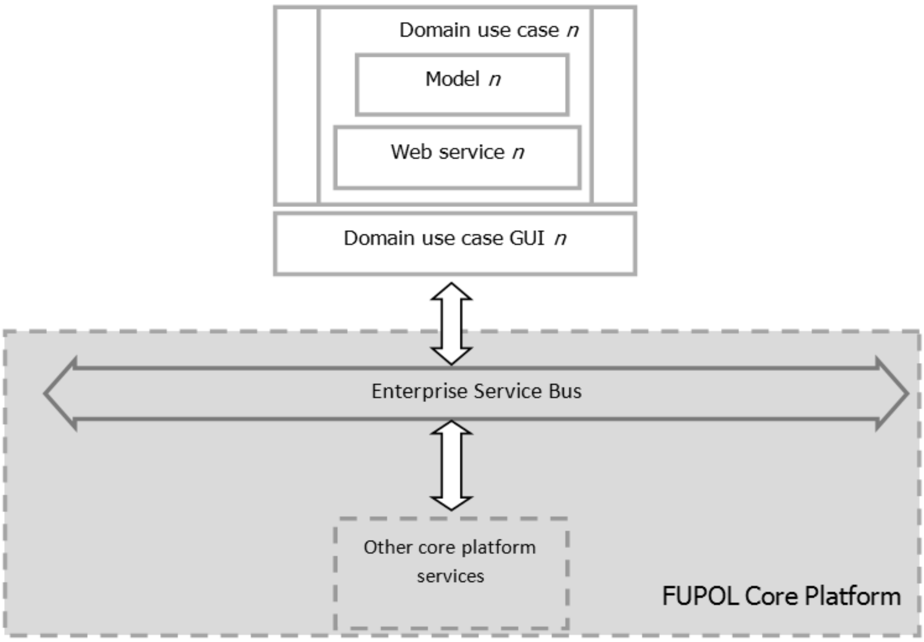


Fig.3. FUPOL Simulator collaboration with "Other core platform services".

The Web service accepts SOA Protocol (SOAP) requests over Hyper Text Transfer Protocol (HTTP). Each policy domain use case simulation model Web service describes its functions using Web Services Description Language (WSDL). All FUPOL software services that intend accessing the policy domain use case Web service must implement SOAP clients using the particular domain use case Web service WSDL.

PostgreSQL/PostGIS DB (see Fig.4) is used to store different data, which can be useful to other modeling tools or FUPOL software services (for example, the Visualization service). Access to the DB is established through ESB. Policy domain use case simulation models use Java Database Connectivity (JDBC) to make PostgreSQLPostGIS DB connections.

Domain use case GUI blocks represent user interfaces for each domain use case simulation model. The FUPOL Simulator user uses a particular domain use case GUI to manipulate with specified domain use case simulation model parameters and view modeling results. The domain use case GUI is built for each domain use case simulation model separately, meaning – each GUI is built based on the particular nature of the domain use case and FUPOL Simulator users' expectations about the usage of this particular domain use case model. Each GUI is also a SOAP client and makes SOAP requests to the Web service of the particular domain use case. The domain use case GUI is made using HTML and JavaScript, but the SOAP client is coded with Java JAX-WS technologies that are implemented into FUPOL Core platform as a Java servlet, which handles GUI requests and transforms them into SOAP requests.

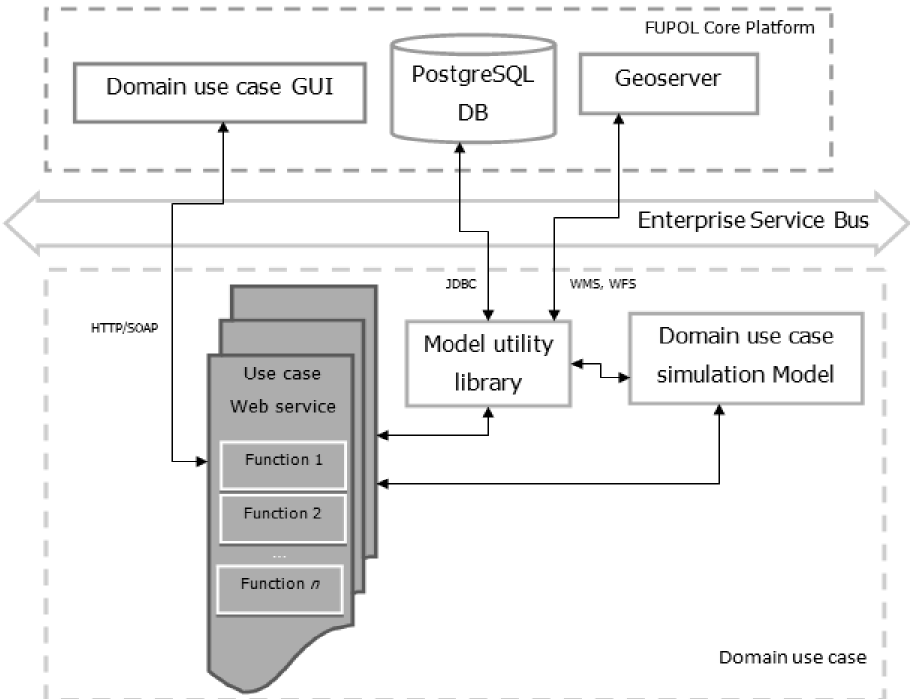


Fig.4. FUPOL Simulator policy domain use case simulation software architecture.

GeoServer is one of the FUPOL software services and can be accessed via ESB. The FUPOL Simulator stores spatial data in GeoServer, reads it later and even reads maps that are stored by other services. GeoServer supports requests by the following services - Web Feature Service (WFS), Web Map Service (WMS) and Web Coverage Service (WCS). For example, the LUCC domain use case simulation model uses GeoServer CORINE [18] maps as the main initial spatial data.

Fig.4 shows the FUPOL Simulator domain use case simulation software architecture and collaboration of its main components with other FUPOL software services through ESB. The main components of the domain use case simulation software are – Domain use case simulation Model, Model utility library, Use case Web service and Domain use case GUI (Simulation initialization and control block).

The FUPOL Core platform provides authorization/authentication for the policy domain use case GUI. FUPOL software service users can access the FUPOL

Simulator only by passing FUPOL Core platform authorization. The domain use case GUI receives user information from the FUPOL Core platform. Afterwards user information is forwarded to the domain use case simulation model and Model utility library. The domain use case simulation model may need user input during the simulation session, for example, to include a user's previous session data in current simulation.

To ensure that the interface of any domain use case is uniform, Web services operating on uniform principles are used separately from domain use case simulation models and the way they are developed. Domain use case simulation model development does not need to follow any unified interface and is mostly implemented in Repast Symphony [19]. Therefore the Web service is necessary to allow other FUPOL software services access to the FUPOL domain use case simulation model in a uniform way.

A domain use case Web service is a web application that implements Web service specifications. The domain use case web server as web application are hosted on an Apache Tomcat 7 web server. It is an open source software implementation of Java Servlet and Java Server Pages technologies. Apache Tomcat 7 requires Java 6 or later [20]. Apache Axis2 is used for Web service deployment as a Web service / SOAP / WSDL engine. The Apache Axis2 web application itself is deployed on Apache Tomcat 7 web server. Apache Axis2 ensures WSDL generation and SOAP request/response handling over deployed Web services.

The Model utility library is a library that promotes the functional operation of a domain use case simulation model. Functionality that is not directly related to modeling is transferred from the model to his library. The Library includes functionality that is required by particular domain use case simulation models or other FUPOL software services, for example the GUI or Visualization service. For example, in the case of the LUCC domain use case, the library may include an informative function that returns land use values. The Library was developed with Java and is platform-independent. It can be run on wide set of operating systems. The model utility library is stored as a JAR (Java Archive) file and other Java based programs, such as domain use case simulation model, can use it as a library.

The domain use case model utility library can have functionality that may require DB usage. The utility library uses PostgreSQL/PostGIS to retrieve and store information related to a particular domain use case simulation. It uses PostgreSQL JDBC Driver to connect to DB.

The domain use case simulation model is written in Java using generic simulation software Repast Symphony. The model is an executable Java JAR archive that can be used by other Java programs or be executed as stand-alone software. During domain use case simulation the model is supposed to be launched only by web service functions. The domain use case simulation model has only two connecting data flows, one with the domain use case Web service and another with the Model utility library. The domain model implements the Model utility library to use its functionality for its own purpose, for example, in the LUCC example, the domain use case simulation model uses spatial data from GeoServer.

5 Verification and Validation

The LUCC use case was selected for the verification and validation of the FUPOL Simulator design approach. This allowed checking architectural solutions, selected generic simulation tools, interoperability with FUPOL Core platform testing, spatial and statistical data processing, GUI designing and data visualisation possibilities testing. The LUCC use case was selected because it is a basis for further use case simulation. The LUCC case is based on well-known modeling algorithms [21]. The user has possibilities to visualise the simulation region map with an overlaying CORINE land-use categories map. The land-use description table is accessible in conformity with CORINE codification. During simulation red lines (polygons) can be drawn on the desktop with limited LUCC possibilities. In the legend, influence (weight) of each land-use category can be selected for the simulation session. The vicinity of changes and simulation duration are defined by the user. After simulation, all intermediate and final spatial and statistical results are stored and visualised.

The verification and validation of the approach is a critical task and there are a number of procedures that were carried out in order to help obtain confidence in the provided approach. These confidence-building activities included:

- Historical validation tests: Comparing the results of the FUPOL Simulator results to known historical values of land-use changes in a predefined region. The statistical paired T-tests show that paired t-statistic calculated in comparison with values corresponding to t-distribution confirm a high probability that the FUPOL Simulator LUCC use case generated simulation results are paired with historical data changes in a region if voluntary policy decision making events were extracted of the representative sample;
- Extreme value verification tests: Testing how the approach behaves when extreme values are entered. Does the approach behave as one would expect or does it exhibit wild behaviour? The verification confirmed that the FUPOL Simulator with LUCC use case works in conformity with the intentions of the simulator's designers;
- Comparative validation: Comparing the simulation results with other similar products. The FUPOL Simulator LUCC use case simulation results were compared to a Metronamica [22] and MOLAND [23, 24] stimulation output series. Statistical Mann-Whitney and Wilcox tests gave similar results approving that with confidence alpha 0.95 all the series are paired and therefore the FUPOL Simulator LUCC application use case can be considered as validated;
- Face validation: Conformity comparison of the designed FUPOL Simulator with requirements defined in Table 1 reveals that simulator fits the previously defined requirements.

6 Conclusion and Future Work

Policy modeling is a complex task due to a huge amount of important influencing factors. Most policy decision makers are not familiar with modern modeling technologies, mathematics and ICT tools. Therefore, a great amount of decisions are intuitive, voluntary and wrong. To change the situation under the framework of FP7, the FUPOL project is intended to develop a new approach in policy decision making, which is based on advanced semantic search methods, data mining, simulation and visualising technologies use granting possibilities for policy decision makers to verify their decisions before introduction in real life. Policy domain use cases are different and complex, simulation software design is labour-consuming, therefore previously used simulator design approaches do not fit today's requirements. The use case simulation scenarios would ask for distributed simulation use and wide spread, easy and direct access to the simulation tools for policy decision makers. Unfortunately, that today that is problematic. The FUPOL Simulator designing approach promotes:

- Multi-level architecture of FUPOL Simulator ensures simultaneous running of heterogeneous use case simulation models enhancing performance of the simulation session;
- Easy Communication Environment (ECE) use supporting easy implementation and running of distributed use case simulation models;
- The modular structure of the simulator provides easy step by step adding of other use case simulation models;
- Belonging to the open source and freeware software group, and granting multilingual support to the potential users network, reducing errors in simulation software and also in policy decision making;
- Web services SOA based architecture enhances integration with other decision making and service tools;
- The FUPOL Simulator is ready for deployment on the Future Internet and/or the Cloud;
- The FUPOL Simulator fits the requirements of beneficiaries, if during validation voluntary accepted policy decisions are excluded from the historical data.

The architecture described in the article will be further developed under the FP7 FUPOL project framework introducing collaboration between the simulation model and FCM, where FCM will be used for weight and impact interactive assessment and correction of the simulation driver. The suitability of virtual and augmented reality (VR/AR) applications will be explored to tune the visualisation services.

Acknowledgments. The FUPOL Simulator described above has been developed under the framework of FP7-ICT-2011-7 IP project FUPOL No. 287119 (2011-2014) "Future Policy Modelling".

References

1. Simulation technology could help prevent future financial crises. European Commission. IP/09/1841. November 2009, http://europa.eu/rapid/pressReleases, Accessed 12/02/2013
2. Blanchard, O., Leigh, D.: Growth Forecast Errors and Fiscal Multipliers, IMF Working Paper, January 2013, http://www.imf.org/external/pubs/ft/wp/2013/wp1301.pdf, Accessed 12/02/2013
3. Silva, S., Fidalgo, J. N., Dalila, B. M.: Fontes. A simulation based decision aid tool for setting regulation of energy grids with distributed generation, Springer Berlin / Heidelberg, ISSN 1109-2858 (Print), 1866-1505 (Online) (2009)
4. Sonntagbauer, S., Hassapi, A., Tomic-Rotim, S., Neophytou, H., Piera Eroles, M.A., Mujica Mota, M.A., Palmisano, E.: Deliverable 2.1 – FUPOL Guidelines on Policy for Cities and Municipalities, 325 p. (2012)
5. Piera, M.A., Mújica, M., Moise, M., Neofytou, H.: Deliverable 2.2 – FUPOL Cognitive and Causal Models for Prototype, 271 p. (2012)
6. Sonntagbauer, S., et. al.: EC FP7-ICT-2011-7 IP project FUPOL No. 287119 (2011-2014) "Future Policy Modelling (IP)", Deliverable 4.1 – FUPOL Simulator Software Requirements Report, 231 p. (2012)
7. Ginters, E., Sakne, I., Lauberte, I., Aizstrauts, A., Dreija, G., Aquilar Chinea, R.M., Merkuryev, Y., Novitsky, L., Grundspenkis, J.: Simulation Highway – Direct Access Intelligent Cloud Simulator. In: Proceedings of 23th Europen Modelling & Simulation Symposium (EMSS 2011), ISBN 978-88-903724-4-5, 12-14 September, Rome, Italy, pp.62-72 (2011)
8. Buil, R., Piera, A.A., Tomic, S., Bandic, M., Boyong, W., Boscolo, P., Matheou, M., Palmisano, E.: Deliverable 2.3 – FUPOL Model Parameterization. Draft Version, (2012)
9. Sonntagbauer, S., Ginters, E., et. al.: EC FP7-ICT-2011-7 IP project FUPOL No. 287119 (2011-2014) "Future Policy Modelling (IP)", Deliverable 4.2 – FUPOL Simulator Software Interfaces Specification, 78 p. (2012)
10. Xue, Li, Zaiane, O.R., Zhanhuai, Li: Advanced Data Mining and Applications: Second International Conference, ADMA 2006, Xi'an, China, August 14-16, 2006, Proceedings, Springer, 1110 p. (2006)
11. Bellinger, G., Scott: Beyond Connecting the Dots, Kill Devil Hills, NC, 2012, http://insightmaker.com, Accessed: 12/02/2013
12. ISEE Systems, STELLA – Systems Thinking for Education and Research. http://www.iseesystems.com/softwares/Education/StellaSoftware.aspx, Accessed 24/12/2012
13. Aizstrauts, A., Ginters, E., Aizstrauta, D. and Sonntagbauer, P.: Easy Communication Environment on the Cloud as Distributed Simulation Infrastructure. //In: Proceedings of the 5th WSEAS World Congress on Applied Computing Conference (ACC '12), Recent Advances in Computing Engineering Series 2, ISBN: 978-1-61804-089-3, ISSN: ISSN: 1790-5109, 2-4 May, Faro, Portugal, pp. 173-179 (2012)
14. Silins, A., Ginters, E., Aizstrauta, D.: Easy Communication Environment for Distributed Simulation. In: World Scientific Proceedings Series on Computer Engineering and Information Science 3 "Computational Intelligence in Business

and Economics”, Proceedings of the MS’10 International Conference, Barcelona, Spain, July 15-17, 2010, ISBN 978-981-4324-43-4, pp.91-98 (2010)
15. Aizstrauts, A., Ginters, E., Aizstrauta, D.: Easy Communication Approach for Data Exchange in Distributed Simulation Environment. In: Proceedings of the 13th WSEAS International Conference on AUTOMATIC CONTROL, MODELLING & SIMULATION (ACMOS '11), May 27-29, 2011, Lanzarote, Canary Islands, Spain, ISBN: 978-1-61804-004-6, ISSN: 2223-2907, pp. 34-39 (2011)
16. Perumalla, K.: Tutorial. Handling Time Management under the High Level Architecture, http://www.ornl.gov/~2ip/doc/perumalla-tutorialslides-iitsec06.pdf, Accessed 15/11/2011
17. Welcome to the Mule Community. Mule ESB – The easiest way to integrate anything, anywhere, http://www.mulesoft.org/, Accessed 24/12/2012
18. CORINE Land cover, http://www.eea.europa.eu/publications/COR0-landcover, Accessed 24/12/2012
19. The Repast Suite, http://repast.sourceforge.net, Accessed 24/12/2012
20. Apache Tomcat 7, http://tomcat.apache.org/tomcat-7.0-doc/setup.html, Accessed 24/12/2012
21. Koomen, E., Stillwell, J., Bakema, A., Scholten, H.J.: Modelling Land-Use Change, Progress and Applications, The GeoJournal Library 90, ISBN 978-1-4020-6484-5, Springer, 392 p. (2007)
22. Linke, Ch.S.: Local level application of the dynamic land use model METRONAMICA. Assessment and modeling – a case study on the Dutch municipality Weert. Diploma Thesis at the Technical University Berlin Institute of Landscape Architecture and Environmental Planning (2008)
23. Brennan, M., Twumasi, B.O.: Does MOLAND work as a tool for the assessment and/or as a predictor of the distribution of urban biodiversity in Dublin city? Urban Institute Ireland Working Paper Series; 08/04, University College Dublin. Urban Institute Ireland (2008)
24. JRC: MOLAND- Monitoring Land Use/Cover Dynamics. Towards Sustainable Urban and Regional Development, http://moland.jrc.ec.europa.eu/technical_tools/model/moland_model.htm, Accessed 12/02/2013

10. Aizstrauts, A., Ginters, E., Baltruks, M., & Gusev, M. (2015). Architecture for Distributed Simulation Environment. *Procedia Computer Science*, *43*, pp. 18-25. doi: 10.1016/j.procs.2014.12.004. Retrieved from https://www.sciencedirect.com/science/article/pii/S1877050914015725

Available online at www.sciencedirect.com
ScienceDirect
Procedia Computer Science 43 (2015) 18 – 25
Procedia Computer Science

ICTE in Regional Development, December 2014, Valmiera, Latvia

Architecture for Distributed Simulation Environment

Artis Aizstrauts[a]*, Eglis Ginters[a], Mikelis Baltruks[a], Marjan Gusev[b]

[a]Sociotechnical Systems Engineering Institute, Vidzeme University of Applied Sciences, Cesu street 4, Valmiera LV - 4200, Latvija
[b]Faculty of Information Sciences and Computer Engineering, University Sts Cyril and Methodius, blvd. Goce Delcev 9, 1000 Skopje, R. Macedonia

Abstract

The article describes a custom made architecture for distributed simulation. Initially intended for Skopje Bicycle Inter-modality simulation, it has proven itself as being universal, adaptable and usable for many different purposes. It supports not only the set of simulation models, GUIs and data representations, but also many users running simulations at the same time. Such architecture brings certain challenges forth. The authors discuss methods for dealing with these challenges.

Peer-review under responsibility of the Sociotechnical Systems Engineering Institute of Vidzeme University of Applied Sciences

Keywords: Distributed simulation; ECE; Repast Symphony

1. Introduction

Policy planning and decision making are among those domains that could benefit from possibilities granted by modeling of potential outcomes. But policy modeling as the object of research is versatile and complex. EC FP7 project No.287119 FUPOL „Future Policy Modelling” (http://www.fupol.eu) aims at a new approach to traditional policy modelling[1]. The simulation and visualization tools assist to governments and policy makers in the whole policy life cycle and help avoiding voluntary and wrong decisions. In FP7 FUPOL the set of policy domains as the object for analysis and simulation involves community facilities (area design, open space), urban transport and economics, land use planning, sustainable tourism and other. Each of them comprises some use cases.

* Corresponding author.
E-mail address: artis.aizstrauts@va.lv

Peer-review under responsibility of the Sociotechnical Systems Engineering Institute of Vidzeme University of Applied Sciences
doi:10.1016/j.procs.2014.12.004

Usually to simulate sociotechnical problems there are several models involved to solve particular problem and not all of them may be designed with the same simulation type. It is common to design Multi-agent system (MAS) as a distributed simulation[2]. Previously for distributed simulation communication Easy communication environment (ECE) was designed to provide communication among simulation models[3], but it cannot be used in this case as it is. ECE is more focused on direct data exchange between simulation models, but Skopje Bicycle Inter-modality simulator requires wider functionalities within the communication environment.

At first this article briefly introduces Skopje Bicycle Inter-modality but after discusses the architecture of distributed simulation environment in detail. Finally some concluding remarks are given and further research directions are defined.

2. Skopje Bicycle Inter-modality simulator use case

The main objective of this project is the development of a software solution that will offer the City of Skopje and its citizens the opportunity to simulate the occupancy and usage of bicycle stations and parking lots. The overall goal is to increase the number of people that use bicycles as transport by taking several different measures such as establishing bicycle inter-modality, initiating the development of parking lots, rent-a-bicycle facilities, new bicycle paths and improving existing bicycle paths. Such project would help the authorities of City of Skopje in improving the scheduling and resource planning, initiation and creation of new projects involving the bicycle stations and parking lots. The citizens of City of Skopje will also be involved in the decision making process by constantly communicating and expressing their opinion to the authorities, making the whole process more transparent and efficient[4]. Public simulation is accessible at http://prod.fupol.lv:8080/skopjebicycle. Skopje Bicycle Inter-modality simulator ensures validation of potential activities of Skopje Municipality using multi-agent systems simulation (ABM/MAS) in Repast Symphony[5] environment. It is based on statistical data provided by Skopje Municipality. It not only gives possibility for citizens to participate directly in policy planning and crafting, but also provides municipality with valuable feedback through ticketing[6].

3. Simulator description and requirements

This section describes the simulation types, user roles and overall functionality provided by Skopje Bicycle Inter-modality simulator. The explanations will help to understand the specific requirements of the architecture in the following section. Skopje Bicycle Routes Simulator software is created to find optimal solution for bicycle station and track building location in the City of Skopje. To encourage the intermodal transport, the citizens are also integrated in the collection of ideas on how bicycle intermodality can be fostered. The citizen participation is supported including the use of a simulation tool.

3.1. Simulation types

There are two types of simulations proposed in the simulator: Simulation 1 simulates bicycle usage for one week with given custom parameters (set of tracks, set of bicycle stations, weather conditions and month of the year). Based on these parameters simulation will generate bicycle usage for each track and station; Simulation 2 is a way of combining multiple object changes to see which changes affect the bike usage the most. Simulation 2 includes multiple project definitions, affordable project set, combination generation and best result displaying. Each combination is a mix of one or more projects added to existing simulation settings. Each Simulation 2 consists of one or more one year simulations with conditions set by projects. These types of simulation were defined by Skopje Municipality.

3.2. User roles

The user roles that were defined by customer are:

- Administrator – creates and manages users;

- Master user – setup (using import) of initial simulator configuration, change configuration, export configuration, and realize both simulation 1 and 2;
- Manager – can view ticket reports, view and answer tickets sent by users, suggest new configuration and projects that will improve the biking experience in Skopje.
- Public user – can access the visual simulation interface for Simulation 2.

These are the main requirements to the simulator. Whatsoever it should be noted that the greatest constraints are the public access to the simulator – simulation can be run many times by many users at the same time – that means, the architecture has to be flexible and able to process large amount of requests. The next section describes the distributed simulator architecture in more detail.

4. Distributed simulator architecture

The simulator is part of FP7 FUPOL project and the requirements are obtained partly from project technical guidelines. Each simulator can be viewed as one of FUPOL functionalities, i.e. end-user can use either one simulator or several. Simulator within the context of this paper is model and GUI that helps to control the model and view simulation results. The project proposes that simulation GUI is a web application, and the model based on ABM/MAS technologies. At the same time the models have to be executable also by third party software that is developed within other work packages of FP7 FUPOL. That means the architecture has to be able to run all models together, and each model might have different GUI. Based on previous research is it known that there must be a central element for such an architecture that is responsible for the data exchange between architecture components, such as Simulator GUI and model[7,8]. Simulation GUI and Model can be considered as plugins within the architecture. The architecture has to ensure that these components can be changed dynamically (plugged or unplugged) without interrupting the operation and without interfering with other components. At the same time it has to be scalable and ensure that its components can be distributed and operate within a cloud. A component (for example, a model) can also exist in several instances, to maximize the amount of simulation sessions within one unit of time. Another important issue to be considered is the fact that simulators can be publicly available. That means that the architecture has to be able to process large amount of requests generated by large amount of users. In this case the amount of users exceeds the number of models (model instances) and it is technically impossible to run all the requested simulation instantly with existing model instances. The architecture has to be able to handle this issue by introducing some waiting queue for requested simulation. From the viewpoint of system integrity the architecture has to ensure mechanisms to be able to operate also in the situations when one of elements is not working properly due to different problems. For example when the connection with the database is interrupted and models have nowhere to store the data. The architecture has to embody mechanisms to ensure that these data are not lost.

Taking into account everything mentioned previously, it was decided to construct the architecture according to Fig.1. The architecture can be divided into logical elements that communicate with each other and some of the parts can be distributed over the cloud technologies.

The further sections describe the elements and operations of the architecture shown in Fig.1.

4.1. Simulation GUI

Simulation GUI is a tool or platform that ensures user interaction with model, lets them define configuration of the simulation, execute it and view the results. The architecture is designed so that the simulation GUI could operate as an independent element, that has to be able to communicate with the architecture using Advanced Message Queuing Protocol (AMQP)[9]. From the viewpoint of architecture, there is no difference, how the simulation GUI is organized, what programming language is used, what additional functionalities it has etc. However it is important that GUI is able to communicate with Message broker/Message queue. So far the FP7 FUPOL experience shows that different additional options are usually annexed to the simulation. For example, user management (including authorisation), simulation reports, advanced analytics and ticketing functionality (public participation). Fig. 2 shows a single view from Skopje Bicycle Inter-modality simulator.

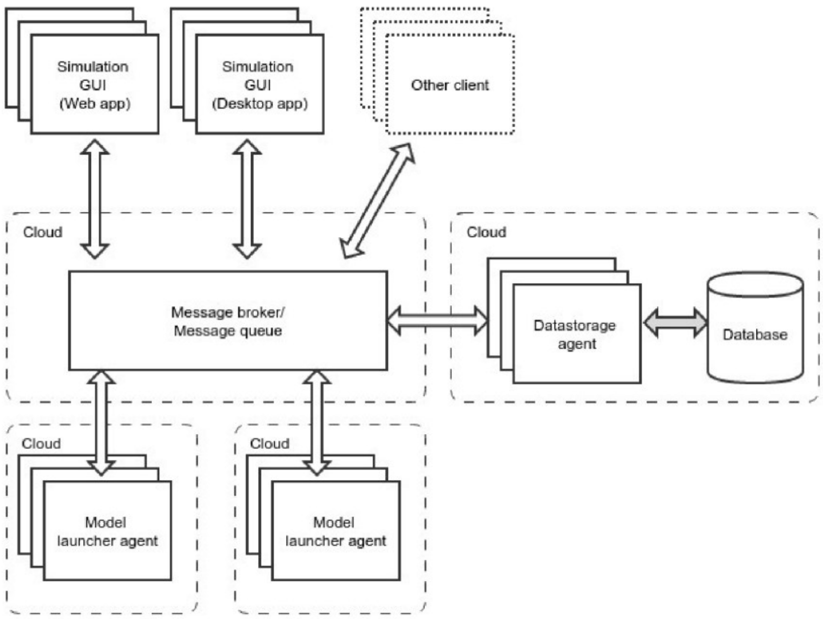


Fig. 1. Distributed simulation architecture.

The architecture is designed in a way that allows the Skopje Bicycle Inter-modality simulator GUI (see Fig. 2) to be fully customizable. Thus GUI is adjustable to specific cases and groups of users and their needs. For example (see Fig. 2), different user-friendly features as interactive maps sometimes are helpful for public use, but not always. Sometimes it is much more important that GUI is orientated towards field specialists that need to be able to fully control simulation and do not need decorative appearance or moving objects.

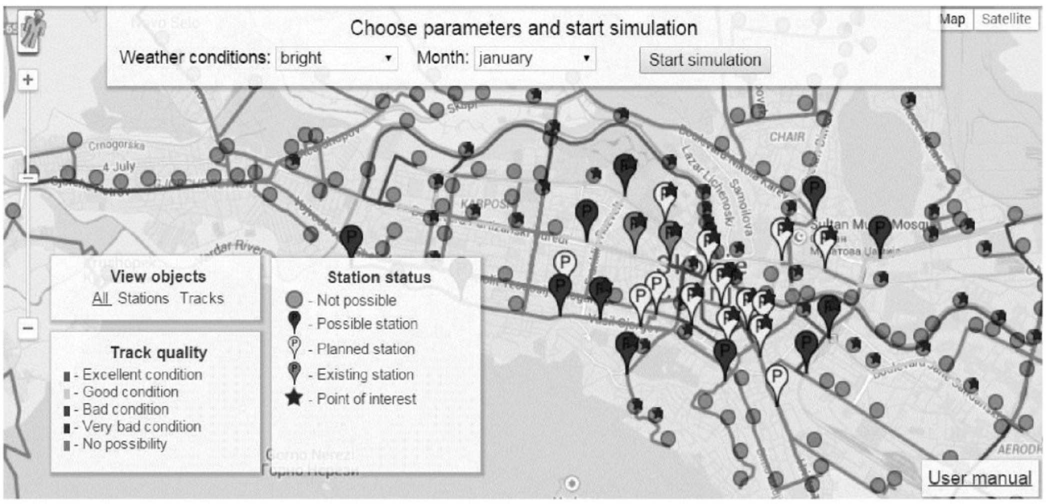


Fig. 2. Skopje Bicycle Inter-modality simulator GUI.

It is worth mentioning that the architecture has to be able to support both mentioned solutions – they are equal from the point of view of the architecture. They both send specific commands to start a simulation with specific parameters. This architecture is built to ensure only the infrastructure for designing of simulator (simulation GUI + simulation model). In other words, the developers of simulator have to define the communication themselves. The architecture has to ensure only two main functions - the delivery of message from simulation GUI to simulation model and also the storage (and retrieval) of messages in database (see Fig.1). The content of the message that is sent from simulation GUI to run the simulation will be explained further.

4.2. Message broker/Message queue

Message broker/Message queue (MQ) ensures all communication within the architecture. It also ensures that simulation GUI is able to deliver message to Model launcher agent to start respective model. And the Model launcher forwards the simulation parameters to the model. MQ may be used also for any custom communication among the architecture components if needed. The message broker is a message-oriented middleware (MOM) - software or hardware infrastructure supporting sending and receiving messages between distributed systems. MOM allows application modules to be distributed over heterogeneous platforms and reduces the complexity of developing applications that span multiple operating systems and network protocols[10]. In fact it would also be possible to use any other task/job queuing tools, but the Message Broker was chosen because it is able to ensure equivalent functionality as task/job queuing tools and is more orientated to message delivery, that is important in this case, because simulation results are sent to database also using MQ.

Before integration into architecture, several MQ were analysed. The summary of conclusions and the mayor differences can be seen in Table 1.

Table 1. Message broker solutions.

MQ	Open source or similar license	Administration/ monitoring panel	Self-hosted	Client language support	Previous experience
ActiveMQ[11]	Yes	Yes	Yes	Many[a]	Yes
Kafaka[12]	Yes	Yes	Yes	Many[b]	No
RabbitMQ[13]	Yes	Yes	Yes	Many[c]	Yes
Amazon SQS[14]	No	Yes	No	Java, .Net, Node.js, PHP, Python, Ruby	Yes

Within FP7 FUPOL project open-source products are considered first. If it is not possible to find an open source product, then the commercial counterparts are considered. At the same time in this case it was important that MQ is self-hosted, and does not use open source cloud platform. Thus Amazon SQS was not further considered.

For the system to be easily maintained it is important that there is a convenient monitoring tool that enables the management of communication – helps understand which nodes are alive/dead, monitor bottlenecks, analyse throughput, etc. Therefore Administration/monitoring panel is used as a criterion and all MQs included in the analysis do have this panel. It is important that MQ has ready libraries for as many programming languages as possible. That allows the expansion of the use of architecture and there are fewer constraints to simulation GUI. Mostly all MQ tools included in the analysis support many programming languages except Amazon SQS.

---

[a] ActionScript, Ajax, C, C++, C#, .Net, Delphi, Erlang, Flash / ActionScript, Haskell, Perl, PHP, Pike, Python, Ruby

[b] Python, Go (AKA golang), C, C++, .net, Clojure, Ruby, Node.js, Storm, Scala DSL , HTTP REST, JRuby, Perl, Java, PHP, Python, Go, Erlang, etc.

[c] Java, Ruby, Python, .NET, PHP, Perl, C / C++, Erlang, Node.js, Go, Common Lisp, Haskell, Ocaml, COBOL, etc.

Taking into account all above mentioned arguments, only RabbitMq and ActiveMQ were analysed in more detail. A throughput test was performed to make the final decision. The test was carried out with out-of-box configuration for each tool (both had persistent mode on). The parameters of the experiment were following:

- Size of message – 5KB;
- One message receiver;
- 1, 5 and 10 message senders;
- Computer - Intel® Core™ i5-3320M CPU @ 2.60GHz × 4, 11.4 GB RAM.

Table 2. MQ performance test results

MQ	1 sender	5 senders	10 senders
ActiveMQ	10000 msg/s	2000 msg/s	1800 msg/s
RabbitMQ	30000 msg/s	6500 msg/s	3500 msg/s

Based on performance test results (see Table 2) it was decided to use RabbitMQ as message broker for simulation environment architecture designing under the framework of FUPOL project.

4.3. Model launcher

Model launcher is custom made software that launches Repast Symphony models. The command to launch the model is received from MQ. To launch the simulation Simulation GUI has to send a message to MQ with a reference that this message has to be included in queue and named "new simulation". Respectively every Model launcher one by one receives messages from this queue and each simulation is designated to different Model launcher. This kind of architecture design allows to dynamically increasing the number of Model launchers, if it is necessary (in case queue "new-simulation" is starting to get crowded). Fig. 3 shows the elements that are involved in launching a simulation and storing the data/results of it in the database. As previously mentioned MQ contains a queue that is named "new-simulations" and "stores" simulation launch requests. To store the data in database Model Launcher sends data to queue named „save-simulation-data".

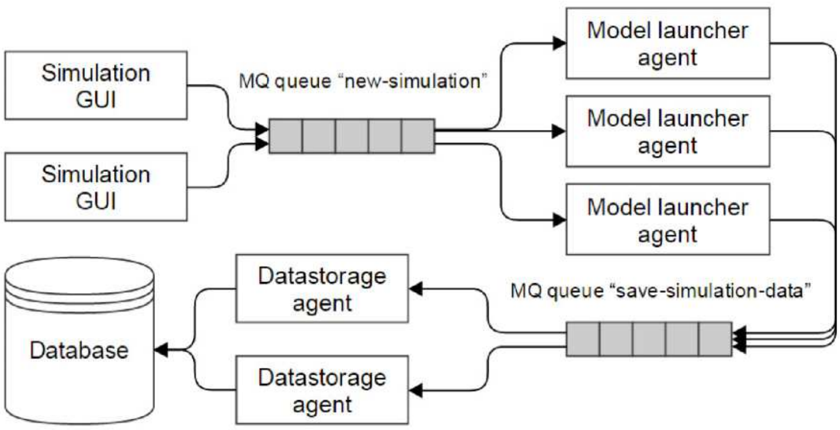


Fig. 3. Start simulation message flowchart.

Simulation GUI inserts message in MQ queue "new-simulation". The message is in JSON (JavaScript Object Notation)[15] format.

The Model launcher determines which model has to be launched according to message attribute “simulation”. The data that can be found in the message attribute “params” are delivered (by Model launcher) to the model before the launch of it. Within this architecture the developers of Simulation GUI and the developers of the model have to decide what data the attribute “params” will contain. It is a closed agreement as the architecture does not provide any mechanism for third parties to find out what parameters are needed for each model. Such mechanisms are provided similarly by HLA with Federation Object Model[9].

4.4. Datastorage agent and Database

Architecture allows data storing in the database. The main function of database is to accumulate simulation results created by model. Datastorage agent’s task is to read these data from MQ queue “save-simulation-data” and store them in database. The architecture anticipates that database vendor has no crucial role – with datastorage agent there are many possibilities to create different database adapters, thus widening the scope of architecture realizations. For now, adapters that support PostgreSQL and MongoDB have been created. Message broker has two queues that are checked by Datastorage agent on a regular basis:

- "Save-simulation-data" – in this queue those JSON format messages are inserted, that consist of data that has to be stored in the database;
- "Get-simulation-data" – in this queue those JSON format messages that include inquiry for simulation data. Datastorage agent understands inquiries with AND and OR comparisons.

Fig. 4 shows all steps, how data is retrieved from database. In this case it shows that Simulation GUI creates data request in QM queue "get-simulation-data” that is then read by Datastorage agent. The message is being parsed and then processed within database. When Datastorage agent has obtained all necessary data, the results are formatted in JSON and returned to Simulation GUI, using MQ queue "reply queue".

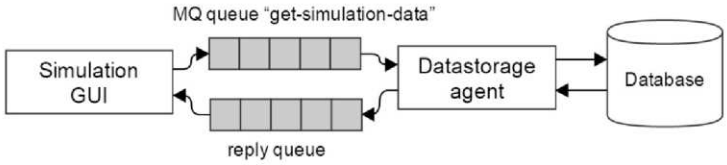


Fig. 4. Fetch data from database.

The architecture provides also the possibility to run many Datastorage agent nodes, thus improving performance.

5. Conclusion

This article describes distributed simulation modelling architecture that was developed for the purposes of FP7 FUPOL project. The requirements defined by project posed certain challenges for the developing of such architecture.

Easy communication environment (ECE) is designed to provide communication among simulation models. Although ECE has some great design ideas and features, it proved to be too tightly bounded for direct simulation model communication. In this case ECE could not be used as it is and for this project a new architecture had to be designed, taking into account previous gained experience with communication in distributed simulation environments[8, 2]. This architecture had to balance between two integration aspects – being easy and universal. The architecture had to be flexible to implement various simulation models, regardless of simulation models and software (AnyLogic, RePast, etc.). At the same time the architecture had to be simple to implement for developer of simulators.

The authors describe the architecture that correspond the various requirements defined by stakeholders. The main elements of the architecture are – Simulation GUI, Message broker/Message queue, Model launcher and Datastorage agent and database.

Further research and developments are needed to make the input parameter structure and data structure generated by model available publicly within the architecture. Another perspective for further work is the Message broker attachment to each Model launcher. This would ensure that data will not be lost in cases when connection with main Message broker is failed.

References

1. FUPOL project homepage. Retrieved: 05.10.2014, URL: http://www.fupol.eu/en.
2. Silins, A., Ginters, E., Aizstrauta. D. Easy Communication Environment for Distributed Simulation. In World Scientific Proceedings Series on Computer Engineering and Information Science 3. "Computational Intelligence in Business and Economics" Proceedings of the MS'10 International Conference. ISBN 978-981-4324-43-4. 15-17 July, 2010. Barcelona, Spain. pp.91-98.
3. Aizstrauts, A., Ginters, E., Aizstrauta, D. Step by step to Easy Communication Environment for Distributed Simulation. In: Annual Proceedings of Vidzeme University of Applied Sciences „ICTE in Regional Development 2009/2010". ISBN 978-9984-633-20-6. 2011. Valmiera, Latvia. pp. 1-13.
4. Ginters, E., Aizstrauts, A., Dreija, G., Ablazevica, M., Stepucev, S., Sakne, I., Baltruks, M., Piera, M-A., Buil, R., Gusev, M., Velkoski, G., 2014. Deliverable 4.6 – Intermediate Release FUPOL Simulator Software advanced prototype. Skopje Bicycle Inter-modality Simulator. 224 p.
5. Repast Simphony, Retrieved: 05.10.2014, http://repast.sourceforge.net/repast_simphony.php
6. Ginters, E., Aizstrauts, A., Dreija, G., Ablazevica, M., Stepucev, S., Sakne, I., Baltruks, M., Piera, M-A., Buil, R., Gusev, M., Velkoski, G. Skopje bicycle inter-modality simulator – E-involvement through simulation and ticketing. Proceedings of the European Modeling and Simulation Symposium. 2014. 978-88-97999-38-6. pp 557-262
7. Aizstrauts, A., Ginters, E., Aizstrauta, D., Sonntagbauer P. Easy Communication Environment on the Cloud as Distributed Simulation Infrastructure. In: Proceedings of the 5th WSEAS World Congress on Applied Computing Conference (ACC '12). Recent Advances in Computing Engineering Series 2. ISBN: 978-1-61804-089-3. ISSN: ISSN: 1790-5109. 2-4 May, 2012. Faro, Portugal. pp. 173-179.
8. Ginters, E., Silins, A., Andrusaitis, J. "Communication in distributed simulation environment". In Proceedings of 6th WSEAS International Conference on Systems Science and Simulation in Engineering. ISBN 978-960-6766-14-5. 21-23 November, 2007. Venice, Italy. pp. 217-221.
9. Advanced Message Queuing Protocol - Wikipedia, the free encyclopedia, Retrieved: 05.10.2014 http://en.wikipedia.org/wiki/Advanced_Message_Queuing_Protocol
10. Message-oriented middleware - Wikipedia, the free encyclopedia, Retrieved: 05.10.2014, http://en.wikipedia.org/wiki/Message-oriented_middleware.
11. Apache ActiveMQ, Retrieved: 01.10.2014, http://activemq.apache.org/
12. Apache Kafka, , Retrieved: 01.10.2014, http://kafka.apache.org/
13. RabbitMQ - Messaging that just works, Retrieved: 01.10.2014, http://www.rabbitmq.com/
14. Amazon Simple Queue Service, Retrieved: 01.10.2014, http://aws.amazon.com/sqs/
15. JSON homepage, Retrieved: 05.10.2014 , http://www.json.org/

11. Ginters, E., Aizstrauts, A., Baltruks, M., Merkuryev, Y., Novickis, L., Grundspenkis, J., & Grabis, J. (2016). VeloRouter - Technology for Urban Transport Intermodal Sustainability. In *City Planning and Urban Design Conference, 07-09 April (CPUD'16)* (pp. 211-220). Istanbul: DAKAM. ISBN 978-605-9207-21-8. Izgūts no https://doi.org/10.5281/zenodo.13925061

INTERNATIONAL CITY PLANNING AND URBAN DESIGN CONFERENCE

CPUD '16

APRIL 8-9, 2016, NIPPON HOTEL MEETING HALLS, TAKSIM, ISTANBUL

BILSAS

DAKAM

CPUD '16 / International City Planning and Urban Design Conference Proceedings

April 8-9, 2016, İstanbul

organized by DAKAM (Eastern Mediterranean Academic Research Center)

Edited by: Hande Tulum

Reviewed by: Barış Öztürk

Cover Design: D/GD (DAKAM Graphic Design)

Print: Metin Copy Plus

Mollafenari Mah., Türkocağı Cad. 3/1, Mahmutpaşa/Istanbul, Turkey

ISBN: 978-605-9207-21-8

VELOROUTER - TECHNOLOGY FOR URBAN TRANSPORT INTERMODAL SUSTAINABILITY

EGILS GINTERS, ARTIS AIZSTRAUTS, MIKELIS BALTRUKS, YURI MERKURYEV, LEONIDS NOVICKIS, JANIS GRUNDSPENKIS, JANIS GRABIS

Egils Ginters, Artis Aizstrauts, Mikelis Baltruks, *Sociotechnical Systems Engineering Institute, Vidzeme University of Applied Sciences, 4 Cesu Street, Valmiera LV-4201, Latvia, e-mail: egils.ginters@va.lv,* ***Yuri Merkuryev, Leonids Novickis, Janis Grundspenkis, Janis Grabis,*** *Riga Technical University, Faculty of Computer Science and Information Technology, 1 Kalku Street, Riga, LV-1658, Latvia, e-mail: merkur@itl.rtu.lv*

Abstract

The European Commission Decision C (2015) 6776 established the specific European Transport Specific Programme implementing Horizon 2020, Part 11 "Smart, green and integrated transport", which promotes using clean transport methods, one of which is cycling. In order to ensure the successful functioning of an integrated urban transport system cycling has to be one of its components, which naturally fits into the overall intermodal transport system.

Due to limited funding it is important to understand which bike path network would be more efficient and ensure the greatest possible population transfer from other types of transport to bicycles.

The sustainability of an urban environment depends on quality of planning and correct crafting, which is significantly influenced by the technologies used to rule out acceptance of voluntary and intuitive decisions. One of recommendable solutions is simulation. The multi agent-based (MAS/ABM) bicycle path network and exploitation simulator (VeloRouter) has dual applicability as it is adapted to both the needs of municipalities and cyclists.

Introduction

Technologies are used to rule out acceptance of voluntary and intuitive decisions influencing the sustainability of an urban environment. One usable method is simulation when a computer imitates a process or phenomenon as if it was happening in real life. The basis of the urban environment multi-level model is a multi-agent Land-Use Category Change (LUCC) simulation model, which allows forecasting changes in urban zoning for a set period of time (Lemp, et. al, 2008; Ravulaparthy and Goulias, 2011). Each zone changes size is based on an empirical simulation algorithm. This determines the activities and the extent to which they can be performed in a specific zone, such as industrial or agricultural manufacturing, public recreation activities, individual or multi-storey buildings, hotels, green areas, etc., which in turn affects the placement of infrastructure, such as water and sanitation, electricity, and transport networks.

There are different LUCC implementation sets, the oldest of which are realised using cell automatons simulation technologies, for example Metronamica (Linke, 2008; Kim and Batty, 2011; Riks, 2015), SLEUTH (Clarke et. al, 2007) and other LandUse Scanner (Koomen, et. al, 2011), WhatIf? (Klosterman, 2007), which determine several development limitations. The younger LUCC solutions, such as Fearlus (Macaulay Land Use, 2015), UrbanSim (Waddell, 2011), FUPOL LUCC (Deliverable 4.3, 2013) (see Figure 1) and others are usually designed using MAS/ABM technologies, which ensure tool compatibility and data exchange capabilities in a heterogeneous and distributed multi-level simulation model (Aizstrauts, et.al, 2014).

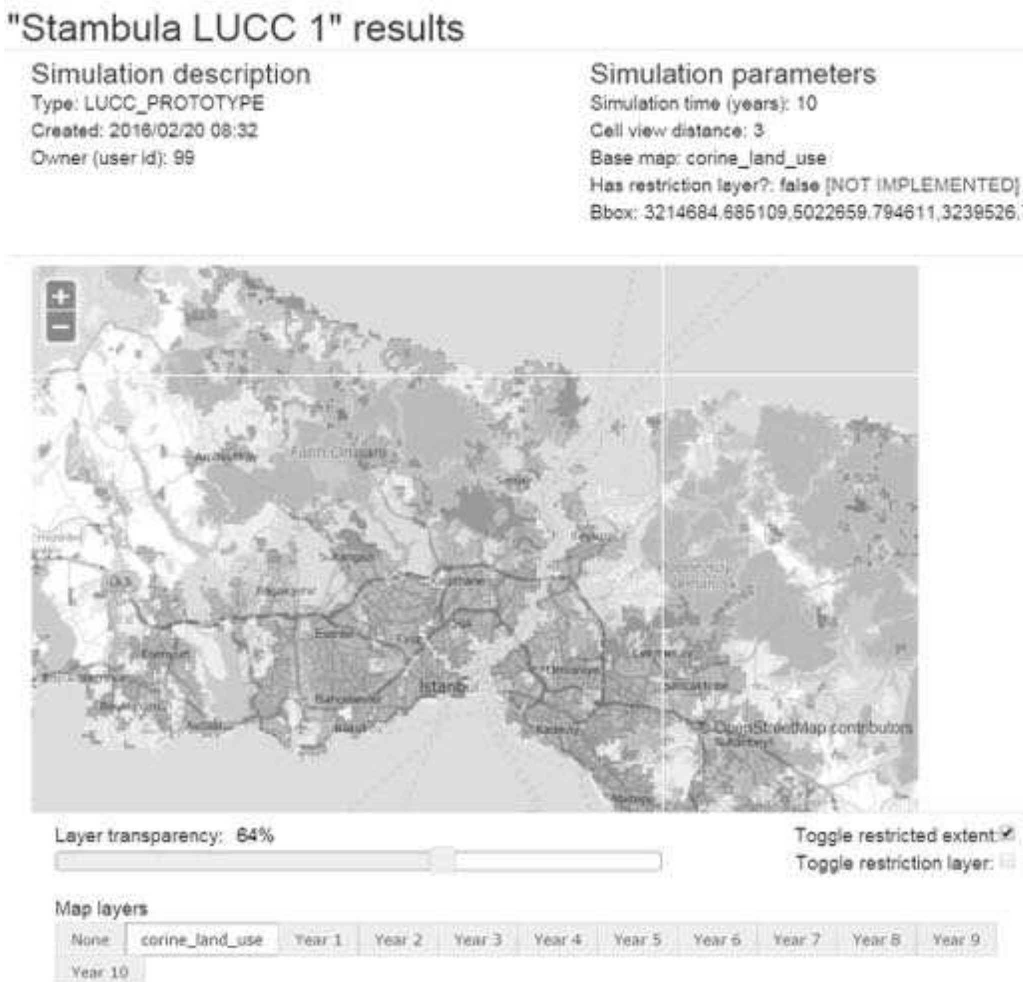


Figure 1. FUPOL Land-Use Category Changes simulator

In this case LUCC is the basement of the urban sustainability model, as it determines the functionality and limitations of any further models. The use of LUCC is the only way in the bottom-up simulation process. Otherwise designers can expect activity planning incompatibilities because there are no land, roads and inhabitants. Only by knowing the projected land area for each application, it is possible to create a reasonable design (see Figure 2).
Based on LUCC simulation results on the higher urban environment design levels it is possible to simulate changes in city demographics and potential tax income and expenses, as well as perform other urban policy crafting simulation activities. A sustainable urban transport system can be one goal of the model.
The European Commission Decision C 6776 established the specific European Transport Specific Programme implementing Horizon 2020, Part 11 "Smart, green and integrated transport" (EC, 2015), which promotes using clean transport methods, one of which is cycling. In order to ensure the successful functioning of an integrated urban transport system, cycling has to be one of its components, which naturally fits into the overall intermodal transport system. This means that bike paths, racks and the lease network must correspond to the transport system. Unfortunately, the lack of financial resources makes it impossible to construct bike paths wherever it would be desirable.

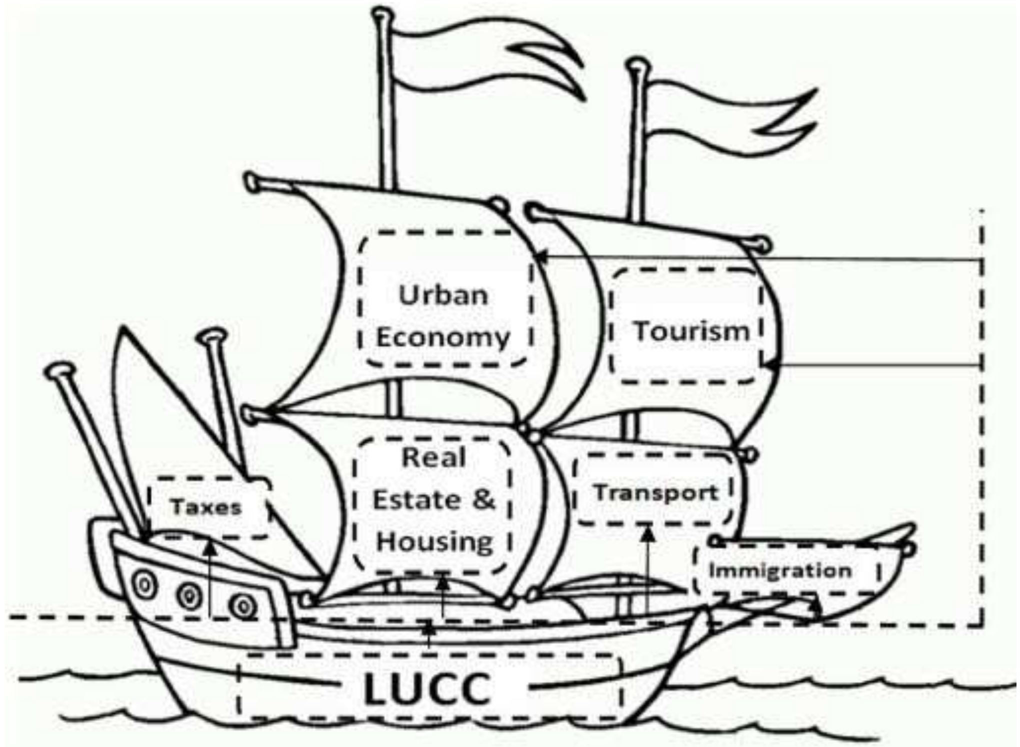


Figure 2. Multi-level policy crafting simulation model

Therefore, it is important to understand which bike path network would be more efficient and ensure the greatest possible population transfer from other types of transport to bicycles.

Cycling routes planning and design

The idea for VeloRouter originated in two EC FP7 projects FP7-ICT-2009-5 CHOReOS No. 257178 (2010-2013) "Large Scale Choreographies for the Future Internet" (Lescevica, et. al, 2013) and FP7-ICT-2011-7 FUPOL No. 287119 (2011-2015) "Future Policy Modelling" (Piera, et. al, 2013). This gave the opportunity to perform market research and understand the needs of potential users.

Cycling route database **Bikemap** (http://www.bikemap.net) is a huge collection of bike routes published by service users. Amongst others there are routes of weekend tours, longer holiday trips or the way to work. These routes can be joyful rides for tourists, single trails for mountain bikers or really long routes crossing whole continents, like the Eurovelo 7. Bikemap is free of charge. Bikemap works without registration. The elevation data has up to 30m (98 ft) deviance, therefore the distance markers and the elevation profile are not 100% accurate. Currently, there are 1,630,303 bike routes with a total length of 154,550,457 km. The user can mark his route on a map and publish it, as well as see its length and elevation profiles.

Similar functionality is offered by **veloroutes.org** (http://veloroutes.org), **routebuilder.org** (http://www.routebuilder.org) and **EuroVelo** (http://www.eurovelo.com/en). EuroVelo is a network of **14 long distance** cycle routes connecting and uniting the whole European continent. The routes can be used by **cycle tourists as well as by local people making daily journeys**. EuroVelo is currently comprised of 14 routes and it is envisioned that the network will be substantially complete by 2020. Similar products worth mentioning are **Pro-Plan**, **Topo-Plan** un **Bike Node Plan** (http://www.routeyou.com/route/routeplanner/overview_all/choose-the-route-planner-that-fits-your-needs), **MapMyRide** (http://www.mapmyride.com) and **CycleStreets** (http://www.cyclestreets.net/about/), which is a UK-wide cycle journey planner system that allows to plan bike routes from start to finish. It is designed by cyclists, for cyclists, and caters for the needs of both confident and less confident cyclists. A similar tool is ***plotaroute.com*** (http://www.plotaroute.com) that is a free worldwide online route planner for walking, running and cycling. It provides people with a simple way to accurately plan, measure and share routes used for leisure purposes. The website was launched by Spidersphere (UK) in January 2014 and is now used by thousands of people across the world every month.

The bike route planner **Flattest Route** (http://www.flattestroute.com) in addition to elevation profiles also shows route complexity, which is important for Sunday cyclists. **Velomap.org** (https://www.velomap.org) allows choosing not only the shortest but also the most user-friendly route. **RouteLoops** (http://www.routeloops.com) is the site that creates free, custom routes for running and biking that begin and end at the same location. There are several specialised route planners, for example for mountain biking **BikeRouteToaster.com** (http://www.bikeroutetoaster.com), which is a course creation application primarily aimed at Garmin Edge/Forerunner owners although other users without a GPS may also find it useful for planning rides.
Another interesting tool is the bicycle route planner **Pro Velo Bern** (http://www.veloroutenplaner.ch) and the city of Bern which is based on OpenStreetMap data. The proposed routes are generated automatically based on this data. The quality and density of the OpenStreetMap data varies by region. The routing can be used all over Switzerland. The operators of the bicycle route planner strive to keep the bicycle-relevant data in the city and agglomeration Bern up to date. Nevertheless, it may happen that the suggested route can’t be driven all the way or that cycling is banned on certain sections. Short passages through banned streets or footpaths may be proposed. Decisive is the signalization in the streets and the road traffic laws. The operators of the bicycle route planner assume no liability for the suggested routes. Regrettably, it does not include a simulation section, which would provide information about the possible load of the chosen route, as well as geofencing capabilities.
Skopje Bicycle Inter-Modality Simulator (EC FP7 FUPOL) (http://www.fupol.eu/en/news/skopje-bicycle-inter-modality-simulator) (see Figure 3) (Buil, et. al, 2015) is created to find a useable solution for bicycle station locations in the City of Skopje. To encourage intermodal transport, the citizens are also involved in the collection of ideas on how bicycle inter-modality can be fostered using a ticketing mechanism. The system helps the Administration of City of Skopje to improve the scheduling and resource planning, initiation and creating new projects involving the bicycle area in Skopje city. The above-mentioned simulator can be considered analogous to VeloRouter, however solution commenting and statistics aggregation capabilities here are limited. Simulation algorithms and results are focused on the needs of a specific large municipality and does not entail a detailed analysis of individual cyclists. Adaptation of the product is difficult and it is intended for use in large cities, which significantly limits the potential user base.

Based on the analysis above, it can be concluded that cycling route design and planning products mainly offer the capability of publishing cycling routes, but do not provide functionality necessary for municipalities to design a bicycle path network, which is critical for successful sustainability management.

Figure 3. Skopje intermodal bicycle simulator

The technologies are not designed to be deployed on the Future Internet (Software as a Service) (SaaS) environment, because the architecture of these tools does not conform to Service Oriented Architecture (SoA) requirements and is not based on web-service solutions.

VeloRouter - Dual use bicycle path network designing and exploitation simulator

The multi agent-based bicycle path network and exploitation simulator (VeloRouter) is designed in the Repast Symphony environment and uses OpenStreetMap spatial data. The product has dual applicability as it is adapted to both the needs of municipalities and cyclists. Each agent is a cyclist or a group of cyclists that move on a chosen route considering route load, traffic restrictions and the quality of the route.

VeloRouter provides municipalities with bicycle path discussion and geofencing opportunities by receiving feedback from cyclists directly and by using semantic search tools on social networks.

VeloRouter user authentication and authorisation is ensured using social networks: Facebook, Linked-in, Twitter, ResearchGate etc., given that cyclists are avid users of social technologies, whereas municipalities have the option to use individual registration options.

VeloRouter has two blocks: municipality un cyclist. The municipality is interested in some basic question: Is the offered cycling route map satisfactory? This is determined by summarizing potential comments during project deliberations. The second question is: Which potential cycling route sections should be built first? (see Figure 4).

The cyclists want to know what the load of a route will be in certain meteorological conditions on a specific date, as well as if the route is suitable for the group i.e. terrain etc. (see Figure 5)? Load MAS/ABM simulation provides monitoring of bicycle path network development scenarios and change management capabilities. Busy route sections are highlighted.

Figure 4. Routes map discussion view

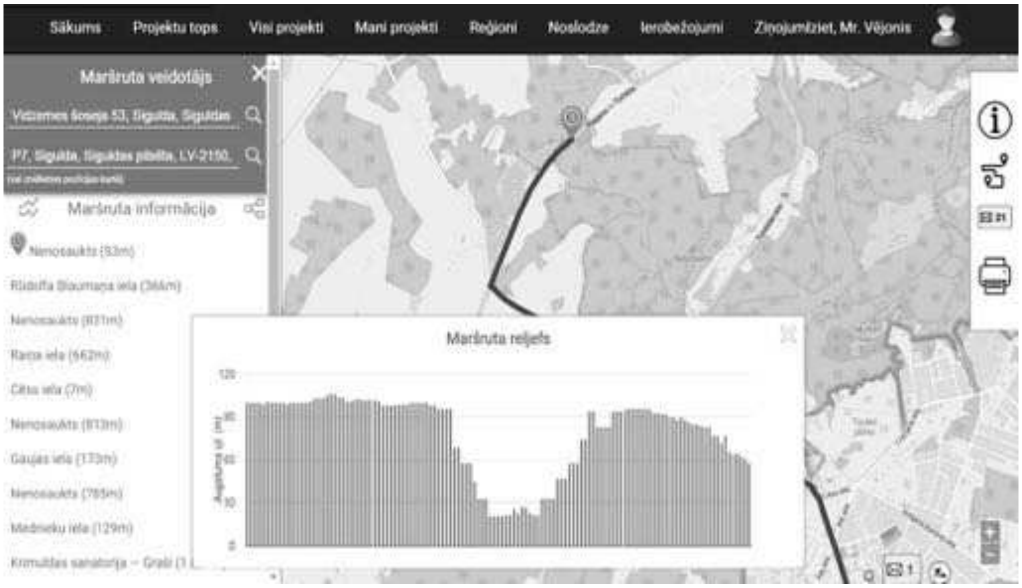


Figure 5. VeloRouter route design view

The cyclist has the opportunity not only to send a message to the municipality, but also to publish his routes for public discussion.
During VeloRouter usage a PostgreSQL database is continually and automatically updated. With each usage this improves the quality of simulated load predictions. The database contains data entered by the municipality, which was gathered during interviews in public events, supermarkets etc., geofencing data, which was added by cyclists and other interested parties, as well as the planned routes of cyclists. Content is complemented by crowdsourcing semantic search data from social networks.
VeloRouter is an interdisciplinary science product that simultaneously uses knowledge management and semantic search methods, geofencing and crowdsourcing, advanced data visualization capabilities, as well as web-service and multi-agent-based simulation (MAS/ABM) approaches, providing further software deployment on the Future Internet environment.

Multi-agent based load simulation
It is preferable to pay attention to the MAS/ABM load simulation algorithm.

Initial data:
- Number of cyclists;
- Type of day (Working day, holiday, all days);
- Start time (HH: mm interval);
- Weather probability (summer: April – September, winter: October - March),
  - Summer day with precipitation (No, not likely, likely, yes),
  - Summer day without precipitation (No, not likely, likely, yes),
  - Winter day with precipitation (No, not likely, likely, yes),
  - Winter day without precipitation (No, not likely, likely, yes);
- Route (Start and end points);
- Weather conditions (Specific date and hour),
  - Temperature (C),
  - Snow (mm),
  - Rain (mm),
  - Atmospheric pressure (hPa),
  - Wind,
    - Speed (m/s),
    - Direction (meteorological degrees).

Output data:

- ▪ Route load per minute (data structure),
- o Route section (ID),
- o Day,
- o Minute,
- o Load;
- ▪ Route section load in total over days(total) (data structure),
- o Route section (ID),
- o Day,
- o Load.

Simulation algorithm:
The basis is Occupancy data, which is entered from the municipality and exploitation views ("Load data" from the input data list). Occupancy data contains the number of cyclists, which is considered the smallest entity. The entity represents a specific cyclist who uses the roads. For each such cyclist an agent has to be created, which, based on other attributes and input data, will participate in traffic at a specific time and day. Figure 6 shows the general calculation algorithm from the Occupancy data point of view.

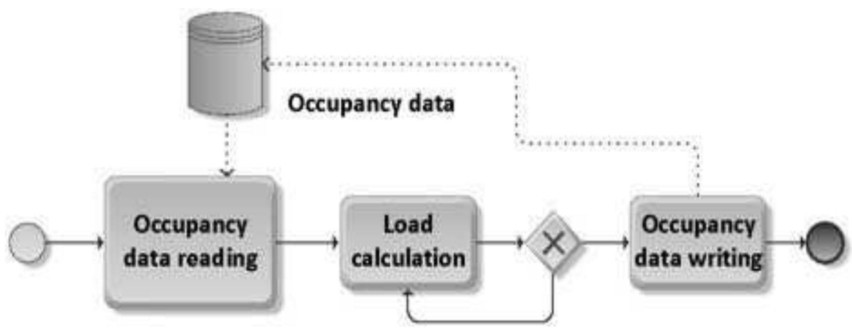


Figure 6. Load simulation general model

The general load algorithm (see Figure 6) shows that data containing all Occupancy entries is read from a database and they are all consecutively processed (subprocess "Load calculation"). When all Occupancy data entries are processed, load data is saved to the database. Figure 7 contains a more detailed look on the "Load calculation" sub process. To estimate the load of each route, initially the attributes of each cyclist are calculated (subprocess "Cyclist parameters calculation") based on model input data.

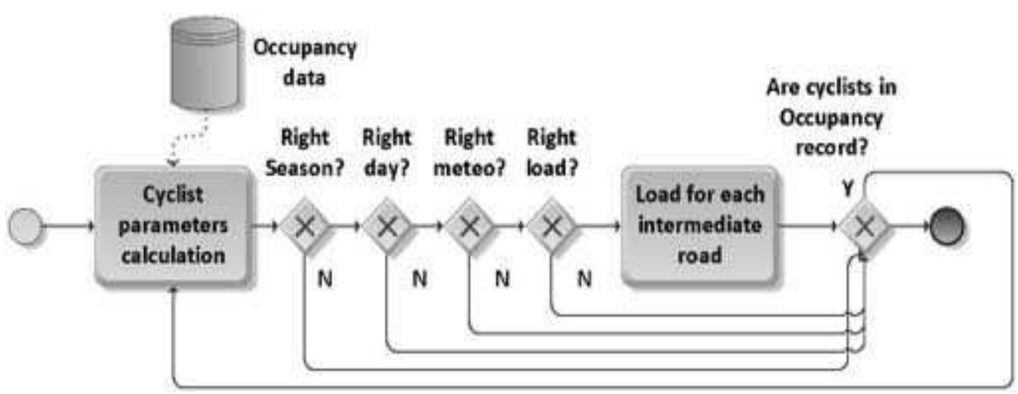


Figure 7. "Load calculation" subprocess expansion

When cyclist attributes are calculated, it is determined whether he will participate in traffic. The following conditions are checked:

- Is it an appropriate season to be cycling?
- Is it an appropriate day (working day / holiday) for the specific cyclist to be cycling?
- Is the weather appropriate for the specific cyclist to be cycling?
- Is road load adequate for the specific cyclist to be cycling?

If all these conditions are met, the cyclist participates in traffic and occupies route sections during a particular time (specific hour and minute) based on the calculated attributes. Figure 8 shows the "Cyclist parameters calculation" subprocess in more detail.

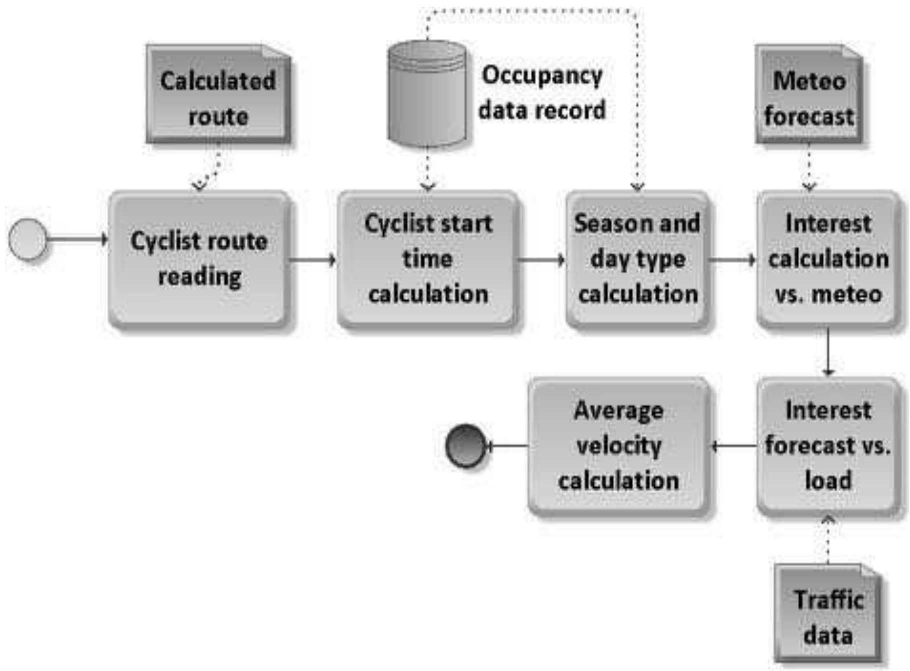


Figure 8. "Cyclist parameters calculation" subprocess content

The following attributes are calculated for each cyclist:

- Route for the particular cyclist (the route calculation algorithm takes into account road restrictions);
- Cycling start time;
- Desire to cycle on a specific date (summer/winter, type of day);
- Desire to cycle in specific weather conditions (data import from the Weather Forecast service based on date and time of day);
- Desire to cycle taking into account route load;
- Average speed.

The cycling route is calculated for each Occupancy entry during each load simulation session. If one Occupancy entry consists of multiple cyclists, all take one route, but the time can vary. The route is recalculated each time to be up-to-date based on the version of each route calculation algorithm, as for example road restrictions can change on a daily basis.

The cycling route is calculated for each Occupancy entry during each load simulation session. If one Occupancy entry consists of multiple cyclists, all take one route, but the time can vary. The route is recalculated each time to be up-to-date based on the version of each route calculation algorithm, as for example road restrictions can change on a daily basis.

Weather data is retrieved from the WeatherForecast service, which sends forecast information for the day by the hour, therefore if cycling takes longer than an hour it is possible to simulate whether the trip has to be interrupted. The MAS/ABM model verification does not rise problems, however validation depends on the capacity of data base therefore it can be done later stages of the project.

Conclusions

Simulation enables users to evaluate and predict possible outcomes to avoid voluntary decisions.

VeloRouter is one of the first simulators that takes into account municipality tasks, performs a social function and is cyclist-friendly.

One of the most important preconditions for a precise forecast is qualitative and credible data selection, which is usually realised using interviews and surveys, however survey confidence especially in sensitive segments is low. Higher impartiality can be ensured using semantic search and data analysis possibilities in social networks.

Informative feedback is a precondition for every successful governance authority. VeloRouter enables this by offering commenting and opinion aggregation capabilities.

Basically all cycling route designers offer route planning capabilities, however there are very few that propose route load predictions for a route on a specific date taking into account meteorological conditions.

Agent-based simulation supports relatively precise load prediction so unpleasant incidents during a trip can be limited.

VeloRouter enables continuous prediction update capabilities, which are determined by the amount and quality of data in the database.

The software is based on open-source principles, therefore not only promoting dissemination of VeloRouter, but also ensuring clean code and user security.

The main tasks for further development are the use of semantic analytics and displaying results in a manner that corresponds to the user's perception, as well as their virtualisation and VeloRouter integration into a unified municipality information system.

References

Aizstrauts, A., Ginters, E., Baltruks, M., Gusev, M., 2014. Architecture for Distributed Simulation Environment. ICTE in Regional Development, December 2014, Valmiera, Latvia. Ginters, E., Schumann, M. (eds.). *Procedia Computer Science*, Elsevier, ISSN: 1877-0509, Volume 43, doi:10.1016/S1877-0509(15)00231-8, pp. 18-26.

Buil, R., Piera, M.A., Gusev, M., Ginters, E., 2015. MAS Simulation for Decision Making in Urban Policy Design: Bicycle Infrastructure. In *Proceedings of 17th International Conference of Harbour, Maritime & Multimodal Logistics Modelling and Simulation* (HMS 2015), 21-23 September, Bergeggi, Italy.

Ravulaparthy, S., Goulias, K.G., 2011. Forecasting with Dynamic Microsimulation: Design, Implementation, and Demonstration. Final Report on Review, Model Guidelines, and a Pilot Test for a Santa Barbara County Application University of California Transportation Center (UCTC) Research Project, University of California Transportation Center. [online] Available at: <http://www.uctc.net/research/papers/UCTC-FR-2011-07.pdf> [Accessed 15 February 2016].

Klosterman R.E., 2007. Deliberating About the Future. In *Engaging the Future: Forecasts, Scenarios, Projects and Plans*. Lewis D. Hopkins and Marisa A. Zapata, eds. Cambridge, MA: Lincoln Institute of Land Policy, pp. 199-220.

Koomen, E., Hilfering, M., Borsboom-van Beurden, J., 2011. Introducing the Land Use Scanner. In *Land-Use Modeling in Planning Practice*. Koomen E., Borsboom-van Beurden J., (eds.) 2011 Series: GeoJournal Library, Vol. 101 , 1st Edition, XVI, ISBN 978-94-007-1821-0.

Lemp, J.D., Zhou, B., Kockelman, K.M., Parmenter, B.M., 2008. Visioning versus Modeling: Analyzing the Land-Use-Transportation Futures of Urban Regions. *Journal of Urban Planning and Development*, Vol. 134, No. 3, DOI: 10.1061/(ASCE)0733-9488(2008)134:3(97).

Lescevica, M., Ginters, E., Mazza, R., 2013. Unified Theory of Acceptance and Use of Technology (UTAUT) for Market Analysis of FP7 CHOReOS Products. ICTE in Regional Development, December 2013, Valmiera, Latvia. *Procedia Computer Science*. Bluemel, E., Ginters, E. (eds.). Elsevier, ISSN: 1877-0509, Volume 26, pp. 51-68.

Linke, Ch.S., 2008. *Local level application of the dynamic land use model METRONAMICA. Assessment and modeling – a case study on the Dutch municipality Weert*. Diploma Thesis at the Technical University Berlin Institute of Landscape Architecture and Environmental Planning.

Macaulay Land Use Research Institute, 2015. *FEARLUS- Framework for Evaluation and Assessment of Regional Land Use Scenarios*. [online] Available at: <http://www.macaulay.ac.uk/fearlus/> [Accessed 19 December 2015].

Clarke, K. C, Gazulis, N., Dietzel, C. K., Goldstein, N. C., 2007. *A decade of SLEUTHing: Lessons learned from applications of a cellular automaton land use change model*. Chapter 16 in Fisher, P. (ed.) Classics from IJGIS. Twenty Years of the International Journal of Geographical Information Systems and Science. Taylor and Francis, CRC. Boca Raton, FL. pp. 413-425

EC. European Commission Decision C 6776, 2015. *Horizon 2020. Part 11. Smart, green and integrated transport*. [online] Available at: <http://ec.europa.eu/research/participants/data/ref/h2020/wp/2016_2017/main/h2020-wp1617-transport_en.pdf> [Accessed 15 February 2016].

Ginters, E., Aizstrauts, A., Dreija, G., Ablazevica, M., Stepucev, S., Sakne, I., 2013. *Deliverable 4.3 – Intermediate Release FUPOL Simulator Software prototype*, 127.

Kim, D., Batty, M., 2011. *Calibrating Cellular Automata Models for Simulating Urban growth: Comparative Analysis of SLEUTH and Metronamica*. Working papers series, Paper 176, ISSN 1467-1298.

Piera, M.-A., Buil, R., Ginters, E., 2013. State space analysis for model plausability validation in multi agent system simulation of urban policies. In *25th European Modeling and Simulation Symposium (EMSS 2013)*. Athens, Greece, Code 100159, pp. 504-509.

RIKS, 2015. *Research Institute for Knowledge Systems*. Metronomica Documentation. [online] Available at: <http://www.riks.nl/products/Metronamica> [Accessed 13 March 2015].

Waddell, P., 2011. Integrated Land Use and Transportation Planning and Modeling: Addressing Challenges in Research and Practice, *Transport Reviews*, 31: 2, DOI: 10.1080/0144164, pp. 209-229.

12. Aizstrauts, A., Burkhardt, D., Ginters, E., & Nazemi, K. (2020). On Microservice Architecture Based Communication Environment for Cycling Map Developing and Maintenance Simulator. In *2020 61st International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS)* (pp. 1-4). doi: 10.1109/ITMS51158.2020.9259299. Retrieved from https://www.researchgate.net/publication/347072223_On_Microservice_Architecture_Based_Communication_Environment_for_Cycling_Map_Developing_and_Maintenance_Simulator#fullTextFileContent

On Microservice Architecture Based Communication Environment for Cycling Map Developing and Maintenance Simulator

Artis Aizstrauts
Sociotechnical Systems OU
Tallinn, Estonia
artis.aizstrauts@gmail.com

Dirk Burkhardt
Darmstadt University of Applied Sciences
Darmstadt, Germany
dirk.burkhardt@h-da.de

Egils Ginters
Department of Modelling and Simulation
Riga Technical University
Riga, Latvia
egils.ginters@rtu.lv

Kawa Nazemi
Darmstadt University of Applied Sciences
Darmstadt, Germany
kawa.nazemi@h-da.de

***Abstract*—Urban transport infrastructure nowadays involves environmentally friendly modes of transport, the most democratic of which is cycling. Citizens will use bicycles if a reasonably designed cycle path scheme will be provided. Cyclists also need to know the characteristics and load of the planned route before the trip. Prediction can be provided by simulation, but it is often necessary to use heterogeneous and distributed models that require a specific communication environment to ensure interaction. The article describes the easy communication environment that is used to provide microservices communication and data exchange in a bicycle route design and maintenance multi-level simulator.**

Keywords—simulation, microservice architecture, easy communication environment

I. Introduction

The development of processes in society and in the economy is accelerating. The introduction of digital technologies in all industries and in public life is changing the way society lives. On the one hand, the processes are becoming more transparent, but on the other hand, they are more vulnerable and subject to external influences. Development and introduction of technologies is non-linear, since it is determined not only by the quality of technology, but exclusively by social demand. The predictable result of management is determined by many random factors and the interaction between technical and social components.

Certain corrections are made by changes in the environment. Today, almost any really functioning system is sociotechnical, where technologies and social environment are inseparable. Modelling such systems is complex and expensive. The use of analytical methods leads to serious simplifications in describing the functioning of the system. Simulation is used to save funding and achieve a more transparent and flexible result. Simulation allows to play around different development scenarios and makes it easier to design software, since the notation for describing the simulation models takes a higher level of abstraction and becomes closer to the perceptions of developers. Basically, simulation is used to identify a development trend or forecast but is rarely used to calculate an accurate result.

Since modern systems are heterogeneous, so are simulation models. To achieve the result, it is necessary to interconnect several different technologies of simulation, data analytics [1] and a set of distributed software modules. There are well-developed and widely used methods of providing communication and interaction of distributed simulation models, for example, HLA [2].

However, in this case, one cannot do without special knowledge in programming, which causes additional problems and financial costs for stakeholders. The communication environment presented by the authors in this article is more accessible for modelers and provides the necessary flexibility to implement rapid changes in the architecture of the simulator.

The modern city and its transport system are a classic example of a complex sociotechnical system with a multi-level and distributed architecture. Considering the functioning as the interaction of engineering and social components can stimulate an increase in the productivity of planning and system safety [3].

The authors of this article have been working on the VeloRouter cycling map simulator in recent years. The designing was based on findings of previous EU FP7 project FUPOL [1]. VeloRouter users are municipal policy planners and citizens, as it serves both infrastructure design and daily travel planning.

The purpose of this article is to explain the fundamentals of the Easy Communication Environment (ECE) and to show its application to ensure the functioning of VeloRouter [5].

The following sections outline the basic assumptions, concepts and functionality of both VeloRouter and the proposed communication mechanisms based on the microservices architecture.

II. VeloRouter – A Simulator for Planning and Operating a Network of Bike Paths

Today, more than half of the world's population lives in cities, with steady growth trends [6].

The rapid spread and development of technologies and social media has led to the development of areas such as urban analytics and participatory planning. Hence, more thoughtful decisions can be made and promoted.

Four main urban planning objectives can be highlighted [7]:

- *Lively cities* are achieved when citizens in the urban space are free to use certain transport routes and, with the help of bicycles and similar means of transportation, acquire the possibilities of free movement and communication.

978-1-7281-9105-8/20/$31.00 ©2020 IEEE

- *Safety* is achieved by reducing the number of necessary heavy urban transport and the density of the coverage.

- *Sustainability* is encouraged through "green mobility", that is, walking, cycling or more environment friendly public transport.

- Reducing environmental pollution improves the *city's health*. To achieve this is necessary to promote movement on foot or by vehicles that do not use fossil fuels.

The above basic principles determine the increase in the share of green transport. One such means of transport is a bicycle, the use of which requires adequate infrastructure. The tracks can also be used by scooters and skaters. The cycling route map must be complete and ensure movement throughout the city, respecting the needs of the population. However, financial constraints do not allow the construction of quality roads in any part of the city. To respect these limitations, cyclists must be directly involved in the planning process (see Fig. 1).

the impact of the developing map of new routes. Existing tools such as petitions.net, bikemap.net or mapmyride.com cannot solve the above complex problems [5].

VeloRouter is a tool for the designing and operation of urban bike paths, based on the participation of the population and other stakeholders [8] (see Fig. 1).

There are three main stakeholder groups: municipalities, infrastructure developers and cyclists. The provided advantages of VeloRouter are as follows:

- The statistics section of VeloRouter allows to calculate and analyze the most popular travel routes. *Municipal professionals* can get feedback from the citizens and offer a new cycle path construction.

- *Developers* can use the digital twin of the cycling infrastructure to predict depreciation and timely plan the repair work.

- *Cyclists* can proactively predict the load on the route and participate in infrastructure development by presenting and discussing their own and other projects.

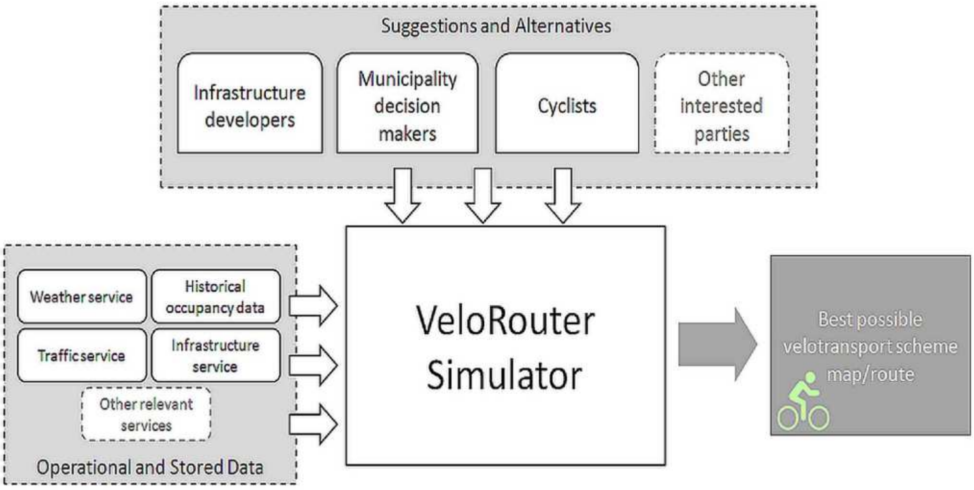


Fig. 1. VeloRouter: Conceptual model of a system for planning and operating urban bike paths and routes.

In order to increase the share of cycling, the infrastructure must be of enough quality. Possible routes must be connected and well-lit at night. Required attributes are bicycle / scooter rental, repair services and monitoring and coordination of the transport network. For a successful trip, proactive route planning must be ensured, which must respect the quality and terrain of each segment. To temporarily plan the arrival at the destination of the trip, it is necessary to predict the load of certain sections of the route on different days of the week and weather conditions.

At the same time, citizens do not have the appropriate tools to effectively interact with decision-makers. There is no feedback on the condition of the existing infrastructure and

The VeloRouter simulator uses four basic data sets: weather data, cycling route load data, traffic forecasts, and infrastructure condition data.

Weather data consists of historical data and forecasts for a possible trip. The cyclist can determine which weather conditions are suitable for each of them. This helps to predict the possible load on the route, that is, it is possible to find out how many cyclists will be on one segment or route at a time. The traffic service provides a forecast of traffic congestion on the streets. The infrastructure data set includes information on the technical characteristics and actual condition of each cycle path. OpenStreetMap spatial data is used to build the road network.

VeloRouter simulator has a multi-level heterogeneous architecture. Although agent-based simulation in Repast Symphony is mainly used, data exchange and synchronization with discrete-event simulation must be ensured. Sub-models' collaboration is implemented based on microservices interaction.

III. Easy Communication Environment

VeloRouter simulator is built on a microservice ECE architecture. The communication mechanism of the micro-service is based on the Advanced Message Queuing Protocol (AMQP), which ensures the interaction of the elements of the application set. In turn, the harmonization of data formats is implemented by assistance of JavaScript Object Notation (JSON). Such basic principles ensure the resilience of the implemented solutions, that is, it is possible to replace damaged and occupied micro-services in critical situations. The implementation becomes independent of the programming platform, because only the interface is important. This facilitates the addition of functionality by the introduction of new features. There is an opportunity to easily create new links and interactions between sub-models by designing larger network architectures [9].

The ECE communication mechanism is specified in A, B, C, and D lanes (see Fig. 2).

1) A - ECE presentation lane. The essence of the presentation lane is similar to the sixth level protocol of the

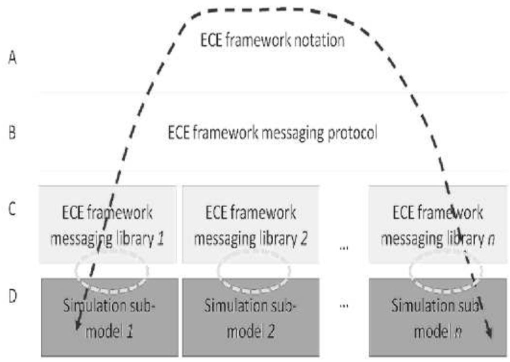


Fig. 2. Stratification of the Easy Communication Environment.

ISO-7498 open systems Basic Reference Model. The main task of A-lane is to harmonize data formats to ensure the exchange of sub-models messages. The JSON language and a harmonized data exchange message format are used. The procedure for exchanging messages is determined by the AMQP protocol.

2) B - ECE messaging control and transportation lane. The protocol is responsible for transmitting messages in the microservice architecture between the individual sub-models that make up the simulator's task network.

The lower lane set of the ECE stratification within the meaning of ISO-7498 describes the transport station.

The ECE model has a minimal set of standard messages. The communication scheme is shown in Fig. 3 and Fig. 4. Any interaction begins with sending a Request message to the respondent's sub-model.

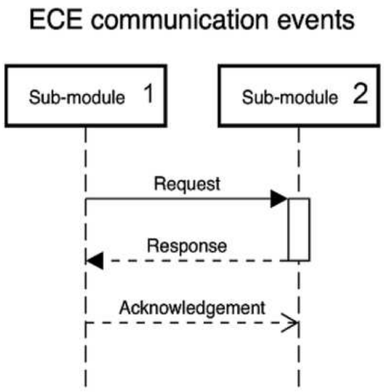


Fig. 3. The sequence of communications in ECE.

```
{
  "type": "request",
  "request_guid": "2fbec59a-7686-4f96-9686-88aaff7ef3a5",
  "service_guid": "9b0ae71b-aeb6-42db-bbcf-2875cafb0a39",
  "service_endpoint": "closes_path",
  "payload": {
    "location_node_start": 847584,
    "location_node_end": 234903
  }
}
```

Fig. 4. Formation of the message Request.

Each simulation sub-model works in parallel and autonomously until it receives a Request from another sub-model. The interrupt called by the request is processed and the respondent sub-model sends a Response message. To consider a session established, the initiator responds with an Acknowledgment message. In this way, an asynchronous handshake procedure is implemented.

For the successful implementation of two or more sub-model interactions, all lanes of the ECE communication model must be involved. Fig. 5 shows in detail the lane cooperation to ensure the exchange of messages between two objects in the ECE environment.

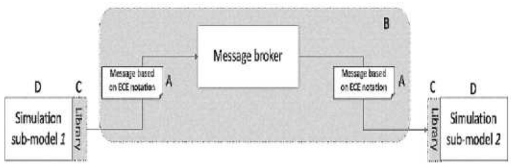


Fig. 5. Message broker use for interaction between the two sub-models.

The message transport and monitoring protocol AMQP [10] provides message registration and queue processing, as well as message routing in the ECE environment. One of the reasons for using AMQP is the high delivery confidence and resilience of the mechanism [11]. Confidence is considered appropriate if the mechanism complies with the procedures set out in the second level of quality of service (QoS) [12].

The basic level allows to save resources of communication session time, however, does not guarantee safe and reliable communication. The message is sent but not acknowledged [13]. Such a mechanism is possible if the messages are insignificant and informative, such as announcements or advices, but this approach is not applicable if each message initiates actions that are important for the functioning of the system. Such a mechanism could be used in broadcasting systems, but it would not be useful to invest resources in sending a message to a specific respondent without asking him to reply.

Although the second level of QoS is considered sufficient in the ECE ideology, there are serious doubts as to whether such a mechanism can be used in simulation models that need to work in real time, such as automatic guided vehicle positioning on the route.

3) C – ECE messaging library. The main task of the library is to harmonize the simulation model notation with JSON and AMQP to ensure interaction with the brokerage mechanism. It is the broker who is responsible for delivering each message to the correct respondent.

4) D - Sub-models. Sub-models provide the implementation of specific simulation algorithms, whose common interaction determines the overall simulation result. Distributed simulators are built in this way [15]. With this approach, sub-models are allocated to specific domains, i.e. load balance management, parallelization of the modeling process, etc. Communication is critical to maintaining distributed modeling and achieving the primary goal of modeling.

The ECE communication model in the VeloRouter simulator is used to ensure cooperation between both simulation sub-models and real services. The cyclist has proactively prepared the planned travel route in appropriate weather conditions, if the load and quality of the bicycle paths meet the wishes of the group of riders. However, road repairs may take place at any time, as well as changes in weather conditions, which cause changes in the load and quality of the previously simulated route. In this case, the load simulation sub-model interacts with other VeloRouter services using both historical and on-line data, while the ECE mechanism provides simulator services interaction.

The above approach ensures the reliability and transparency of data exchange between respondents of the ECE architecture.

IV. CONCLUSIONS

Designing and operating a multimodal and sustainable urban transport system is challenging. The use of simulation helps to achieve the desired result. However, the heterogeneous structure of tasks determines the distributed architecture of the simulator, which also consists of several interacting sub-models.

The communication environment of the VeloRouter - urban cycling modeling and operation system is built based on microservices. Microservices allow to separate the performance of different functions, as well as facilitate maintenance and prototyping. To ensure distributed simulation sub-models interaction reliability and scalability the ECE communication and data exchange mechanism is based on JSON data formats and the AMQP protocol. The present architecture can subsequently be used to build communication systems for other distributed simulators, the operation and modification of which are implemented without the participation of software engineers.

ACKNOWLEDGMENT

The article publication is initiated by FP7 FLAG-ERA FuturICT 2.0 (2017-2020) "Large scale experiments and simulations for the second generation of FuturICT" STIMUL project "Intermodal bicycles network as efficient kind of urban green transport (VeloRouter 2.0)".

REFERENCES

[1] I. Sitova, and J. Pecerska, “Process Mining Techniques in Simulation Model Adequacy Assessment”, in *Proc. of 60th International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS 2019)*. ISBN 978-1-7281-5710-8, IEEE, pp. 49-53, 2019.

[2] IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)— Object Model Template (OMT) Specification. ISBN 978-0-7381-6249-2, IEEE Computer Society, 2010.

[3] L. J. Hettinger, A. Kirlik, Y. M Goh, and P. Buckle, “Modelling and Simulation of Complex Sociotechnical Systems: Envisioning and Analysing Work Environments”, *Ergonomics*, 58(4), pp. 600-614, 2015.

[4] E. Ginters, A. Aizstrauts, G. Dreija, M. Ablazevica, S. Stepucev, I. Sakne, M. Baltruks, M. A. Piera Eroles, R. Buil, M. Gusev, and G. Velkoski, "Skopje Bicycle Inter-modality Simulator – E-involvement Through Simulation and Ticketing", in *Proc. of 26th European Modelling & Simulation Symposium (EMSS 2014)*. ISBN 978-88-97999-38-6, Bordeaux, pp. 557-563, 2014.

[5] E. Ginters, M. Baltruks, I. Sakne, and J. Merkurjevs, “Dual Use Bicycle Path Network Designing and Exploitation Environment – VeloRouter”, in *Proc. of European Modeling and Simulation Symposium (EMSS 2016)*, Cyprus, pp. 10-14, 2016.

[6] United Nations Population Division. World Urbanization Prospects: 2018 Revision, 2019. Available; https://data.worldbank.org/indicator/SP.URB.TOTL.in.zs. [Accessed: October 27, 2019].

[7] I. Gehl, “Cities or People”, Island Press, 2010.

[8] E. Ginters, M. Mezitis, and D. Aizstrauta, "Sustainability Simulation and Assessment of Bicycle Network Design and Maintenance Environment", in *Proc. of the IEEE 2018 International Conference on Intelligent and Innovative Computing Applications (ICONIC)*. ISBN 978-153866477-3, Mauritius, pp. 1-7, 2018.

[9] M. Amundsen, M. McLarty, R. Mitra, and I. Nadareishvili, “Microservice Architecture: Aligning Principles, Practices, and Culture”, O'Reilly Media, 2016.

[10] S. Vinoski, “Advanced Message Queuing Protocol”, IEEE Internet Computing, pp. 87–89, 2006.

[11] OASIS Advanced Message Queuing Protocol (AMQP) Version 1.0, 2017. Available: http://docs.oasis-open.org/amqp/core/v1.0/. [Accessed: October 25, 2019].

[12] N. Naik, “Choice of Effective Messaging Protocols for IoT Systems: MQTT CoAP AMQP and HTTP”, in *Proc. IEEE Int. Syst. Eng. Symp. (ISSE)*, pp. 1-7, 2017.

[13] Message Quality of Service (QOS), 2019. Available: https://www.ibm.com/support/knowledgecenter/en/SSFKSJ_9.1.0/com.ibm.mq.dev.doc/q123795_.htm#q123795___qos. [Accessed: October 25, 2019].

[14] JavaScript Object Notation, 2019. Available: https://www.json.org. [Accessed: October 25, 2019].

[15] A. Aizstrauts, E. Ginters, M. Baltruks, and M. Gusev, “Architecture for Distributed Simulation Environment”, *Procedia Computer Science*, vol. 43, pp. 18-26, 2014.

13. Aizstrauts, A. & Ginters, E. (2024). Easy Communication Environment for Heterogeneous and Distributed Simulation Models Design. In *2024 36th European Modeling & Simulation Symposium* (p. 1-13). Tenerife. doi: 10.46354/i3m.2024.emss.010. Izgūts no https://www.cal-tek.eu/proceedings/i3m/2024/emss/010/pdf.pdf

36th European Modeling & Simulation Symposium, 010
21th International Multidisciplinary Modeling & Simulation Multiconference

2724-0029 © 2024 The Authors.
doi: 10.46354/i3m.2024.emss.010

Easy Communication Environment for Heterogeneous and Distributed Simulation Models Design

Artis Aizstrauts[1] and Egils Ginters[2,*]

[1]Sociotechnical Systems OU, Sakala 7-2, Tallinn, 10141, Estonia
[2]Riga Technical University, Zunda Krastmala 10-522, Riga, LV-1048, Latvia

[*]Corresponding author. Email address: egils.ginters@rtu.lv

Abstract

The rapid and pervasive development of digital technologies not only affects but determines the life of society and the economy, as well as changing our living models. Today, society can be described as a complex socio-technical system, the model of which is heterogeneous, stochastic, and the results are therefore difficult to predict. To simulate the functioning of such a system, a distributed and heterogeneous model is needed. However, the designing of such simulation models is difficult due to the lack of easy-to-use communication tools that are not laborious and do not require specific software engineering knowledge from the modeler. The article explains the Easy Communication Environment (ECE) architecture and functionality that has been developed and validated over time, changing implementation stack but keeping the basic idea of designing a communication environment unchanged. The ECE methodology developed by the authors provides the modeler, who does not have specific skills in software engineering, the ability to design distributed and heterogeneous simulation models, as well as to ensure their interaction. The content of the article will be useful both for researchers and professionals in various fields who need to use simulation for the analysis of complex systems.

Keywords: Distributed simulation; Easy Communication Environment; High-Level Architecture; Simulation technologies

1. Introduction

Digital technologies have become the cornerstone of our society and economy. The quality of these technologies determines our well-being and living conditions. The market for digital technologies is growing at immeasurable speed and volume. There is a digital transformation of our lives, where digital technologies are changing not only industry, but also our behaviour and communication patterns. The penetration of artificial intelligence (AI) and development of Big Data analytics, cloud computing, mobile technologies, business intelligence and Internet of Things (IoT), as well as the global and pervasive impact of social media have a significant impact on our life.

The size of the global digital transformation market in 2023 was $880.28 billion, but it is expected that it reaches $1,070.43 billion in 2024. The market growth is characterized by a compound annual growth rate of about 27.6%. Projections suggest that the size of the digital transformation market will reach $4,617.78 billion by 2030 (Grand View Research, 2024).

The existence of society today can be described as a sociotechnical system, the sustainability of which depends on the development of the sustainability of digital technologies. The stochastic nature of digital technology, which includes a hidden set of influences (Ginters and Revathy, 2021), as well as a pervasive nature, has the most diverse and unexpected impacts on the development of society, the economy, and the environment, as well as on the daily life of everyone.

The number of resources available and explored by mankind is decreasing, climate changes are taking place, the life cycle of technologies is shortening, AI tools are creating both a positive effect and have a negative impact on society. The above determines the increasing relevance of the need for reasonable forecasts. In turn, the development of a reliable forecast and the validation of various scenarios require modeling of the problem.

The model of a sociotechnical system is a simplified reflection of objective reality, however, respecting the high proportion of stochastic influence factors, it remains complex. Complex problems cannot be explained by primitive and/or homogeneous patterns. The model of a real sociotechnical system is heterogeneous and usually requests for different simulation technologies use. If, for example, discrete-event simulation allows for good modeling of processes related to queues and delays, but it is better to use agent-based models (ABM) to study the behaviour and interaction of individual objects, then the overall changes will be better specified by system dynamics simulation equations. If also the sections of applied data processing and analysis must be added, then a heterogenous distributed modeling system must be designed and exploited.

In 2012, when conducting an analysis of distributed simulation platforms and tools, the authors (Aizstrauts et al., 2012) found that the number of options is very limited with Distributed Interactive Simulation (DIS), High-Level Architecture (HLA), Common Object Request Broker Architecture (CORBA), or customizable web service-based solutions, but there are practically no universal and easy to use tools for designing distributed simulation systems models.

And even today, nothing fundamental has changed. Separate sufficiently fast communication tools Robot Operating System (ROS) and Lightweight Communications and Marshalling (LCM) have emerged, which were created for real-time robotics applications, and the situation has forced modelers to use those to somehow compensate for the lack of tools needed to design distributed models. (Xu et al., 2021). Also, the development of IoT has contributed to the development of special device-device communication tools such as IoTivity (Xu et al., 2021). However, these tools serve poorly to provide heterogeneous model collaboration. A ray of hope was the protocol and standard Data Distribution Service (DDS) (OMG, 2015) created by Object Management Group for real-time distributed operational systems. However, these are only instructions and guidelines to achieve model interoperability, but to bring the recommendations to life, the author of each model will have to invest serious time and software engineering resources.

It must be honestly acknowledged that the still dominant tool to ensure the interaction of simulation models is the High-Level Architecture (HLA), which was created by the US Department of Defence back in 1995 (Dahmann, Fujimoto and Weatherly, 1997), but it is also supplemented, improved and widely used in the civilian sectors (Jabbour, Possik and Zacharewicz, 2023; Possik, 2021). In fact, HLA has become the standard solution for providing distributed models interoperability. One of the main drawbacks of HLA was and still is "*Standard is too "heavy", i.e. very complex, difficult to learn and thus time consuming to adopt and use*" (Strassburger, Schulze and Fujimoto, 2008), and without good skills in software engineering its use is impossible.

The use of the above-mentioned solutions requires well financial justification, since high level software engineering knowledge is untypical for representatives of other industries who are aimed at creating distributed problem-oriented models. So, for professionals in other industries, this path to creating a distributed model is still closed, or it is necessary to attract software engineering specialists who will have to translate the insights and regularities of the industry. So, the life cycle of this model will most likely end with a loss of interest from the software engineering specialist. Moreover, the licensing terms of commercial tools will impose an additional financial burden on further distribution of the designed model.

To address the above problem, simulation environment developers incorporate multiple simulation technologies at the same time, which makes it possible to construct heterogeneous models, such as AnyLogic (Borshchev, 2014). However, the question remains how to integrate models already created earlier from other simulation environments, such as FlexSim (Wang, Wang and Zhang, 2022), NetLogo (Wilensky and Rand, 2015) and STELLA (Dissanayake, 2016), etc., as well as provide a link to the subsystems of data analysis.

There is an urgent need for a tool that would enable modelers to make the previously impossible possible, and to allow industry professionals with basic knowledge in information technology and modeling to create distributed heterogenous simulation models themselves, without the involvement of coders and additional financial expenses.

If the modeler works with ABM, then he has at least a minimal understanding of object-oriented programming. If the modeler uses discrete-event tools, then he has knowledge of statistical processing of data and understands probabilities. If the modeler develops system dynamics models, then he is familiar with differential equations. That is, he has enough knowledge to create a distributed and heterogenous simulation model using Easy Communication Environment (ECE), which has been developed and improved since 2008 (Aizstrauts et al., 2012).

The aim of the article is to discuss the Easy Communication Environment (ECE) concept, which provides heterogeneous and distributed simulation models design capabilities to any specialist in other fields with appropriate knowledge in modeling.

The audience of the article is researchers and industry professionals engaged in the modeling of processes and phenomena, as well as designers of data processing systems who want to create distributed data processing and exchange systems. The authors' approach could also be of interest to designers of digital twins.

In Section 2, the authors review the development of ECE, starting with the first attempts to replace HLA with CORBA and similar solutions used in software engineering, and ending with the design of an HLA alternative. Section 3 of the article analyzes the structural architecture and functionality of ECE, while Section 4 examines a specific application example and various issues related to the use of ECE.

2. State of the art

In 2007, the authors conducted research on Ligatne Natural Trails, which is part of the Gauja National Park (Ginters and Silins, 2007). In a limited area, moose, deer, wild boar, wolves, bears, and other representatives of the national fauna live there in a natural environment. In order not to damage natural resources, the sightseeing places connect pedestrian tourist trails. However, for the convenience of lazy visitors, an asphalt road has been arranged, for which private cars can move along a certain route and order, since there is a parking lot next to each sight. The question of the research was, what is the permissible load on the natural resource, so that the damage caused by visitors does not lead to irreversible consequences and the resource is able to recover at a certain time interval?

It was necessary to create a discrete-event model that provides an analysis of the movement of cars to understand how large parking spaces should be allocated and how equipping them with WC and other should be carried out. The model was implemented in the Extend Suite environment and served as a data source for the second ABM model in the NetLogo environment, which assessed the regenerative capacity of the natural resource.

Then the question arose, how to ensure the interoperability of models? An analysis of existing distributed simulation support mechanisms was carried out. DIS and HLA were found to be usable. However, the implementation of an interoperability mechanism between the models in the HLA was laborious and complex. It became clear that designing distributed models using the tools HLA and similar is impossible without good knowledge of software engineering, and for nature researchers this path is closed.

The authors continued looking for distributed simulation communication tools that are usable to other industry professionals (Ginters, Silins and Andrusaitis, 2007).

An analysis of the suitability of the Aggregate Level Simulation Protocol (ALSP), CORBA, The Foundation for Physical Intelligent Agents (FIPA) and HLA was carried out. Special attention was paid to the CORBA mechanism, whose Object Request Broker (ORB) was used to ensure communication between Extend Suite and NetLogo models. Simulation environments extension modules C++ (Extend Suite) and Java (NetLogo) were implemented. The solution provided models communication, unfortunately, change management without serious software engineering knowledge was impossible.

In 2008, the adaptation of the CORBA ORB mechanism for distributed simulation continued (Ginters and Silins, 2008a), however, the low CORBA performance, which affected the modelling quality, began to cause problems.

The authors (Ginters and Silins, 2008b) returned to performance measurements and made comparisons between CORBA and HLA communication mechanisms. The suspicions were confirmed as the length of the message packets increased, significant HLA advantages over CORBA began. It was decided that the use of CORBA in distributed simulation is unsustainable.

In 2010, the authors published a solution (Silins, Ginters and Aizstrauta, 2010) that eased access to the HLA federation. The solution was based on the Communication Adapter, which provided a link between the simulation model and HLA. The adapter was created as part of the HLA environment and interpreted the HLA data in a format that was understandable by the simulation model. The communication adapter listened to the models and accumulated the sent messages in the data storage. Another model that needed the data could read them from the data storage on demand. At the same time, data storage provided time synchronization options. Each simulation environment required a library to interact with the adapter. Cooperation between the simulation tool and adapter was initiated by requests in XML format. In conformity with the model parameters the adapter generated the Federation Object Model (FOM), which allowed to connect to the HLA federation. However, this adapter should have been easy enough to manage since the parameters of the simulation models are variable. The implementation of this requirement turned out to be more complicated than it seemed at first glance, as it provoked corresponding changes in the chain of compatibility with HLA. Thus, the first version of Easy Communication Environment (ECE) was born.

In 2011, the improvement of the first version of ECE continued, working on supplementing the simulation tools Communication library. An intermediate

Communication Gateway was created that provided the data exchange among Communication Adapters prior to connection to the HLA environment (Aizstrauts, Ginters and Aizstrauta, 2011). The improvement reduced the modeler's workload by creating HLA-based distributed simulation models.

Since the first versions of the ECE were based on the presence of HLA, the search for more suitable communication environments, architectures, protocols, and message data formats continued (Ginters et al., 2011).

The authors attempted to apply the ideology of ISO OSI 7498 in distributed simulation, creating the Simulation Highway concept as a multi-layered open architecture. The Simulation Highway was formed by the set of simulation cells, but the distributed simulation task was accomplished by the inclusion of appropriate simulation cells in the task chain. The requirements for specific software engineering skills were eased, as high-level languages such as SimAL and SimQL were allowed for modeling requests in accordance with the client-server approach. However, the communication backbone was still the HLA. In addition, the implementation of Simulation Highway required very significant resources. After a self-assessment of the sustainability development of Simulation Highway, the authors abandoned this idea.

In 2012, the FP7 project No.287119 FUPOL was launched. In the framework of the project, the development of several heterogeneous simulators was carried out both for the design of the zoning of the public park in Zagreb, the forecasting of the occupancy of tourist facilities and the planning of the bicycle route network of the city of Skopje (Ginters et al., 2014).

The FUPOL project was the next step of ECE development (Aizstrauts et al., 2012), as it made it possible to validate the quality of the previously developed ECE version. To implement and test response time boundary intervals in a territorially distributed model, the ECE was deployed in a virtual computing environment of Amazon Elastic Compute Cloud (EC2), while simulation model management was done with Remote Desktop Protocol (RDP). In this version of the ECE and in the future, the authors abandoned the use of HLA, expanding the functionality of the previously designed Communication Gateway.

One of the main tasks of FUPOL project was to create a simulation-based support mechanism for policy decision-making. The consortium conducted an analysis of more than 60 generic and problem-oriented simulation tools, which confirmed earlier suspicions that most problem-oriented simulation environments are closed, and it is not possible to attach external models. Unfortunately, closed environments usually offer limited functionality that did not provide an implementation of heterogeneous distributed models.

The basic concept of the FUPOL simulator was developed (Aizstrauts et al., 2013), where Fuzzy Cognitive Maps and Colored Petri Networks were used at the higher level of design, while simulation models were implemented in RePast Symphony. However, STELLA, a system dynamics simulation environment, was used to simulate system changes over time. Data processing and analysis were implemented in Python. The data was stored in PostgreSQL base. To improve the visualization of the results and facilitate the management of the modeling sessions, the NetLogo environment was used, because the visualization mechanism of the RePast Symphony was rather poor. Data exchange in FUPOL simulation subsystem was provided by the ECE. To ensure an interoperability with other FUPOL systems an ECE alignment with the Enterprise Service Bus was designed. The FUPOL project provided excellent validation opportunities for the ECE concept.

One of FUPOL's use cases where the ECE environment was validated was the Skopje Bicycle Routes Simulator (Aizstrauts et al., 2015). Simulator was aimed to provide route choices for cyclists, depending on the load on the route, the quality of the pavement and infrastructure, as well as meteorological conditions. Here the ECE had to ensure interoperability between NetLogo, RePast and Python.

The analysis of several Message Brokers was carried out, recognizing as the most promising RabbitMQ, which was subsequently used as an ECE component. The Advanced Message Queuing Protocol (AMQP) was used by simulation model to communicate with the Message Broker. Henceforth the authors of the ECE abandoned the use of any clouds for communication needs, which did not provide a guaranteed response time and raised doubts about the timeline of the simulation session. The FUPOL validation results replenished the ECE set of requirements, and specifically, not only the convenience of the modeler is important, but also the universality of the communication environment.

In the following a prototype of the cycle route planning tool Velorouter was created (Aizstrauts et al., 2020). The research was initiated by previous FUPOL results and FP7 FLAG-ERA FuturICT 2.0 (2017-2020) project. Velorouter users had the ability to develop their own routes, as well as recommend them to the municipality, providing feedback through ticketing, which is a basic rule for the development of sustainable management and control.

Here, for the first time, the multi-layer architecture of the ECE was presented and the functionality of each layer was defined. Based on the universality and compatibility requirement, the ECE communication and data exchange mechanism was based on JSON data formats and the AMQP protocol disclaiming further use of XML.

The ECE versions were validated in several national and international research projects:

- No. 2006/11 "Simulation Tools EXTEND and NetLogo use for Ecosystems Analysis" funded by the Latvian Ministry of Education and Science.
- No. 2007/1-17/26 "Communication Environment of Hybrid Simulation Systems" funded by the Latvian Ministry of Education and Science.
- No.2DP/2.1.1.2.0/10/APIA/VIAA/001 "Support for preparation of IST FP7 STRE project "Simulation Highway"" funded by ERDF.
- FP7 No.287119 FUPOL „Future Policy Modelling" funded by European Commission.
- FP7 FLAG-ERA FuturICT 2.0 (2017-2020) "Large scale experiments and simulations for the second generation of FuturICT" funded by European Commission.

The research results were validated and used in the study course "Sociotechnical Systems Modeling" (Vidzeme University of Applied Sciences and Riga Technical University). Research results have been reported at 15 international conferences.

The ECE's multi-layer architecture and a description of the functionality will be discussed below.

3. Methodology

The structural and functional description of the Easy Communication Environment explains the basic principles of architecture and layer interoperability.

3.1. Structural Model of Easy Communication Environment

The Easy Communication Environment (ECE) is a versatile and user-friendly communication tool. The primary goal is to facilitate communication between diverse simulation models. Additionally, it aims to be user-friendly for distributed simulation model designers, even those with limited software engineering knowledge.

Easy Communication Environment's structural multi-layer model has an open system architecture based on the basic building principles of OSI ISO 7498 (Ginters et al., 2011) and Dijkstra machines (Dijkstra, 1968).

The ECE structural model (1) (see Figure 1) consists of four layers, each of which is responsible for specific and functional tasks - ECE framework messaging protocol (A), ECE framework notation layer (B), ECE framework messaging library (C) and Simulation sub-model (D).

ECE framework is determined by

$$ECE = \langle A, B, C, D \rangle \quad (1)$$

where

A – ECE framework messaging protocol. Messaging protocol consists of two functional parts – message transportation and message carrier format.

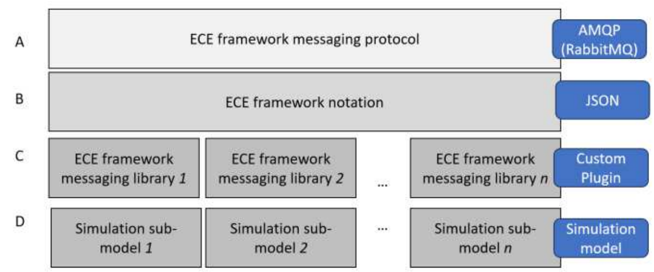


Figure 1. ECE structural model.

For message transportation ECE uses Advanced Message Queuing Protocol (AMQP) protocol. It is an open standard application layer protocol for message-oriented middleware. The defining features of AMQP are message orientation, queuing, routing (including point-to-point and publish-and-subscribe) and reliability (Naik, 2017). Reliability is one of the core features of AMQP, and it offers two preliminary levels of Quality of Service (QoS) for delivery of messages – "At most once" and "At least once".

ECE uses "At most once" service level, the recipient does not acknowledge receipt of the message. Data storage functionality is implemented at ECE architecture abstraction layer C. For the message carrier ECE uses JSON data format.

ECE combines AMQP and JSON to deliver sophisticated communication architecture that is highly reliable and easy maintainable. AMQP ensures that message is delivered to the right client/subscriber (and resilience) and JSON allows to transmit complicated data that can be parsed and understood by most of the programming languages.

B – ECE framework notation layer. Notation layer is one of the highest levels in ECE communication architecture. This layer ensures that each node (sub-model) under-stands each other and speaks the same language. ECE notation is based on JavaScript Object Notation (JSON) data format and requires three fields for the message:

- *tick* - represents simulations time when this message has been sent.
- *datatype* – describes what type of data has been sent.
- *data* – the data to send to other simulation models.

C – ECE framework messaging library. Messaging library is custom built software that integrates into modelling toolkit, simulation suits, etc. as extensions or plugins. The abstraction layers of the ECE architecture are generic and open, so it is possible to develop such extensions/plugins for different simulation tools independently.

The ECE messaging library (C-layer) includes the basic functions that enable interoperability with the D-layer (see Table 1).

Table 1. ECE messaging library (C-layer).

Function	Description
init	Initialises the connection to AMQP server.
Close	Closes the connections to AMQP server.
isConnected	Returns the status of the connections to AMQP server.
Subscribe	Subscribes to communications topic. Will receive and store all the messages that are sent to that topic.
Unsubscribe	Unsubscribes from topic, will not receive any messages to this topic and purges all messages that were received before.
sendString	Sends String type message to specific topic.
sendInteger	Sends Integer type message to specific topic.
sendDouble	Sends Double type message to specific topic.
sendBoolean	Sends Boolean type message to specific topic.
getStringData	Returns String data for specific topic at specific tick.
getIntegerData	Returns Integer data for specific topic at specific tick.
getDoubleData	Returns Double data for specific topic at specific tick.
getBooleanData	Returns Boolean data for specific topic at specific tick.
getLastStringData	Returns String data for specific topic with the latest (largest) tick.
getLastIntegerData	Returns Integer data for specific topic with the latest (largest) tick.
getLastDoubleData	Returns Double data for specific topic with the latest (largest) tick.
getLastBooleanData	Returns Boolean data for specific topic with the latest (largest) tick.
getLastTick	Returns the latest (largest) tick for specific topic.

D – Simulation sub-model. Sub-model is any program or algorithm that consumes or produces any data. Such sub-model uses API of extensions/plugin from C-layer to send or receive data from other models. To ensure mutual interoperability of two or more simulation models (D-layer) and data exchange during the session, a chain of cooperation between ECE layers is established (see Figure 1).

The next chapter will explain the ECE communication algorithm in more detail.

3.2. Functionality and Implementation

The interoperability of the ECE A, B, C and D-layers is shown in the BPMN2 diagram (see Appendix A), demonstrating the collaboration between the two simulation models (D-layer). The processes occurring in each layer during the simulation session proceed in parallel. Parallel processes exchange messages and signals, and a process that occurs in one layer can initiate and/or affect another process in the other layer.

The ECE initialization operation is activated at the D-layer "Simulation sub-model" by sending a signal to the C-layer "ECE framework messaging library". If the model wants to send data, then the appropriate library in the C-layer is called. If the model does not have any data to be transmitted, and the simulation session is over, then a signal is sent that stops the plug-in of the model in the C-layer.

C-layer processes are activated by a signal that is received from the D-layer simulation model. Upon receiving a signal about the launch of the simulation session, the C-layer provides a connection to the AMQP server. If a data message is received from the D-layer, then a raw data object is created from it, which is transferred to the B-layer "ECE framework notation layer". Receiving a data object activates the B-layer, where the message is transformed in the JSON data format, and then the message is sent back to the C-layer. After that the message in JSON format is sent to the AMQP server on the A-layer "ECE framework messaging protocol". If the C-layer has not received a signal about the end of the simulation session, then the process of sending data continues, otherwise the connection to the AMQP server is interrupted. The A-layer is activated by data message received from the C-layer. The message is recorded to an AMQP storage queue corresponding to a specific data topic. At the same time, the AMQP server broadcasts a queued data message.

The sent message receives/process the model whose topic it is needed. In the C-layer of the other simulation model, the received AMQP message is converted to internal raw format and recognized, but then converted to JSON format again in B-layer and sent back to C-layer, where the message is stored in topic local storage. If the process is not interrupted by a signal received from the D-layer about the end of the simulation session, then it is divided in two directions - a request to read data from the D-layer is pending and a signal to subscribe to topic is expected at the same time. The data object in the internal format is then sent to the simulation model (D-layer), while the message in JSON format goes to the queues in the A-layer, which correspond to the relevant topic and are deployed in the AMQP storage.

4. Results and Discussion

The extract from FUPOL's cycling route planning heterogeneous and distributed simulation model, where communication is provided by the ECE, is discussed below.

The task of the distributed and heterogenous model is to simulate the load in different segments of the route, so that the cyclist can combine the route that suits him.

It is not the full model of Skopje network simulation, but its middleware, which is aimed to show the use of the ECE.

The model concept map (see Figure 2) includes simulator objects that provide the necessary functionality.

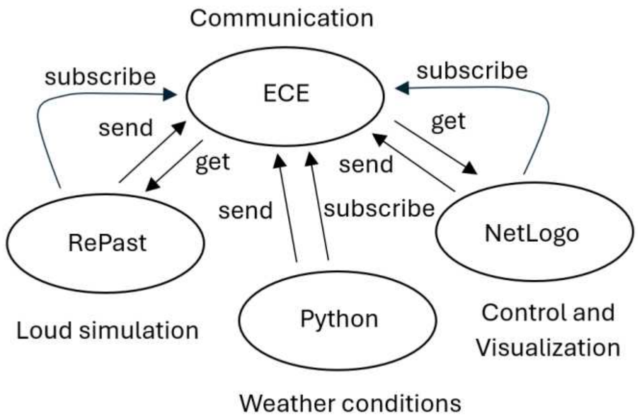


Figure 2. Simulator concept map.

Segments load simulation is provided by RePast agent-based model. Weather data is coming from the Python part, while simulator control and information visualization are implemented in the NetLogo environment.

All simulator sub-models are connected to the ECE, which provides communication and data exchange options. Models and requests to libraries make layer D of the ECE. Each model specification (D-layer) includes commands that ensure that notifications/data are sent and received (subscribe/send/get).

The task of the modeler is to embed the appropriate commands in the model specification. ECE parts A, B and C are not the subject of interest to the modeller.

Causal loop diagram (CLD) (see Figure 3) describes the concept of simulator functionality.

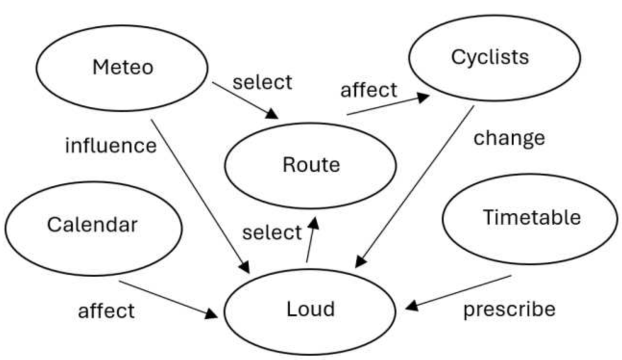


Figure 3. Causal loop diagram of route planning.

The simulator helps to the cyclist in selecting right segments of travel route. The selection of segment is influenced by meteorological conditions, which can impair the quality of the path surface. On the other hand, in bad weather, the load on routes decreases due to a decrease in the number of cyclists. Route load is affected by the season (winter, summer) and calendar (weekdays and holidays), during which the number of cyclists increases/decreases. The load affects the choice of a particular route segment, since the profiles of the cyclist are different and determined by the set of attributes: skills, type of bike, age, presence of children in the group, size of the group, etc. Each selection increases the number of cyclists on the segment, which in turn leads to an increase in load. The increase in loud reduces the probability of route selection in further iterations. The initial conditions of the simulation session are adjusted with real load data creating a balanced system.

Simulation workbench (see Figure 4) is implemented in NetLogo and ensures control and visualization options. The user interface has three compartments – Control, Map visualization and Results.

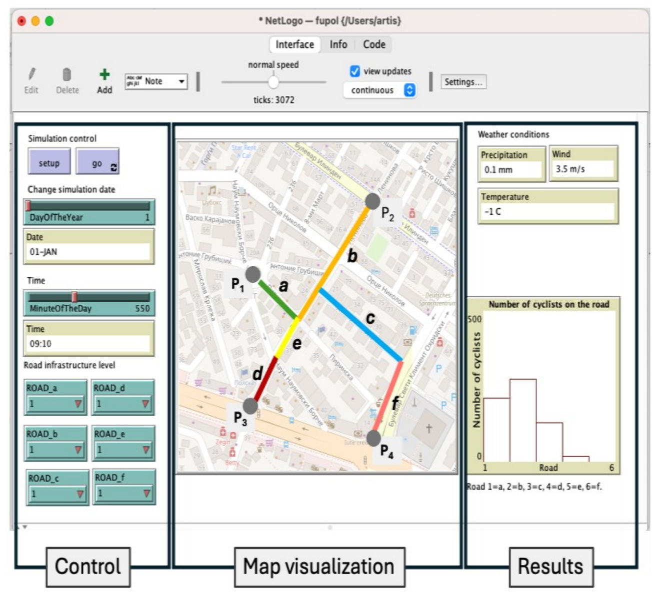


Figure 4. Simulation model workbench.

Setup button initializes model and ECE communication, that is, defines global variables and subscribe for ECE session. In this case all the global values will be received from other models via ECE. ECE Netlogo plugin instance (C-layer) will asynchronously receive data for each of these subscribed topics and store it locally with specific timestamp (tick from the message itself) (see Figure 5).

The C-layer libraries at the beginning of the simulation session must connect to ECE AMQP server. It is done with commands *ece: init*, in this case connection is to *localhost*.

Go button start/stop simulation process. Starting time and selected travel date are manually adjustable. Time is defined as minute of 24 hours, but date is consecutive day of year.

User can define the bicycle lane coverage and infrastructure quality with drop-down menus (ROAD_i). In this model, the user simulates the load on six segments of a possible route. The map shows four starting points/destinations (P_1, P_2, P_3 and P_4), bicycle lanes among these points are split into six edges or segments (a, b, c, d, e, and f). Lane load is simulated in RePast for each segment separately.

```
extensions [ece]
globals[
  last_temperature
  last_wind
  last_precipitation
  new_cyclist_data
  last_new_cyclist_data_tick
]

to setup_ece
  ece:init "localhost"
  ece:subscribe "FUPOL_SIMULATION_weather_temp"
  ece:subscribe "FUPOL_SIMULATION_weather_wind"
  ece:subscribe "FUPOL_SIMULATION_weather_precip"
  ece:subscribe "FUPOL_SIMULATION_new_cyclist_agent"
end
```

Figure 5. NetLogo Setup implementation.

The Results compartment displays the possible weather conditions provided by Python part for the selected day and time (precipitation, wind speed and temperature) and shows a bar chart representing the potential load for each segment simulated. According to the load results, the cyclist can decide to include this segment in his route or reject it.

NetLogo model software (D-layer), includes requests for C-layer libraries (see Figure 6).

```
extensions [ece]
globals[last_temperature last_wind last_precipitation new_cyclist_data
        last_new_cyclist_data_tick]
to setup_ece
  ece:init "localhost"
  ece:subscribe "FUPOL_SIMULATION_weather_temp"
  ece:subscribe "FUPOL_SIMULATION_weather_wind"
  ece:subscribe "FUPOL_SIMULATION_weather_precip"
  ece:subscribe "FUPOL_SIMULATION_new_cyclist_agent"
end
to send_time_and_date
  ece:sendNumber "FUPOL_SIMULATION_time" ticks MinuteOfTheDay
  ece:sendNumber "FUPOL_SIMULATION_date" ticks DayOfTheYear
end
to send_road_configuration
  ece:sendNumber "FUPOL_SIMULATION_road_a" ticks ROAD_a
  ece:sendNumber "FUPOL_SIMULATION_road_b" ticks ROAD_b
  ece:sendNumber "FUPOL_SIMULATION_road_c" ticks ROAD_c
  ece:sendNumber "FUPOL_SIMULATION_road_d" ticks ROAD_d
  ece:sendNumber "FUPOL_SIMULATION_road_e" ticks ROAD_e
  ece:sendNumber "FUPOL_SIMULATION_road_f" ticks ROAD_f
end
to get_last_weather_data
  set last_temperature ece:getLastNumberData "FUPOL_SIMULATION_weather_temp"
  set last_wind ece:getLastNumberData "FUPOL_SIMULATION_weather_wind"
  set last_precipitation ece:getLastNumberData "FUPOL_SIMULATION_weather_precip"
end
to get_new_cyclist_data
  if last_new_cyclist_data_tick != ece:getLastTick("FUPOL_SIMULATION_new_cyclist_agent")[
    set new_cyclist_data ece:getLastStringData("FUPOL_SIMULATION_new_cyclist_agent")
    set last_new_cyclist_data_tick ece:getLastTick("FUPOL_SIMULATION_new_cyclist_agent")
  ]
end
```

Figure 6. NetLogo model D-layer specification.

Weather simulation is developed in Python programming language. ECE library (C-layer) provides Python ECE class collection (D-layer) with functions to communicate via ECE framework. The main purpose of this model is to broadcast the weather conditions for travel date and time.

Load simulation is implemented in RePast Symphony according to a previously validated algorithm (Ginters, Aizstrauts et al., 2016). RePast C-layer library is designed in generic pattern and can be implemented in any Java application. The user's D-layer includes requests for C-layer libraries, such as NetLogo and Python.

And now a little about challenges and problems. In the example considered, to access the ECE, the task of the modeler is to include D-layer commands in the sub-model code, while the higher layers of the ECE are responsible for the rest of the communication. If the modeler has sufficient knowledge to prepare NetLogo, RePast, or other simulation model code, then he will have sufficient skills to incorporate the necessary ECE commands. The level of complexity is not even comparable to those software engineering skills required for HLA use.

The performance of the distributed model is determined by the model's interoperability algorithm designed by the modeler, while ECE provides only communication tube. The same can be said for HLA. The exciting question would be, which of the communication environments HLA or ECE has higher performance?

Since ECE is based on AMQP, which is one of the fastest protocols, the possible latency associated with ECE operation will be lower than in an HLA environment. AMQP is significantly faster than HTTP, so if the modeler deploys the simulations on the web environment, then ECE will not be the reason for the delay.

Is AMQP fast enough for real-time applications when a modeler creates digital twins for real-time control of IoT objects? IoT applications usually use more primitive protocols, such as Message Queue Telemetry Transport (MQTT), because shorter data messages are used to control the technical system. If ECE will be based on the MQTT protocol, then its use for heterogeneous systems design would be severely limited. However, the RabbitMQ stack also includes MQTT, so customizing ECE wouldn't be a problem.

In recent years, the term "digital twins" has become popular, which describes a simulation model working in parallel with the controlled object. The simulation model receives data from the external environment and simulates the operation of a real object. The peculiarity is that the result of modeling must be achieved in real time, otherwise an error is detected. This is nothing new, because successfully working simulation models that control real physical objects were tested back in the late 90s. For example, Alexander Verbraeck of Delft University of Technology demonstrated to the authors the Automatic Guided Vehicle (AGV), which transported flowers through underground tunnels between Schiphol Amsterdam and Flower Auction (Versteegt and Verbraeck, 2002). The CORBA environment was used. This was a very interesting example, since it was problematic in underground tunnels to provide control feedback, so the possible location of the AGV was calculated by a simulation model that worked in real time. Could the ECE be used here? Of course, because CORBA and ECE performance are similar, but the convenience of applying ECE is significantly better.

HLA is rarely used in real-time applications. The cornerstone of HLA is not maximum performance, but safety. HLA provides callback functions that reside in each federate, which increases safety and the load of computing resources, which in turn causes delays in communication. The ECE broadcasting mechanism is simpler and therefore faster.

ECE will provide the modeler with reasonable timeliness, scalability, modularity, safety, and performance parameters, however, if the modeler wants to design distributed simulation systems that support human life-critical functions, and programming in C++ does not cause any problems, then it would still be more reasonable to use the old, the well-tested, standardized, heavy, inconvenient, complex and expensive HLA mechanism.

Still, attempts to humanize HLA and replace it with Service Oriented Architecture (SOA) in critical simulation applications have failed (Iagaru, 2022).

5. Conclusions

The sociotechnical systems that permeate our society and industry are complex because they are influenced by many different stochastic factors, of which human effect is the most significant and least predictable. To check scenarios of the development of various situations and the possible consequences of our actions, simulation is used. Analytical solutions in conditions of high uncertainty and real-time, when the system evolves during a modeling session, are not sufficiently usable. Modeling of the behaviour and interaction of individual objects using analytical approach are especially difficult.

Complex processes and phenomena cannot be specified by homogeneous simulation models. Patterns become distributed and heterogeneous. A major problem is ensuring the exchange of data between these models and coordinating their operation over time.

This is amazing, but still distributed simulation uses HLA, which is not possible to modelers without specific knowledge and good skills in software engineering.

This is a problem that slows down adequate policy decision-making and analysis of various scenarios of development in any industry, as it becomes dependent on the coding skills of third parties. In addition, part of the model knowledge usually disappears in translation.

A second option remains, to use one of the multifunctional but closed simulation environments and hope that this environment will be sustainable and versatile, and with a long enough life cycle.

Over 17 years, the authors have developed and validated the Easy Communication Environment (ECE) data exchange environment for distributed and heterogeneous simulation, which is accessible to professionals in other industries without specific software engineering skills.

A stack of flexible protocols and data formats has been designed, the changes of which do not cause problems in running simulation models. Multi-layer open architecture provides interoperability of various simulation technologies.

The advantages of ECE are its application not only in the development of distributed simulation models, but also in the provision of software subsystems interoperability, so ECE audience is not only researchers and modelers, but also developers of distributed data processing systems.

When developing the ECE communication mechanism, the authors deliberately did not use message confirmation, which creates a delay in information exchange. Therefore, ECE is not recommended for simulating systems with specific safety requirements.

The concept and functionality of the ECE have been validated in several national and European Commission funded projects, while research results have been published in various sources and announced at international conferences.

For the time being, ECE lacks methodological and training materials, as well as not having a large enough library of simulation tools, however, the authors will continue to work on the preparation of these deliverables. The next step of ECE will be connecting heterogeneous simulation models to a Bayesian network to support cyber threat forecasting for individual nodes connected to the data transmission network.

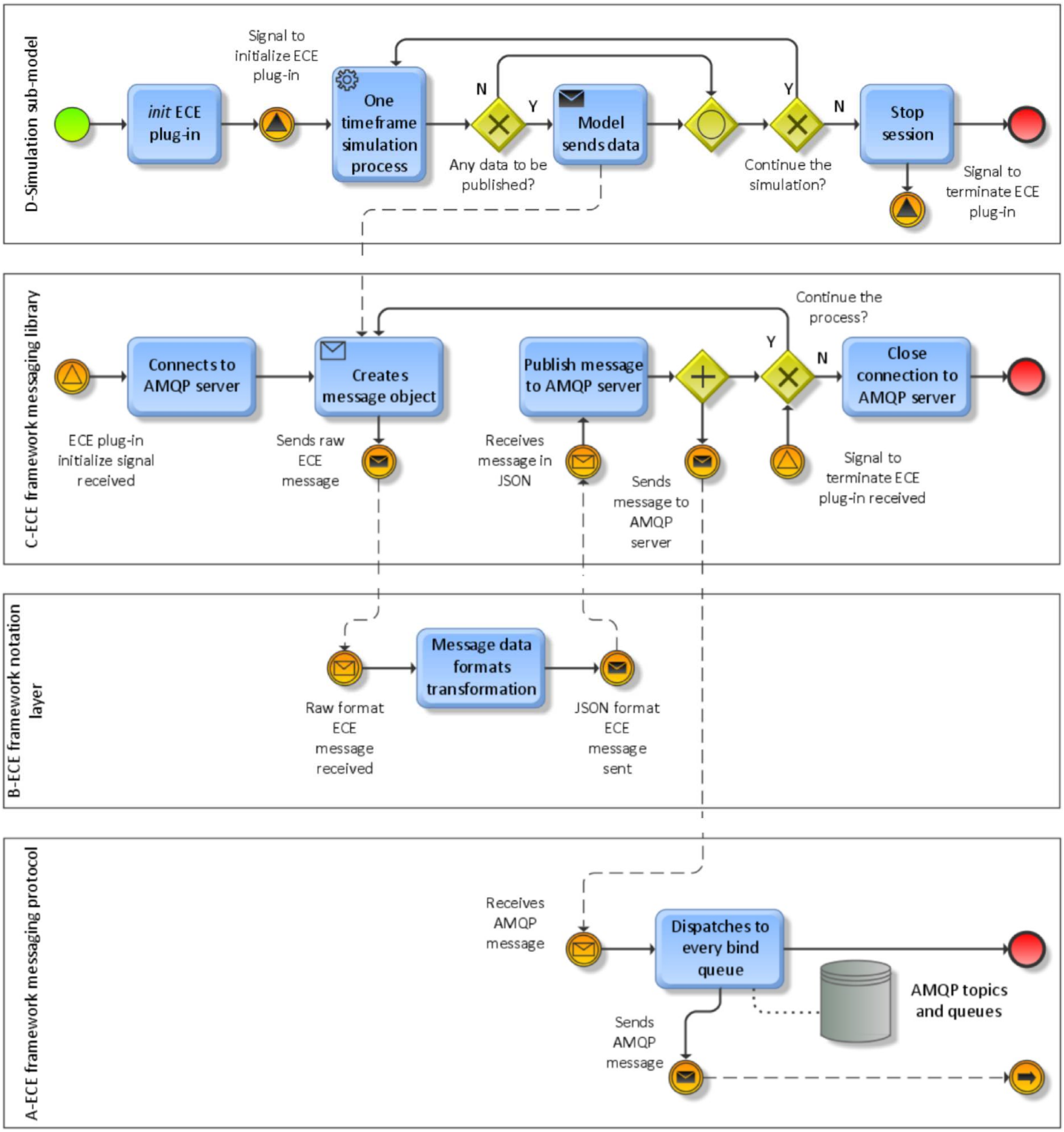
Acknowledgements

The authors would like to thank Vidzeme University of Applied Sciences and Information Technology Institute of Riga Technical University for their support during the development of ECE research.

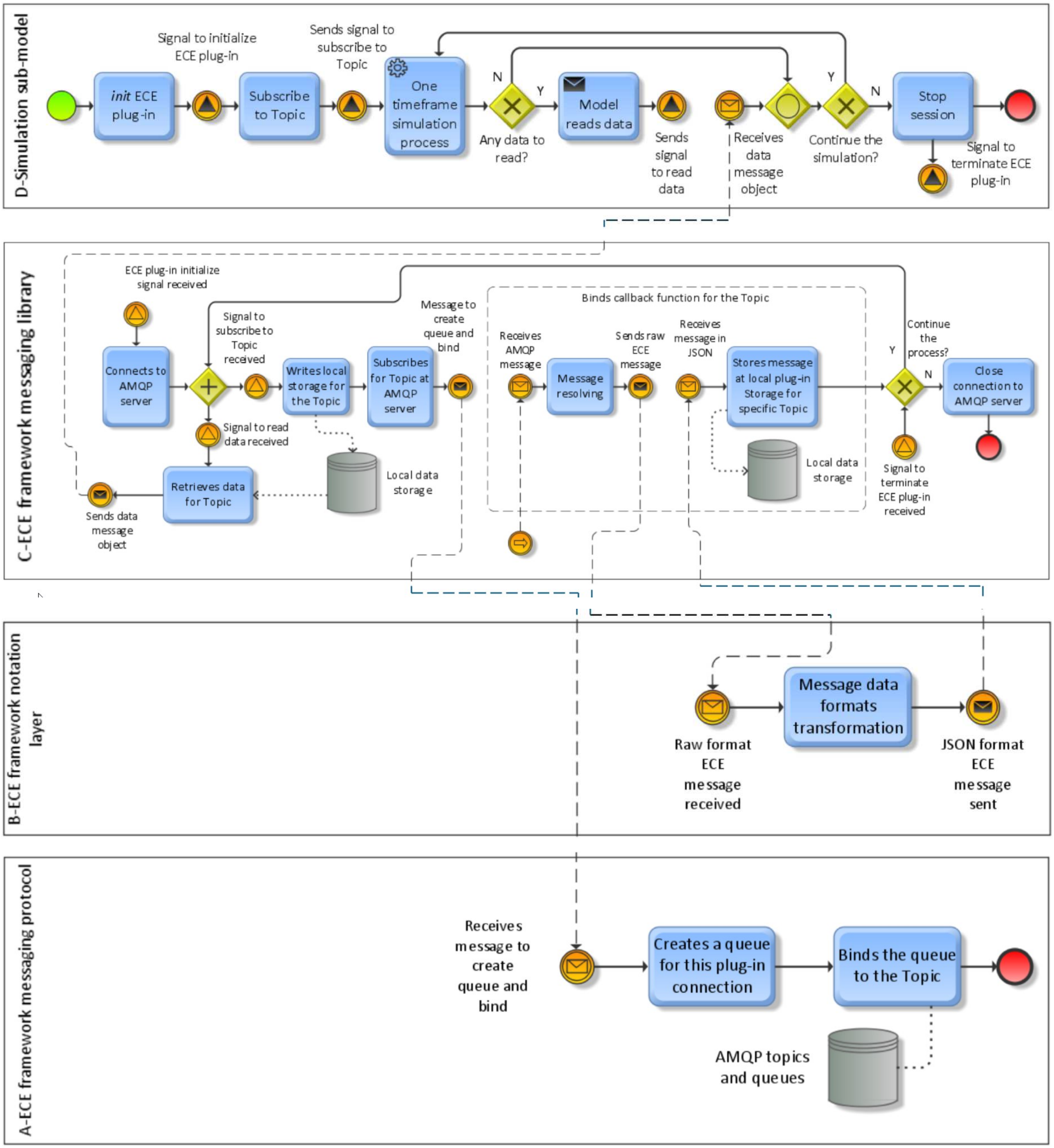
The development of the ECE is the longest project in the simulation life, so the authors would like to thank not only their colleagues who endured the never-ending improvements and modifications of the ECE, but also their families, who reconciled and allowed them to devote time to ECE research.

Appendix A. BPMN2 diagram of ECE layer interactions

The diagram describes the send/receive operations between two simulation models, looking at the macro-level interoperability of the ECE layers implemented at both ends of one and the other model.



Appendix A. BPMN2 diagram of ECE layer interactions (continuation)



References

Aizstrauts, A., Burkhardt, D., Ginters, E., & Nazemi, K. (2020). On Microservice Architecture Based Communication Environment for Cycling Map Developing and Maintenance Simulator. In *2020 61st International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS)* (pp. 1-4). doi: 10.1109/ITMS51158.2020.9259299. Retrieved from https://www.researchgate.net/publication/347072223_On_Microservice_Architecture_Based_Communication_Environment_for_Cycling_Map_Developing_and_Maintenance_Simulator#fullTextFileContent

Aizstrauts, A., Ginters, E. & Aizstrauta, D. (2011). Easy Communication Approach for Data Exchange in Distributed Simulation Environment. In *Proceedings of the 13th WSEAS international conference on Automatic control, modelling & simulation (ACMOS'11)* (pp. 34-38). Stevens Point: World Scientific and Engineering Academy and Society (WSEAS). Retrieved from http://www.wseas.us/e-library/conferences/2011/Lanzarote/ACMOS/ACMOS-04.pdf

Aizstrauts, A., Ginters, E., Aizstrauta, D., & Sonntagbauer, P. (2012). Easy Communication Environment on the Cloud as Distributed Simulation Infrastructure. In *Proceedings of the 5th WSEAS congress on Applied Computing conference, and Proceedings of the 1st international conference on Biologically Inspired Computation (BICA'12)* (pp. 173-178). Stevens Point, Wisconsin: World Scientific and Engineering Academy and Society (WSEAS). Retrieved from http://www.wseas.us/e-library/conferences/2012/Algarve/BICA/BICA-29.pdf.

Aizstrauts, A., Ginters, E., Baltruks, M., & Gusev, M. (2015). Architecture for Distributed Simulation Environment. *Procedia Computer Science*, *43*, pp. 18-25. doi: 10.1016/j.procs.2014.12.004. Retrieved from https://www.sciencedirect.com/science/article/pii/S1877050914015725

Aizstrauts, A., Ginters, E., Lauberte, I., & Piera Eroles, M.A. (2013). Multi-level Architecture on Web Services Based Policy Domain Use Cases Simulator. In *Lecture Notes in Business Information Processing* (vol 153, pp.130-146). Berlin: Springer. doi:10.1007/978-3-642-41638-5_9. Retrieved from https://www.researchgate.net/publication/282289690_Multi-level_Architecture_on_Web_Services_Based_Policy_Domain_Use_Cases_Simulator#fullTextFileContent

Borshchev, A. (2014). Multi-method modelling: AnyLogic. In *Discrete-Event Simulation and System Dynamics for Management Decision Making* (eds S. Brailsford, L. Churilov and B. Dangerfield). doi: 10.1002/9781118762745.ch12

Dahmann, J.S., Fujimoto, R.M., & Weatherly, R.M. (1997). The Department of Defense High Level Architecture. In *Proceedings of the 29th conference on Winter simulation (WSC '97)* (pp. 142-149). IEEE Computer Society. doi: 10.1145/268437.268465

Dijkstra, E.W. (1968). The Structure of the “THE” - Multiprogramming System. *Commun. ACM*, *11*, 342–346.

Dissanayake, S. T. M. (2016). Using STELLA Simulation Models to Teach Natural Resource Economics. *The Journal of Economic Education*, 47(1), 40–48. doi: 10.1080/00220485.2015.1106358

Ginters, E., Aizstrauts, A., Baltruks, M., Merkuryev, Y., Novickis, L., Grundspenkis, J., & Grabis, J. (2016). VeloRouter - Technology for Urban Transport Intermodal Sustainability. In *City Planning and Urban Design Conference, 07-09 April (CPUD’16)* (pp. 211-220). Istanbul: DAKAM. ISBN 978-605-9207-21-8. Retrieved from https://www.dakamconferences.org/_files/ugd/bac820_3252301ff80b453e928fb159e78a9910.pdf

Ginters, E., Aizstrauts, A., Dreija, G., Ablazevica, M., Stepucev, S., Sakne, I., Baltruks, M., Piera Eroles, M.-A., Buil, R., Gusev, M., & Velkoski, G. (2014). Skopje Bicycle Inter-modality Simulator – e-Involvement Through Simulation and Ticketing. In *Proceedings of 26th Europen Modelling & Simulation Symposium (EMSS 2014)* (pp. 557-563). Retrieved from https://www.msc-les.org/proceedings/emss/emss2014/emss2014_557.html

Ginters, E., Sakne, I., Lauberte, I., Aizstrauts, A., Dreija, G., Aquilar Chinea, R.-M., Merkuryev, Y., Novitsky, L., & Grundspenkis, J. (2011). Simulation Highway – Direct Access Intelligent Cloud Simulator. In *Proceedings of 23rd European Modelling & Simulation Symposium (EMSS 2011)* (pp. 62-72). Retrieved from https://docplayer.net/16307931-Simulation-highway-direct-access-intelligent-cloud-simulator.html

Ginters, E., & Silins, A. (2007). Multi-level approach for environmental systems modelling in the Ligatne Natural Trails. *WSEAS Transactions on Systems*, 6(4), 795-801. Retrieved from https://www.researchgate.net/publication/297118169_Multi-level_approach_for_environmental_systems_modelling_in_the_Ligatne_Natural_Trails

Ginters, E., & Silins, A. (2008a). Exchange Mechanisms in Distributed Simulation of Sociotechnical Systems. In *Information Society and Modern Business. Proceedings of 3rd International conference, 21-22*

September 2007, Ventspils (p. 362). Ventspils: Ventspils University College. Retrieved from https://daugavpils.biblioteka.lv/alisepac/Details?title=Information-Society-and-Modern-Business.-3rd-International-conference-%3A-proceedings-21-22-September,-2007,-Ventspils&Id=85013&Ident=1218592&InstanceId=13&LibraryId=0#bibliographic

Ginters, E., & Silins, A. (2008b). Simulation Data Exchange in Distributed E-learning Environment. In *Proceedings of the 4th WSEAS/IASME International Conference on Education Technologies (EDUTE'08)* (pp. 138-143). Corfu: WSEAS. Retrieved from http://www.wseas.us/e-library/conferences/2008/corfu/edute/edute23.pdf

Ginters, E., Silins, A., & Andrusaitis, J. (2007). Communication in Distributed Simulation Environment. In *ICOSSSE'07: Proceedings of the 6th WSEAS international conference on System science and simulation in engineering* (pp. 217-221). Stevens Point: World Scientific and Engineering Academy and Society. doi: 10.5555/1974442.1974475 Retrieved from http://www.wseas.us/e-library/conferences/2007venice/papers/600-139.pdf

Ginters, E., & Revathy, J. G. (2021). Hidden and Latent Factors' Influence on Digital Technology Sustainability Development. *Mathematics,* 9(21), 2801. doi: 10.3390/math9212801

Grand View Research,. (2024). *Digital Transformation Market Size, Share, Growth & Trends Analysis Report by Solution, By Deployment, By Service, By Enterprise Size, By End-use, By Region, And Segment Forecasts, 2024 – 2030. Report ID: GVR-1-68038-851-0.* Retrieved from https://www.grandviewresearch.com/industry-analysis/digital-transformation-market

Iagăru, E. (2022). Comparative Analysis Between High Level Architecture (HLA) and Service Oriented Architecture (SOA) in the Field of Military Modelling and Simulation. *Scientific Bulletin,* 27(1), pp. 30 - 40. doi: 10.2478/bsaft-2022-0004

Jabbour, J., Possik, J., & Zacharewicz, G. (2023). A Modeling and Distributed Simulation Platform for Immersive Experience. In *JIAE 2023 - Journées sur l'Interopérabilité des Applications d'Entreprise, Pôle Grand Sud-Ouest (PGSO)*. Lille, France. Retrieved from https://hal.science/hal-04199808/document

Naik, N. (2017). Choice of Effective Messaging Protocols for IoT Systems: MQTT, CoAP, AMQP and HTTP. In *2017 IEEE International Systems Engineering Symposium (ISSE)* (pp. 1-7). IEEE. doi: 10.1109/SysEng.2017.8088251

OMG (Object Management Group),. (2015). *About the Data Distribution Service Specification Version 1.4*. Retrieved from https://www.omg.org/spec/DDS/1.4

Possik, J., Gorecki, S., Asgary, A., Solis, A.O., Zacharewicz, G., Tofighi, M., Shafiee, M.A., Merchant, A.A., Aarabi, M., Guimaraes, A., & Nadri, N. (2021). A Distributed Simulation Approach to Integrate AnyLogic and Unity for Virtual Reality Applications: Case of COVID-19 Modelling and Training in a Dialysis Unit. In *2021 IEEE/ACM 25th International Symposium on Distributed Simulation and Real Time Applications (DS-RT)* (pp. 1-7). Valencia: IEEE.

Silins, A., Ginters, E., & Aizstrauta, D. (2010). Easy Communication Environment for Distributed Simulation. In *Computational Intelligence in Business and Economics* (pp. 91-98). https://doi.org/10.1142/9789814324441_0014. Retrieved from https://www.researchgate.net/publication/282288675_Easy_Communication_Environment_for_Distributed_Simulation#fullTextFileContent

Strassburger, S., Schulze, T., & Fujimoto, R. (2008). Future Trends in Distributed Simulation and Distributed Virtual Environments: Results of a Peer Study. In *2008 Winter Simulation Conference* (pp. 777-785). Miami, FL, USA. doi: 10.1109/WSC.2008.4736140. Retrieved from https://ieeexplore-ieee-org.resursi.rtu.lv/stamp/stamp.jsp?tp=&arnumber=4736140

Versteegt, C., & Verbraeck, A. (2002). The Extended Use of Simulation in Evaluating Real-Time Control Systems of AGVs and Automated Material Handling Systems. In *Proceedings of the 2002 Winter Simulation Conference* (pp. 1659-1666). San Diego. doi: 10.1109/WSC.2002.1166448

Xu, L., Lin, SY., Hlynka, A.W. *et al.* (2021). Distributed Simulation Platforms and Data Passing Tools for Natural Hazards Engineering: Reviews, Limitations, and Recommendations. *Int J Disaster Risk Sci*, *12*, 617–634. doi: 10.1007/s13753-021-00361-7

Wang, S., Wang, S.M., & Zhang, N. (2022). Flexsim-based Simulation and Optimization of Green Logistics Distribution Center. In *Proceedings of the 14th International Conference on Computer Modeling and Simulation (ICCMS '22)* (pp.76-82). New York: Association for Computing Machinery. doi: 10.1145/3547578.3547590

Wilensky, U., & Rand, W. (2015). An Introduction to Agent-Based Modeling. Modeling Natural, Social, and Engineered Complex Systems with NetLogo. MIT Press.