DATU APMAIŅAS ARHITEKTŪRU UN MULTIMEDIJU TEHNOLOĢIJU ANALĪZE

ANALYSIS OF DATA EXCHANGE ARCHITECTURES AND MULTIMEDIA TECHNOLOGIES

Autori: **Artūrs ZALUŽINSKIS**, Bc. comp., e-pasts: artchijs2@inbox.lv, telefons: 27812630,
Edgars TABORS, Bc. comp., e-pasts: mania1993@inbox.lv, telefons: 28672283
Zinātniskais vadītājs: **Imants Zarembo, Dr. sc. ing.,** e-pasts: imants.zarembo@rta.lv
Rēzeknes Tehnoloģiju akadēmija

Abstract. *The main objectives of the research work are to classify information systems by the field of usage and to analyze the data exchange architectures and multimedia technologies for successful development process of electronic information systems. The main data exchange architectures are described, the possible methods of interconnection possibilities of technologies are analyzed.*

Anotācija. *Darbā ir dota informācijas sistēmu klasifikācija, analizētas datu apmaiņas arhitektūras un multimediju tehnoloģiju izmantošanas iespējas jaunu elektroniskās komercijas sistēmu izstrādei. Ir aprakstītas pamata datu apmaiņas arhitektūras, kā arī to apvienošanas iespējas efektīvā datu apmaiņas procesa nodrošināšanai.*

Atslēgas vārdi: *datu bāzu tehnoloģijas, datu pārraide, datu indeksēšana, informācijas sistēmas.*

Ievads

Interneta un programmēšanas tehnoloģiju attīstība pēdējos 20 gados veicināja elektronisko informācijas sistēmu un resursu attīstību. Mūsdienu informācijas sistēmas apvieno sevī programmatūras un aparatūras resursu, tehnoloģiju, cilvēkresursu, kā arī uzglabājamo datu mijiedarbību.

Atbilstoši pielietojumam, informācijas sistēmas var klasificēt pēc sekojošajiem kritērijiem[1]:

- ✓ Statiskas informācijas sistēmas: uz teksta bāzes veidotais saturs (informatīvās datubāzes, ziņojumu apmaiņas sistēmas un citi informatīvie resursi);
- ✓ Dinamiskas informācijas sistēmas: dažāda rakstura audio-vizuālais saturs (piem. Youtube, Vimeo), kā arī sociālo tīklu resursi (piem. Facebook, Twitter);
- ✓ Interaktīvās informācijas sistēmas: dažāda veida 3D CAD/CAM/CAE *(Computer Aided Design/Manufacturing/Engineering)* projektēšanas sistēmas (Autodesk Autocad, Dassault Systems Solidworks), programmatūras adaptīvie izstrādes līdzekļi (Microsoft Visual Studio, Android Studio, Xamarin), kā arī Interneta pārlūkprogrammas;
- ✓ Tiešsaistes aplikāciju informācijas sistēmas: tiešsaistes dokumentu un prezentāciju satura vadības sistēmas (Google Docs, Prezi), reāllaika analīzes un monitoringa sistēmas (piem. Microsoft Honolulu, dažāda rakstura datorsistēmu tehnisko raksturlielumu un stāvokļu monitoringa sistēmas).

Jaunu elektroniskās komercijas sistēmu izstrādes procesa pamatā ir pareizu izstrādes tehnoloģiju un līdzekļu izvēle. Efektīvu e-komercijas produktu izstrādi un ieviešanu ietekmē sekojošie faktori:

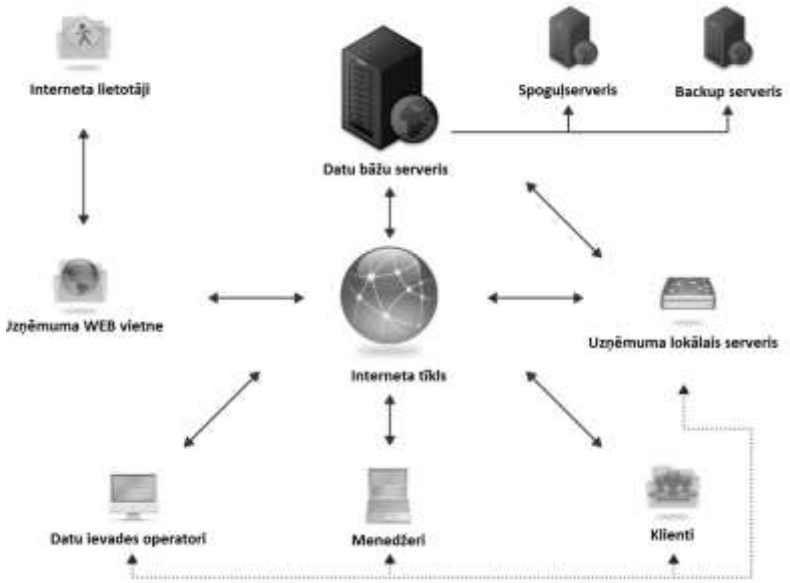
- ✓ Aktuālu programmēšanas līdzekļu, tehnoloģiju un platformu izvēle;
- ✓ Tiešsaistes pakalpojumu sniegšanas veiktspēja un tās optimizācija;
- ✓ Pareizu uzturēšanas līdzekļu izvēle;
- ✓ Risinājuma adaptīvā dizaina izstrādes iespējas, pakalpojumu sniegšanas nodrošināšanai dažādām platformām (galddatoriem, portatīvajiem datoriem, planšetdatoriem, mobiliem telefoniem u.c.);
- ✓ Risinājuma uzturēšanas drošības aspekti, aizsargātā datu pārraides procesa nodrošināšanai (lietojot drošus kriptogrāfijas līdzekļus un aizsardzības protokolus).

http://dx.doi.org/10.17770/het2018.22.3632

Materiāli un metodes

Tiešsaistes elektronisko resursu uzturēšana paredz dažādu elektronisko informācijas sistēmu apvienošanu, kas savukārt nodrošina datu apmaiņu starp pārdevēju un pircēju. Sniedzamo pakalpojumu drošības paaugstināšanai datu pārraide starp pārdevējiem un pircēju tiek nodrošināta, izmantojot dažādus pakalpojumu sniegšanas un datu uzturēšanas resursus:

- ✓ Elektroniskās tirdzniecības nodrošinājuma resursus, izmantojot dažādus šablonus uz satvaru pamata piem. Magento, Joomla! vai Drupal, kā arī individuāli izstrādātus elektroniskās komercijas tirdzniecības līdzekļus;
- ✓ Uzturēšanas risinājumus, izmantojot WEB serverus vai pakalpojumu sniedzēju satura mitināšanas pakalpojumus;
- ✓ Lokāli uzturamus resursus, izmantojot failu serverus vai mākoņrisinājumu servisus;
- ✓ Datu bāžu uzturēšanas līdzekļus, izmantojot relāciju datu bāzes (piem. MySQL, Microsoft SQL Server, Oracle Database, MariaDB vai Firebird), kā arī dokumentorientētus datu uzturēšanas līdzekļus piem. (MongoDB, Redis, Cassandra u.c.);
- ✓ Datu rezerves kopēšanas līdzekļus (piem. Symantec Backup Exec u.c.);
- ✓ Spoguļserverus un datu indeksēšanas resursus (piem. Apache Solr), datu pārraides ātrdarbības paaugstināšanai, uzturēšanas resursu pārslodzes gadījumos, vai aparatūras atteikumpolitikas realizācijai izmantojot administrēšanas rīkus.



1.attēls. Starpsistēmu datu apmaiņas procesu realizācija uz loģistikas kompānijas piemēra [2]

Izstrādājot elektronisko resursu uzturēšanas shēmas, liela uzmanība ir jāpievērš uzturēšanas līdzekļu izvēlei, plānotam klientu skaitam, kā arī sniedzamo pakalpojumu drošībai, lietojot aizsardzības līdzekļus lietotāju autentifikācijai un risinājuma aizsardzībai pret nesankcionētiem uzbrukumiem.

Datu apmaiņas protokolu un tehnoloģiju analīze

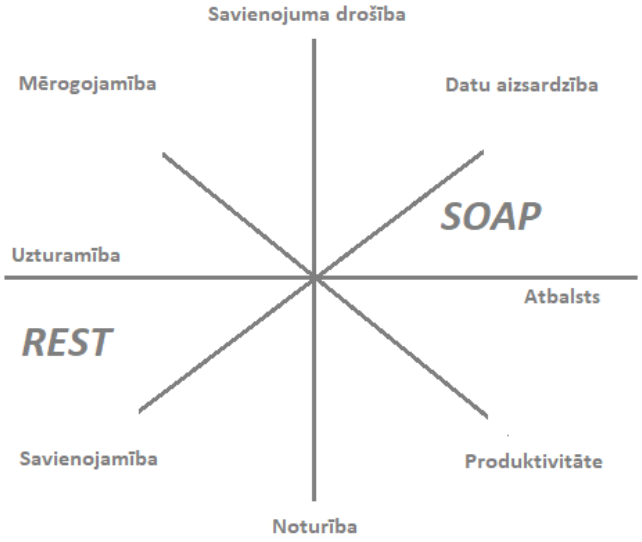
Izmantojot dažādas integrētās izstrādes vides ir iespējams izmantot dažādus datu apmaiņas protokolus un tehnoloģijas elektroniskās komercijas līdzekļu izstrādei. Datu apmaiņas tehnoloģiju, arhitektūru un līdzekļu izvēli ietekmē programmēšanas vides izvēle.

Izstrādājot tiešsaistes elektroniskās komercijas līdzekļus, ir iespējams izvēlēties gan datu apmaiņas dizaina šablonus piem. MVC (*Model-View-Controller*), MVP (*Model-View-Presenter*), Singleton, Spring MVC u.c., gan arī datu apmaiņas tehnoloģijas SOA (*Service*

Orientied Architecture), REST (*Representational State Transfer*), uz kuru pamata iespējams veidot aplikāciju servisus un interfeisus, turpmākai datu apmaiņas procesu realizācijai.

Salīdzinot REST datu apmaiņas arhitektūru un datu pārraidi izmantojot SOAP *(Simple Object Access Protocol)* protokolu, ar kura starpniecību pamatā tiek nodrošināta datu apmaiņa servisorientētā arhitektūrā, var atzīmēt, ka datu pārraides arhitektūras izvēli ietekmē sekojošie faktori, tādi kā [3]:

- ✓ Savienojuma drošība;
- ✓ Datu aizsardzība;
- ✓ Datu pārraides produktivitāte;
- ✓ Sistēmu darbības noturība, saglabājot vai atjaunojot sistēmu pareizu funkcionēšanu kļūdu situācijās;
- ✓ Savienojamības iespējas starp citām saistītām sistēmām vai iekārtām;
- ✓ Uzturamības iespējas – sistēmu veiktspējas uzlabošana, kā arī sistēmu adaptācija darbam uz citām uzturēšanas platformām;
- ✓ Mērogojamības iespējas – sistēmas resursu vai lietotāju skaita palielināšana, saglabājot sistēmu esošo veiktspēju un funkcionālās iespējas.
- ✓



2.attēls. Datu pārraides arhitektūru funkcionālās raksturiezīmes

Izvērtējot SOA un REST datu pārraides arhitektūru apvienošanas iespējas vienas sistēmas ievaros var secināt, ka REST datu pārraides arhitektūras risinājumus ir iespējams pielāgot izmantošanai darbam ar servisorientētās arhitektūras risinājumiem (apmainoties ar XML *(Extensible Markup Language)* ziņojumiem ar HTTP *(Hyper Text Transfer Protocol)* un SOAP datu pārraides protokolu starpniecību), kā arī ziņojumu apmaiņa izmantojot REST datu pārraides arhitektūru tiek nodrošināta ar ievērojami augstāku veiktspēju[4,5].

Analizējot SOA un REST arhitektūru pielietojuma iespējas dažāda veida aplikāciju izstrādē, var secināt, ka:

- ✓ Risinājumu uz servisorientētās datu pārraides arhitektūras pamata darbība ir nodrošināta ar standartizētu protokolu un programmēšanas līdzekļu starpniecību SOAP, WSDL *(Windows Services Discovery Language)*, UDDI *(Universal Description, Discovery and Integration)*, kas nodrošina integrētās aizsardzības funkcijas. Datu pārraidei tiek izmantoti standartizēti XML *(Extensible Markup Language)* formāta ziņojumi, kas netiek kodēti un tilpsaspiesti, kas ievērojami palielina pārsūtamo datu apjomu un palēnina to apstrādi. Uz servisorientētās datu pārraides arhitektūras pamata tiek uzturētās sistēmas ar

paaugstināto aizsardzības pakāpi – dažāda rakstura biznesa klientu vadības sistēmas, tiešsaistes maksājumu sistēmas u.c.;

- ✓ Ziņojumu apmaiņa starp SOAP WEB servisa nodrošinātājiem un saņēmējiem notiek anonīmi, līdz ar to nenotiek apziņošana vai saņēmējs uz atbildes saņemšanas brīdi atrodas tiešsaistē, kas savukārt var novest pie datu zudumiem un servisa darba nespējas gadījumiem. Šādu situāciju apstrādei izstrādātājiem jāparedz izņēmuma gadījumu apstrādi, kas netiek definētas standarta programmas kodā[6].
- ✓ REST tipa datu apmaiņas arhitektūra nodrošina ziņojumu apmaiņu vairākos formātos – XML, JSON *(Java Script Object Notation),* HTML *(Hyper Text Markup Language)*, līdz ar to datu apstrāde tiek realizēta ar ievērojami augstāku veiktspēju, bet salīdzinājumā ar SOA tipa arhitektūru nenodrošina integrētās aizsardzības iespējas. Lai izstrādātu efektīvu informācijas sistēmu uz REST tipa datu apmaiņas arhitektūras pamata, izstrādātājiem, lai izvairītos no potenciālajiem uzbrukumiem, ir nepieciešams lietot papildus aizsardzības līdzekļus.

Ņemot vērā statistikas datus, var secināt, ka REST aplikāciju interfeisu risinājumi iegūst arvien lielāku popularitāti izstrādātāju vidū. Pieprasījums pēc REST tipa arhitektūras pielietojuma ir izskaidrojams ar labākas veiktspējas nodrošināšanas iespējām, mazāka izmēra ziņojumu izmantošanas iespējām datu apmaiņas procesos, kā arī ar labāku savietojamību starp dažādām programmēšanas vidēm.

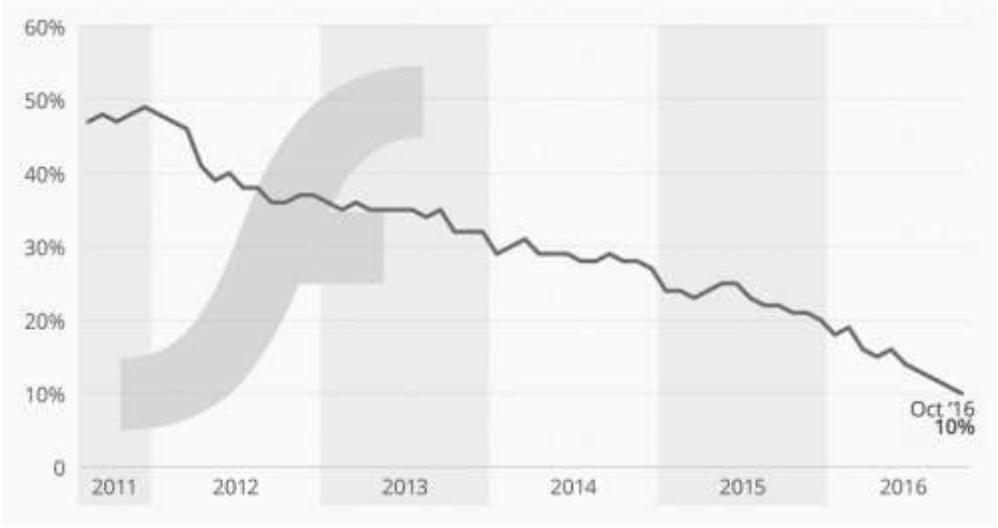
Interaktīvu multimediju tehnoloģiju analīze

Lai uzlabotu preču un pakalpojumu realizācijas iespējas, izmantojot elektroniskās komercijas sistēmas, bieži vien nepieciešams izmantot dažādus interaktīvus līdzekļus: audio un atskaņošanu, procesu izpildes procesu un dažāda specializētā aprīkojuma vizualizāciju, datorspēļu industrijā. Šo procesu izpildes nodrošināšanai tiek izmantoti dažādi interaktīvie līdzekļi un tehnoloģijas.

Digitālā satura animācijas un vizualizācijas rīku un tehnoloģiju izvēli ietekmē šādi faktori:

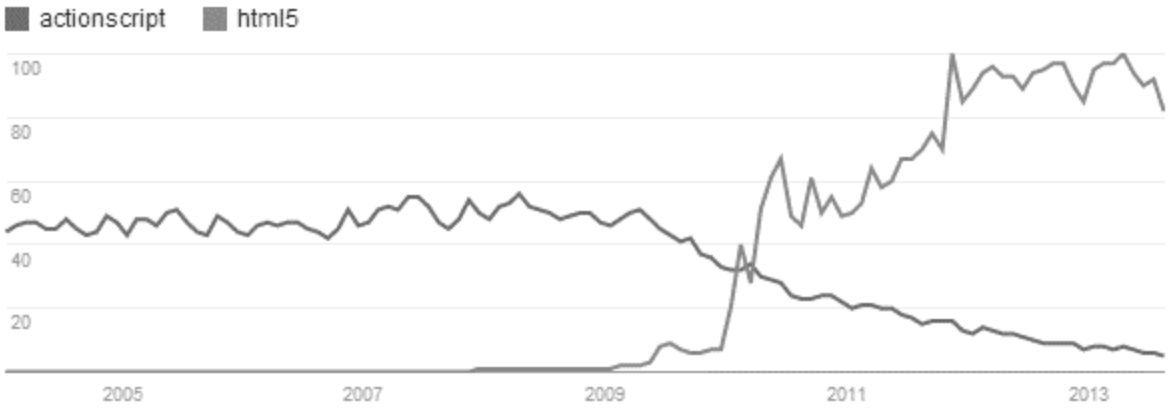
- ✓ izvēlēto tehnoloģiju veiktspēja, salīdzinājumā ar citām tehnoloģijām (objektu renderēšanas un ielādes ātrums);
- ✓ risinājumu un pievienojumprogrammu atbalsts plaši izplatītajās pārlūkprogrammās (Microsoft Egde, Mozilla Firefox, Google Chrome);
- ✓ uzturēšanas vides atjaunināšanas iespējas, lai nodrošinātu aizsardzību no potenciālajiem uzbrukumiem Interneta vidē.

Atsaucoties uz korporācijas Adobe oficiālo informāciju[7], sadarbības rezultātā ar vadošajiem pārlūkprogrammu un servisu ražotājiem (Apple, Mozilla, Facebook un Google) tika panākta vienošanās, par to ka, 2020. gadā tiks pilnīgi pārtraukta Adobe Flash tehnoloģiju uzturēšana un atjaunināšana, ņemot vērā citu saistītu tehnoloģiju strauju attīstību. Ņemot vērā statistikas datus uz 2016 gadu, Adobe Flash tehnoloģijas pielietojuma rādītājs pārlūkprogrammu lietotāju vidū sastādīja tikai 10%, un līdz šim laikam turpina sarukt, salīdzinājumā ar saistītu HTML 5 *(Hyper Text Markup Language)* tehnoloģiju.



3.attēls. Web resursu skaits ar Adobe Flash tehnoloģijas pielietojumu

HTML 5 iezīmēšanas valodā izmantotās sintakses un elementu izmantošana ir reglamentēta uz W3C *(World Wide Web Consortium)* un WHATWG *(Web Hypertext Application Technology Working Group)* standartu pamata un tās izmantošanai nav nepieciešamas speciālās pievienojumprogrammas. HTML 5 attīstība tika uzsākta jau sākot ar 2008. gadu, kad tika radīts pirmais publiskais paraugs. Atšķirībā no Adobe Flash tehnoloģijas, HTML 5 nodrošina plašākas programmēšanas iespējas izstrādātājiem. Statistikas dati liecina par HTML 5 tehnoloģijas strauju attīstību[8].



4.attēls. Adobe Flash un HTML 5 tehnoloģiju izmantošana WEB komponenšu izstrādē

Analizējot Abobe Flash un HTML 5 tehnoloģiju priekšrocības, var secināt ka:

- ✓ Abobe Flash programmatūras lietošanai pārlūkprogrammās, atšķirībā no HTML 5 ir nepieciešama papildus atbalsta programmatūra;
- ✓ Adobe Flash atbalsta programmatūra bieži vien ir pakļauta dažādām problēmām: programmatūras darbības atteikumiem, drošības problēmām, atbalsta problēmām, strādājot ar jaunākām pārlūkprogrammām, līdz ar to atbalsta programmatūru regulāri jāatjaunina;
- ✓ HTML 5 ir savietojams ar dažādiem SEO *(Search Engine Optimization)* risinājumiem, kas padara tehnoloģijas izmantošanu par lietderīgu dažādu risinājumu izstrādei;
- ✓ risinājumi kas ir izstrādāti uz HTML 5 tehnoloģijas pamata var nebūt savietojami ar dažādām pazīstamām pārlūkprogrammām[8].

Rezultāti

Jaunu informācijas resursu izstrādes un uzturēšanas procesos svarīgākā loma ir programmēšanas līdzekļu, datu apmaiņas tehnoloģiju un uzturēšanas risinājumu izvēlē. Attīstoties jaunām tehnoloģijām, izstrādātājiem ir pieejams arvien plašāks instrumentu loks savu risinājumu izstrādei[9,10].

Programmēšanas komponenšu un uzturēšanas rīku un metožu izvēli ietekmē sekojošie faktori:

- ✓ programmatūras izstrādes rīku izmaksas;
- ✓ izvēlēto tehnoloģiju savietojamības iespējas ar klienta vides aplikācijām un citām informācijas sistēmām un tehnoloģijām;
- ✓ tehnoloģiju integrācijas un datu indeksēšanas iespējas, apvienojot vairākas sistēmas vienā;
- ✓ izstrādāto risinājumu uzturēšanas izmaksas.

Analizējot pētījumā aprakstītās tehnoloģijas, var secināt, ka uz atvērtā koda pamata izstrādātās tehnoloģijas (HTML 5, REST) nodrošina plašākas programmēšanas iespējas izstrādātājiem, kā arī to integrācija un uzturēšana ir salīdzinoši vienkāršāka, nekā maksas analogu programmatūra. Atvērtā koda programmatūra nodrošina izstrādātājiem uzlabot un papildināt programmas kodu, katram risinājumam individuāli, tādā veidā paplašinot tā funkcionālās iespējas.

Kopsavilkums

Pētījumā ir aprakstītas datu apmaiņas arhitektūru un multimediju tehnoloģiju pielietojuma iespējas elektroniskās komercijas sistēmu izstrādes procesā. Tika veikta tehnoloģiju salīdzinājuma analīze, aprakstītas tehnoloģiju priekšrocības un trūkumi. Apkopota tehnoloģiju izmantošanas statistika un attīstības iespējas.

Darba nobeigumā ir aprakstīti galvenie faktori, kas ietekmē aprakstīto tehnoloģiju un risinājumu izvēli elektronisko informācijas sistēmu izstrādes procesā.

Bibliogrāfija

1. Hallan A. “Difference between static, dynamic and interactive content”, 2015 *(http://credible-content.com/blog/difference-static-dynamic-interactive-content/), Resurss pārbaudīts 16.04.2018.*
2. Awery Flight & Cargo Management Software, online documentation, 2010, *(http://awery.net/pages/software.html), Resurss pārbaudīts 16.04.2018.*
3. Zozol N. “RESTful Web Services”, 2008 *(https://nicolas-zozol.developpez.com/tutorial/java/rest-jsp-english/), Resurss pārbaudīts 16.04.2018.*
4. “SOAP vs REST Web Services”, 2011 *(https://www.javatpoint.com/soap-vs-rest-web-services), Resurss pārbaudīts 19.04.2018.*
5. “How do I use web service?”, 2017 (*https://www.quora.com/How-do-I-use-web-service-1*), *Resurss pārbaudīts 16.04.2018.*
6. “Advantages & Disadvantages of Webservices”, November 2007, (*https://social.msdn.microsoft.com/Forums/en-US/435f43a9-ee17-4700-8c9d-d9c3ba57b5ef/advantages-disadvantages-of-webservices?forum=asmxandxml*), *Resurss pārbaudīts 16.04.2018.*
7. “Flash & Future of Interactive Content”, Adobe Corporation, 2017 *(https://theblog.adobe.com/adobe-flash-update/), Resurss pārbaudīts 17.04.2018.*
8. “Adobe Flash is officially dead… how do you feel?”, 2015 *(https://www.sitepoint.com/community/t/adobe-flash-is-officially-dead-how-do-you-feel/208938), Resurss pārbaudīts 17.04.2018.*
9. Koranne N. “HTML5 vs Flash:The technical perspective”, 2017 *(https://www.chetu.com/blogs/technical-perspectives/html5-vs-flash.php), Resurss pārbaudīts 17.04.2018.*
10. Lublinsky B., “Is REST the future for SOA?”, 2011, *(https://www.infoq.com/articles/RESTSOAFuture), Resurss pārbaudīts 17.04.2018.*