LATVIJAS UNIVERSITĀTE

DATORIKAS FAKULTĀTE

MĀKOŅTESTĒŠANAS RISINĀJUMI

BAKALAURA DARBS

Autore: Sabīna Kupča

Studenta apliecības Nr.: sk11311

Darba vadītājs: profesors Dr.dat. Guntis Arnicāns

RĪGA 2015

ANOTĀCIJA

Bakalaura darbs ir veltīts mākoņtestēšanas risinājumu izpētei un to pielietošanai reāla projekta ietvaros bakalaura darba autores darba vietā.

Darbs sastāv no četrām daļām. Pirmajā daļā ir aprakstīta mākoņtestēšana un ap to saistītie objekti. Otrajā daļā ir aprakstīti populārākie mākoņtestēšanas rīki tīmekļa lietotnēm. Trešajā daļā ir aprakstīti trīs praktiski izmēģināti mākoņtestēšanas risinājumi tīmekļa lietotnēm un šo rīku salīdzinājums. Ceturtajā daļā ir aprakstīti vairāki mākoņtestēšanas rīki mobilajām ierīcēm un trīs praktiski izmēģināti mākoņtestēšanas risinājumi, kā arī veikts šo rīku salīdzinājums.

Darba nobeigumā ir aprakstīti autores iegūtie rezultāti, un pieredze mākoņtestēšanas ieviešanā gan tīmekļa bāzētā lietotnē, gan mobilajo ierīcu lietojumprogrammās.

Atslēgvārdi: Mākoņtestēšana, mākoņtestēšanas risinājumi, mākoņtestēšanas rīki, mākoņdatošana.

ABSTRACT

Cloud testing solutions.

Bachelors work is devoted for cloud testing solutions and exploring the usage of it and implementing it on a real project in bachelor author's workplace.

This work consists of four parts. The first chapter describes cloud testing and everything that is connected with it. The second chapter describes most popular cloud testing tools designed for web applications. The third chapter describes three practically tested cloud testing solutions for web applications and taking conclusions. The fourth chapter describes several cloud testing tools for mobile devices and three practically tested cloud testing solutions for mobile devices and taking conclusions.

At the end of this work author's experience of web-based application as well as mobile device applications and acquired results are described.

Keywords: Cloud testing, cloud testing solutions, cloud testing tools, cloud computing.

SATURA RĀDĪTĀJS

APZĪMĒJUMU SARAKSTS

Apzīmējums	Skaidrojums
Mākoņdatošana	Datu apstrādes tehnoloģija, kurā programmatūrai piekļūst internetā. Lietotājam ir piekļuve saviem datiem, bet viņam nav jārupējas par infrastruktūru, operētājsistēmu un personisko programmatūru.
Kapitāla izdevumi (*Capital expenditures (CAPEX)*)	Kapitāla izdevumi rodas, kad bizness tērē naudu, vai nu, lai nopirktu pamatlīdzekļus vai pievienot vērtību esoša pamatlīdzekļa ar lietderīgās lietošanas plešas ārpus taksācijas gada.
Mākoņtestēšana	Programmatūras testēšana, kurā tīmekļa lietojumprogrammas izmanto mākoņdatošanas vidi, lai simulētu reālā laika lietotāju darbību.
Lietojumprogrammas saskarne (*API*)	Tas ir protokolu, lietojumprogrammu izveidošanas rīku kopums. Lietojumprogrammas saskarne definē neatkarīgu no attiecīgas implementācijas funkcionalitāti, kas ļauj definīcijas un implementācijas variāciju vienam otru neaiztiekot.
AJAX (*asynchronous JavaScript and XML*)	Grupa ar savstarpēji saistītām tīmekļa izstrādes tehnikām, ko izmanto klienta puse, lai izveidotu asinhronas tīmekļa lietojumprogrammas.
HTTP	Lietotnes protokols, kas ir pamats datu pārraidei globālajā tīmeklī.
HTTPS	Lietotnes protokols, kas ir pamats drošai datu

	pārraidei globālajā tīmeklī.
Apache JMeter	Lietotne, kas paredzēta veiktspējas, u.c. testu izpildei.
Selenium	Pārvietojams programmatūras testēšanas ietvars tīmekļa lietojumprogrammām.
XML (*Extensible Markup Language*)	Teksta formatēšanas valoda, kas paredzēta, lai no neapstrādāta teksta, tajā ievietojot procedurālas un aprakstošas iezīmes, iegūtu strukturētus dokumentus.
SQL	Strukturēta vaicājumvaloda. Firmas IBM izstrādāta valoda, ko lieto datu bāzes pārvaldības sistēmās dažāda tipa datoros.
Daudzpakāpju cenas modelis (*Tiered pricing model*)	Nosaka cenu par vienu vienību.
HP veiktspējas centrs	Vairāku komponenšu apvienojums, lai atvieglotu LoadRunner resursu apmaiņu starp daudziem lietotājiem

IEVADS

Mākoņtestēšana ir jauna, bet ļoti aktuāla un daudzsološa joma mākoņdatošanā. Vairāki pēdējo 4 gadu pētījumi [3-7] norāda, ka testēšanai ir plaša tendence migrēt uz mākoni. Tādas vadošās kompānijas, kā IBM, Microsoft, Google un Amazon ir ieinteresētas vārdā "mākonis". Mākoņdatošana ir sasniegusi vērā ņemamu interesi pēdējos gados kā jauna izstrādes, lietotņu un servisu piegādes paradigma. Tā ietekmē visus programmatūras dzīves cikla posmus, tajā skaitā arī programmatūras testēšanu [8], tāpēc balakaura darba autorei šķita interesanti to izzināt.

Mākoņtestēšanas servisu pakalpojumi piedāvā risinājumus gan tīmekļa lietotnēm, gan mobilo ierīču lietojumprogrammām, kas atvieglos autores darba vietā izstrādāto lietotņu testēšanu.

Darba mērķis ir iepazīties ar mākoņa testēšanas pieejamajiem servisiem un pārbaudīt, cik labi tie ir reālā programmatūras testēšanā.

Lai sasniegtu izvirzītos mērķus, tiek izvirzīti vairāki uzdevumi:

1. Izzināt mākoņtestēšanas teoriju;
2. Apskatīt kā mākoņtestēšana tiek izzināta pētniecībā;
3. Izzināt kādi risinājumi tiek piedāvāti zinātniskajos darbos;
4. Izpētīt kādas iespējas sniedz tīmeklī pieejamie mākoņtestēšanas risinājumi - gan tīmekļa lietojumprogrammām, gan mobilajām ierīcēm;
5. Sešu mākoņtestēšanas rīku praktiska izpēte un secinājumu veikšana par tiem;

Darbs sastāv no četrām daļām:

- Pirmajā daļā tiek raksturota mākoņtestēšana, kas ietver priekšrocības, trūkumus, mākoņbāzētas programmatūras testēšanas modeļus un citas raksturīgākās iezīmes;
- Otrajā daļā tiek vispārīgi aprakstīti tīmeklī pieejamie mākoņtestēšanas rīki;
- Trešajā daļā autore praktiski pielieto trīs mākoņtestēšanas rīkus, kas veic funkcionālos testus un veiktspējas analīzi. Autore veic secinājumus un novērtē rīku priekšrocības, kā arī izvērtē labāko;

- Ceturtajā daļā praktiskitiek pielietoti trīs mākoņtestēšanas rīki mobilajām ierīcēm, kas atvieglo testēšanu un ko autore apskata detalizētāk konkrētas aplikācijas testēšanā un veic novērtējumu, cik šie rīki ir bijuši lietderīgi reālā projektā testējot lietotni;

Darba aktualitāte ir saistīta ar lielo mobilo iekārtu un operētājsistēmu versiju skaitu un to testpiemēru veikšanu uz populārākajām ierīcēm, kā arī ar to, ka tīmekļa lietotņu testēšanā ir nepieciešamība pēc ticamiem veiktspējas, slodzes testiem un lietotāju pieslēguma vietas simulācijas.

1. MĀKOŅTESTĒŠANAS RAKSTUROJUMS

1.1. Mākoņdatošana

Mākoņdatošana izmanto interneta pieslēgumu, lai piekļūtu programmatūras lietojumprogrammām un datiem, kas tiek glabāti uz serveriem un ir no attāluma pārvaldāmi. Mākoņdatošana ir saistīta un atkarīga no virtualizācijas, interneta pieslēguma, skaitļojamām mašīnām, u.c.

Mākoņdatošana ļauj klientiem un uzņēmumiem izmantot programmatūru, kuru nav nepieciešams instalēt vai lokāli glabāt failus savos datoros, līdz ar to tiem var piekļūt un tos var izmantot no jebkura datora ar interneta pieslēgumu. [2]

Mākoņdatošanai ir raksturīgi [9]:

- darbošanās uz virtuālas aparatūras un programmatūras;
- Var piekļūt izmantojot standarta tīkla protokolus;
- Nodrošināta lietojumprogrammas saskarne (*API*) integrācijai un vadībai;
- Izmanto trešās puses lietojumprogrammas saskarnes(*API*) , kā lietotāju autentificēšana(facebook), datu glabāšana(DropBox, Google Drive), ziņojumi(gmail), u.c.;
- Mēdz neizmantot SQL datu glabātuves - galvenokārt lielu, nestrukturētu datu pārvaldībai [9].

1.2. Mākoņtestēšana

Mākoņtestēšana programmatūras testēšanai izmanto mākoņa infrastruktūru. Tas piedāvā efektīvu, neierobežotu glabāšanu, ātri pieejamu un mērogojamu infrastruktūru, testēšanas vides pielāgojamību un pieejamību, kas līdz ar to samazina lielu lietojumprogrammu izpildes laiku un rada rentablus risinājumus. [1]

Mākoņtestēšana izmanto mākoņdatošanas resursus (aparatūra, lietojumprogrammatūra, infrastruktūra), kas nepieciešama testēšanas vajadzībām un testu izpildei.[2]

Mākoņtestēšanas vide dod testēšanas komandām lielāku kontroli testu izveidošanai un izpildei, veiktspējas analīzei un sastrēgumu meklēšanai kamēr testpiemēri izpildās. Mākonis ļauj testētājiem apskatīt jaudas robežvērtības, ja skalā no tūkstošiem līdz miljoniem vienlaicīgiem lietotājiem slēdzas klāt. Tas dod testētājiem skaidrāku priekšstatu par iespējamo darbības kļūdu, kas samazina kļūdas produkcijā un labāk sagatavo uzņēmumus maksimālā pieprasījuma reizēm. Tas viss ir ļoti svarīgi, lai uzņēmums būtu spējīgs piedāvāt labu produktu pašlaik pastāvošajā konkurences tirgū.

Mākoņtestēšanai ir jāatrod optimāls pielietojums dažādos projektos un tai ir izmaksas un riski, kas jāņem vērā un kas tiks turpmāk darbā izvērtēti.

1.3. Pētniecība mākoņtestēšanā

Mākoņtestēšana ir ātri augoša programmatūras inženierijas pētniecības joma. Pirmie mākoņtestēšanas pētniecības darbi parādījās četrus, piecus gadus atpakaļ. Divi specializēti semināri Programmatūras testēšana mākonī (*Software Testing in the Cloud (STITC)*) tika organizētas 2009., 2010. gadā. Taču vēl ir pāragri runāt par nozīmīgiem panākumiem vai fundamentāliem sasniegumiem mākoņtestēšanā. Pētījumi par mākoņtestēšanu zināmā mērā atpaliek no praktiskajiem rezultātiem šajā jomā [8].

2010. gada otrajā internacionālajā mākoņtehnoloģiju un zinātnes conference [10] notika visaptveroša mākoņtestēšanas diskusija starp tās dažādu organizāciju dalībniekiem. Tās gaitā rezultāti tika apkopoti un atspoguļo praktiskās vajadzības un cerības mākoņtestēšanai.

Saskaņā ar konferencē nolemto, programmatūra ir piemērota mākoņtestēšanai, ja:

- Testpiemēri ir neatkarīgi viens no otra;
- Programmatiski pieejama saskarne, kas ir piemērojama automatizētu testu veikšanai

Pamatojoties uz šiem apsvērumiem, autori secina, ka vispiemērotākie testēšanas veidi mākoņtestēšanai ir:

- Vienībtestēšana un regrestesti;
- Liela apjoma automatizētie testi;
- Veiktspējas un slodzes testi;

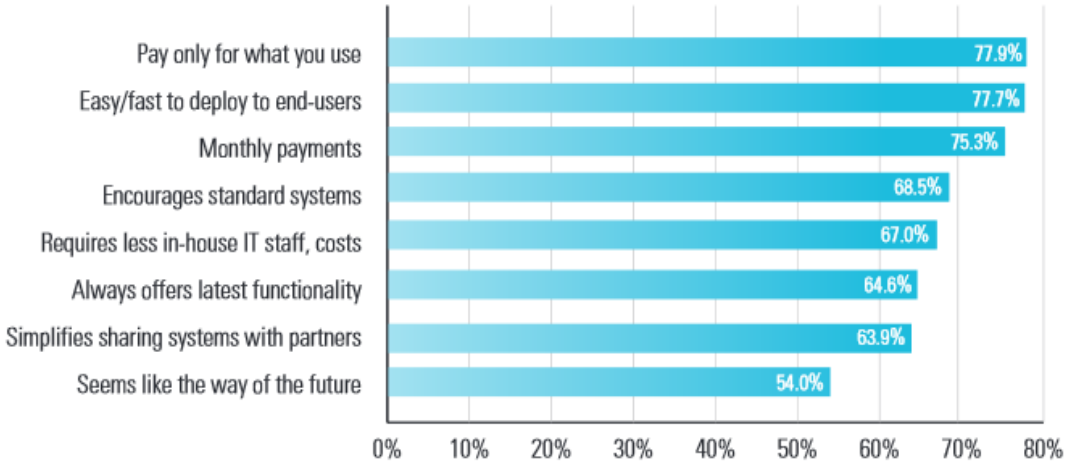
Ņemot vērā šos testēšanas veidus, tālākajā darbā tie tiks apskatīti sīkāk.

Mākoņtestēšanas priekšrocības

Mākoņdatošana paver plašas un jaunas testēšanas iespējas. Testēšana salīdzinoši ar citiem izstrādes procesiem prasa vislielākos resursus, tajā skaitā arī infrastruktūru. Mākoņtestēšanai ir liels potenciāls samazināt nepieciešamos līdzekļus, kas ietver gan izmaksas, gan resursus.

Mākoņtestēšanai ir daudz priekšrocību, no kurām nozīmīgākās ir:

- Mākoņtestēšana ļauj vieglāk un efektīvāk simulēt lietotāju pieslēgšanos no dažādām ģeogrāfiskām vietām, testēt ugunsmūri, izpildīt slodzes testus, aparatūras un programmatūras noslodzi. [1]
- Viegli pieejams – pakalpojums ir neatkarīgs no lietotāja ierīces un atrašanās vietas, lietotāji var piekļūt datiem neskatoties uz izmantoto ierīci, operētājsistēmu vai atrašanās vietu.
- Aparatūra nav nepieciešama - Tā kā testpiemēri un vide glabājas mākonī, fiziski datu centri nav nepieciešami, taču par rezerves kopijām tomēr nevajadzētu aizmirst.
- Izmaksas - Izmantojot mākoņbāzētus pakalpojumus, novērš nepieciešamību uzstādīt un uzturēt aparatūru, infrastruktūru. Tā vietā samaksa ir par laiku, cik ilgi tiek izmantota attiecīgā lietotne vai pakalpojums, tajā skaitā pēdējā izlaiduma versijas (piemēram, operētājsistēma) ir ātri pieejamas.
- Laiks - kas nepieciešams, lai veiktu pilnīgus testus uz visām ierīcēm un aparatūras.
- Mobilām aplikācijām ir lielāka mobilo iekārtu izvēle - Jebkurš testētājs var izvēlēties mobilās iekārtas no mobilo iekārtu laboratorijas nevis izmantot fiziskas iekārtas.
- Uzticamība – dati atrodas vairākos datu centros, tādēļ to pazušana ir praktiski neiespējama, tāpat mainot kādas datnes saturu vai to dzēšot, parasti tiek veidotas arī rezerves kopijas.



1.1. att. **Mākoņtestēšanas pakalpojumu izmaksu ietaupījums**

Izpētot vairāku mākoņtestēšanas pakalpojumu sniedzēju sniegto informāciju par pagājušo 2014. gadu, izmaksu ietaupījums svārstās starp 40% - 70% (skatīt 1.1. att.). Neliela un vidēja lieluma uzņēmumi, kas nevar atļauties kapitāla izdevumus, redz mākoņtestēšanu kā risinājumu, jo nav nepieciešams investēt infrastruktūrā, programmatūras licencēs, konfigurēšanā un testēšanas vides uzturēšanā. Līdz ar to tiek maksāts tikai par patērēto [11].

1.4. Mākoņtestēšanas trūkumi

Neskatoties uz to, ka mākoņtestēšanai ir dažādas priekšrocības, tajā pašā laikā mākoņtestēšanai jārēķinās ar jauniem izaicinājumiem un ar tiem saistītajiem riskiem, piemēram, tādiem kā:

- Datu drošība - kā jau lielākoties visām lietotnēm, kas izmanto interneta pieslēgumu, ir jārēķinās ar drošības draudiem un riskiem. Drošība publiskajā mākonī joprojām rada lielas bažas un šobrīd pieejamās šifrēšanas metodes tiek uzskatītas par nepietiekamām. Tiek izstrādātas procedūras, lai uzlabotu drošību un veiktspēju publiskajā makonī. Piemēram, pakalpojumu sniedzēji attīsta virtuālos privātos mākoņpakalpojumus un klientu sadaļu.
- Interneta pieslēgums un tā ātrums – lietotājam, zaudējot pieeju interneta pieslēgumam, zūd pieeja arī izmantot jebkādas mākoņlietotnes. Arī lēns interneta ātrums stipri apgrūtina testēšanu.

- Standartu trūkums - Šobrīd nav neviena standartizēta risinājuma, lai integrētu mākoņa resursus ar uzņēmuma iekšējiem datu centra resursiem. Mākoņpakalpojumu sniedzējiem ir izveidota sava arhitektūra, darbības modeļi un cenu noteikšanas mehānismi, kas piedāvā ļoti maz spēju sadarboties.
- Infrastruktūra - Daži mākoņpakalpojumu sniedzēji piedāvā tikai ierobežotus konfigurācijas veidus, serverus un uzglabāšanu, tīklu un joslas platumu, padarot grūti izveidojamu reāllaika testa vidi.
- Veiktspēja - Mākoņapakalpojumu izmantošanas gadījumā nevar izslēgt traucējumus, kas var rasties tīkla pārtrūkuma rezultātā.

1.5. Mākoņtestēšanas modeļi

Izvērtējot visus mākoņtestēšanas ieguvumus un trūkumus ir iespējams izvēlēties pakalpojums sniedzēju pēc tā modeļa. Programmatūras testēšana mākonī pamatā tiek iedalīta trīs dažādos modeļos:

1. Infrastruktūra kā serviss (*IaaS*) - Infrastruktūra kā serviss nodrošina datora infrastruktūru - kas parasti ir virtualizācijas vides platforma. Tā vietā, lai iegādātos serverus, programmatūru, datu centru vai tīkla aprīkojumu, klients šādus resursus iegādājas no ārējā pakalpojumu sniedzēja. IaaS ir veids, kā nogādāt mākoņpakalpojuma infrastruktūru - serverus, datu glabāšanu, ugunsmūri, IP adreses, tīklu, operētājsistēmas. Atkarībā no mākoņpakalpojumu sniedzēja, komerciālā modeļa, pieprasījuma un noslodzes, šie resursi var būt ļoti elastīgi un pielāgojami [12].

   Piemēram: Windows Azure, Amazon, Google Compuer Engine, u.c.

2. Platforma kā serviss (*PaaS*) - Platforma kā pakalpojums ir mākoņskaitļošanas forma, kas papildus infrastruktūrai nodrošina arī operētājsistēmu, datu bāzi, tīmekļa serveri un programmatūras izstrādes vidi, kur klients var veidot savas lietojumprogrammas [12].

   Piemēram: Windows Azure, Heroku, Google App Engine.

3. Programmatūra kā pakalpojums (*SaaS*) - Šajā gadījumā mākoņpakalpojumu sniedzējs piedāvā pilnu lietojumprogrammu gala lietotājiem. Programmatūra tiek glabāta mākonī un piegādāta izmantojot pārlūkprogrammu, kur nav jāuztraucas par programmatūras instalāciju un iestatīšanu [12].

Piemēram: Vairums mākoņtestēšanas risinājumu, kas ir turpmāk darbā apskatīti - Soasta CloudTest, Loadstorm, Itko Lisa, HP LoadRunner, Blaze Meter, Load Impact, AppThwack, Testdroid, Perfecto Mobile, u.c.

1.6. Mākoņa veidi

Ir trīs veidu mākoņa platformas: Publiskais, privātais un hibrīda. Katram no šiem veidiem ir savas priekšrocības un trūkumi [13].

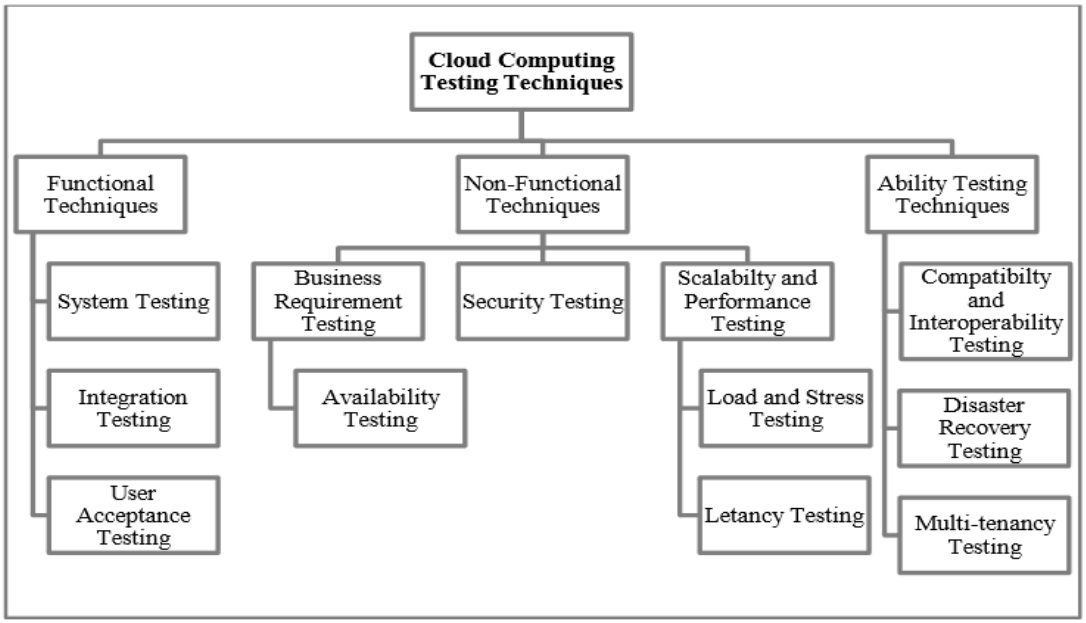
- **Publiskais mākonis –** pakalpojumi ir pieejami ikvienam un resursi tiek dinamiski piešķirti pēc pieprasījuma [14]. Visa infrastruktūra glabājas un tiek pārvaldīta mākoņpakalpojuma sniedzēja pusē. Klientam nav pieejas infrastruktūrai. Tā tiek dalīta starp vairākiem lietotājim, visbiežāk izmantojot daudzpakāpju cenas modeli (*tiered pricing model*).

- **Privātais mākonis –** parasti pārvalda konkrētas organizācijas ugunsmūra noteikumi un pieeja tiek nodrošināta tikai organizācijas lietotājiem. Līdz ar to infrastruktūra glabājas un tiek pārvaldīta konkrētas organizācijas pusē [14].

- **Hibrīda mākonis -** ir privātā un publiskā mākoņa apvienojums. Organizācijas var izlemt, kādus pakalpojumus viņi vēlas piedāvāt ikvienam un kādus pakalpojumus piedāvāt tikai organizācijas lietotājiem [14].

1.7. Mākoņtestēšanas veidi

Mākoņtestēšanai nav jānodrošina tikai tas, vai funkcionālās prasības tiek izpildītas, bet spēcīgs uzsvars tiek likts uz nefunkcionālo testēšanu. Visbiežāk, apkopojot vairāku resursu rekomendācijas, tiek izpildīti šādi testēšanas veidi:



1.2. att. **Mākoņtestēšanas veidi [15]**

1.7.1. Funkcionālā testēšana

Funkcionālie testi nodrošina to, ka biznesa prasības tiek izpildītas. Tā analizē tikai sistēmas atbildes reakciju uz noteiktām ieejas iedarbībām, bet ignorē tās iekšējo struktūru. Funkcionālie testi ir jāveic, lai pārliecinātos, ka piedāvājums nodrošina visus servisus, kas ir paredzēti un ko lietotājs sagaida. [13]

Daži funkcionālo testu veidi:

Sistēmas pārbaude *(System Verification Testing)* - Tas nodrošina, ka vairāki moduļi savā starpā nekonfliktē un funkcionē pareizi, kā paredzēts.

Akceptestēšana *(Acceptance Testing* - Programmatūras vai aparatūras galīgā pārbaude, kas parāda izstrādātā produkta funkcionālās iespējas. Šī pārbaude nosaka, vai pārbaudāmais objekts darbojas atbilstoši specifikācijai, t.i., vai tas atbilst akceptēšanas kritērijiem.

Sadarbspējas testēšana *(Interoperability Testing)* - Jebkurai lietotnei jāspēj darboties ne tikai uz dažādām platformām, bet tai jāstrādā nevainojami pārvietojot to no mākoņa infrastruktūras uz citu.

1.7.2. Nefunkcionālā testēšana

Nefunkcionālie testi lielākoties koncentrējas uz tīmekļa lietojumprogrammu testēšanu nodrošinot vēlamās prasības.

Daži nefunkcionālo testu veidi:

Pieejamības pārbaude *(Availability Testing)* - Visbiežāk lietotne tiek darbināta noteiktu laika periodu, kura laikā tiek apkopoti lietotnes neveiksmes gadījumi. Ir arī gadījumi, kad vietne tiek ielādēta ļoti lēni. Risinājums katrā gadījumā var būt atšķirīgs.

Veiktspējas un slodzes testēšana *(Performance and Load Testing)* - Ar unikālām mākoņa vides īpašībām ir iespējams veikt precīzus veiktspējas un slodzes testus. Kā veiktspējas kvantitatīvie kritēriji parasti tiek izmantoti atbildes laiks, caurlaidspēja un izmantojamība. Tīkla latentums arī ir viens no kritiskajiem faktoriem veiktspējas novērtēšanai - kas notiek, ja slodze tiek palielināta vai samazināta.

Drošības testēšana *(Security testing)* - Mākoņa platforma un lietotnes ir pakļautas vairākām drošības ievainojamībām. Svarīga drošības problēma ir sķērsojošā (traversal) ievainojamība. Tas nozīmē to, ka viens lietotājs ir spējīgs pāriet no vienas virtuālās mašīnas klienta vides uz citu klienta vidi tajā pašā laikā tiekot viena pārvaldnieka pārvaldīts. Īsāk sakot, tas ļauj klientam piekļūt citu lietotāju aplikāciju virtuālajām instancēm. Vairāku lietotāju vidē ir svarīgi modelēt ļaunprātīga lietotāja darbības un testēt vairākas iespējamās ievainojabības, tādas kā SQL injekcijas, starpvietņu skriptošana, u.c.

Citas drošības vietas testēšana iekļauj lietotāju lomu piekļuve atsevišķām vietām, piekļuves tiesības, u.c. [16]

Datorkļūmes novēršanas testēšana *(Disaster recovery Testing)* - Pieejamības pārbaudē mākonim ir jābūt pieejamam visu laiku un nav pieļaujami tīkla pārrāvumi, sistēmas kļūmes slodzes dēļ, utt.

1.8. Mākoņtestēšanas rīki

Šajā sadaļā tiks minēti dažādi rīki, kas tiek izmantoti dažādiem testēšanas veidiem mākonī. Vairāki bezmaksas un maksas mākoņtestēšanas rīki ir pieejami tīmeklī. Tādi, kas ietver gan platformas, gan aparatūras saskarni, datu glabāšanas sistēmu kā arī lietotnes sistēmu.

1.8.1. Rīki veiktspējas testēšanai mākonī

Vairāki rīki tiek izmantoti veikstspējas un slodzes testēšanai. Daži no zemāk minētajiem rīkiem var tikt izmantoti arī funkcionālaiem testiem:

- SOASTA CloudTest;
- LoadStorm;
- CloudTestGo;
- Cloudslueth;
- Itko Lisa;
- Load Impact [8];
- CloudTestGo by CSS [8];
- IBM Testing Services for Cloud ;
- IBM Rational Load Testing on the IBM Cloud;

1.8.2. Rīki drošības testēšanai mākonī

Visbiežāk izmantotie rīki drošības testēšanai:

- Nessus;
- Wireshark;
- Nmap;

2. MĀKOŅTESTĒŠANAS RĪKI TĪMEKĻA LIETOTNĒM

Organizācijas, kas veic dažāda veida testēšanu - slodzes, veiktspējas testus, produkcijas servisa monitoringu, u.c., parasti saskaras ar dažādām problēmām, piemēram, ierobežotu budžetu, lielu daudzumu ar testpiemēriem, nespēju atkārtot tos, kā arī izaicinājums ir lietotāju pieslēguma vietas simulācija no dažādām pasaules vietām.[1]

2.1. Soasta Cloud test

Soasta CloudTest ir mākoņbāzēts veiktspējas testēšanas rīks tīmekļa lietotnēm. Tas spēj simulēt tūkstošiem virtuālu lietotāju. Tas ir publiskā tipa mākoņtestēšanas serviss, līdz ar to pakalpojums ir pieejams ikvienam un resursi tiek dinamiski piešķirti pēc pieprasījuma.

Nākamajā sadaļā šis rīks tiek apskatīts sīkāk.

2.2. Loadstorm

LoadStorm ir mākoņbāzēts rīks slodzes testu veikšanai. Pamatā tāpat kā minētais Soasta CludTest, tā arī šis rīks piedāvā vairāku virtuālo lietotāju darbošanās simulāciju veicot norādītu darbību kopumu.

Loadstorm piedāvā veikt testu ierakstīšanu, ierakstīto testu glabāšanu un rezultātu analīzi.

Nākamajā sadaļā šis rīks tiek apskatīts sīkāk.

2.3. Itko Lisa

Itko Lisa piedāvā dāžādu testu veikšanu mākonī:

- funkcionālo testu veikšanu mākonī dinamiskām tīmekļa lapām.
- saskarnes testēšanu
- slodzes un veiktspējas testēšana

Itko Lisa bakalaura darba ietvaros netiks apskatīts, bet izpētot dažādos resursos publicēto informāciju, tas ir produktu kompleks, lai uzlabotu lietojumprogrammas izstrādes komandas efektivitāti, īpaši, kas ir iesaistīti pielāgojamās lietojumprogrammās un mākoņdatošanā.

Itko Lisa mērķis ir nodrošināt mākoņbāzētu vidi un saliktas lietojumprogrammas izstrādes virtuālos servisus. Zinātniskie raksti apgalvo, ka programmatūras piegādes laiks samazinās par 30%, izmantojot savu inovatīvo pieeju, lai atbalstītu nepārtrauktu izstrādes un testēšanas integrāciju [16].

2.4. HP LoadRunner

HP LoadRunner ir vēl viens automatizēto veiktspējas testu veikšanas produkts, kas paredzēts lietotnes slodzes testu veikšanai. HP LoadRunner darbības pamatā ir virtuālie lietotāji, kas simulē reālo lietotāju darbības, izmanto protokolu HTTP un sūta pieprasījumus IIS vai Apache web serverim, līdz ar to simulējot tūkstošiem vienlaicīgu lietotāju pieslēgumus.

HP LoadRunner ir lejuplādējama un lokāli uzstādāma lietotne, kas sazinās ar datubāzi izmantojot interneta savienojumu, kurā tiek glabāta visa lietotāja informācija. Kā otra iespēja ir būt daļai no HP Veiktspējas Centra [16].

2.5. Blaze Meter

Blaze Meter piedāvā slodzes un veiktspējas testēšanas platformu pielāgotu programmatūras izstrādātājiem. Tā ir aprīkota ar augstu skriptu iespējām piesaistīt JMeter slodzes testiem un Selenium rīku lietotāju piedzīvojuma(user-experience) testiem. Tāpat kā iepriekš apskatītiem rīkiem, arī Blaze Meter atbalsta slodzes testus ar vairāk kā 1,000,000 virtuālajiem lietotājiem. Lietotaji var būt publiskā mākoņa vai arī privātā tipa jeb iekš korporatīvā ugunsmūra, ļaujot klientam atrast un noteikt darbības vājās vietas pēc iespējas ātrāk.

Šis rīks nodrošina arī testpiemēru izveidi grafiskajā tīmekļa vidē, kur testus var pārvaldīt, arhivēt, glabāt, monitorēt, kā arī piedāvā atbalstu HTTP/S, xml, Sql, lietotāja pieteikšanos sistēmā, u.c. [16]

2.6. Rīku pārskats

2.1. tabula [16]

Rīks	Priekšrocības	Trūkumi	Izcenojums
Soasta CloudTest	Pilnīga vide testu uzstādīšanai. Labs testa lietotāju pārklājums pasaulē. Detalizēta tīmekļa vietnes analīze(rezultāti). Bezmaksas versijā pieejami 100 virtuālie lietotāji.	Lai palaistu testēšanas vidi, ir nepieciešama VM iestatīšana.	Izcenojums ir par servera stundām - nav limits testētāju skaitam, bet gan patērētajiem resursiem.
Loadstorm	Mērogojams līdz 300,000 lietotājiem. Testēšanas datu centrs atrodas Sidnejā, Austrālijā. Nav nepieciešana vides iestatīšana. Nav iesaistīts/nepieciešams skripts.	Nav iebūvētas funkcijas, lai attēlotu servera veikstspējas rādītājus.	Izcenojums ir pēc mēneša maksas(sākot no bezmaksas versijas, kur ir 100 lietotāji un 25 lietotāji vienam testam līdz "Uzzvaniet mums") vai pēc patērētajām stundām (sākot no $39(1,000 lietotāji testpiemēram)).
Itko Lisa	Iepriekš ieplānoti automatizētie regrestesti, funkcionālie un veiktspējas testi.	Paredzēts lielākoties izstrādes komandām, nevis testētājiem un kvalitātes analīzei.	Nelielai grupai piecu cilvēku sastāvā piekļuve sistēmai ar 250 virtuālajiem lietotājiem mēnesī izmaksā 45$.
HP LoadRunner	Teicams saskarnes monitorings un analīze. Grafiskā lietotāja saskarnes skripta ģenerēšana no ieraksta. HP atbalsts rīkam.	Pārmērīgi augstas izmaksas	0.56 $ par vienu virtuālā lietotāja dienu. Tātad, ja ir vēlme pēc 100 virtuālajiem lietotājiem, tad tas izmaksās 56 $ konkrētajā dienā. Nākamajā dienā summa par 100 lietotājiem atkal

			tiek pierēķināta.
Blaze Meter	Padevīgs savietojamībā ar JMeter. Viegli saprotama saskarne. Divu testa rezultātu salīdzinājums. Testpiemēru izpildes laika plānošana.	Pieejamas ierobežotā skaitā dažādas ģeogrāfiskas vietas.	Mēneša plāns - sākot no $199(1,000 lietotāji, 20h) Pēc pieprasījuma - sākot no $19(1,000 lietotāju, 1 serveris) līdz $299(40,000 lietotāji, 40 serveri)

Salīdzinot rīkus var redzēt, ka katram rīkam ir savas priekšrocības un trūkumi (skatīt 2.1. tabulu). Katrai komandai ir jāizvērtē tai piemērotākais rīks konkrētu testu veikšanai.

Autore no apskatītajiem ir izvēlējusies praksē pielietot divus no tiem – Soasta Cloud Test savas plašās funkcionalitātes dēļ un Loadstorm sava detalizētā rezutātu analīzes skata dēļ.

3. TĪMEKĻA LIETOTŅU MĀKOŅTESTĒŠANAS RĪKU PRAKTISKA APSKATE

Bakalaura darba izstrādes gaitā praktiski tika apskatīts rīks, kas veic funkcionālos testus un divi citi, kas veic veiktspējas analīzi, tajā skaitā slodzes testus.

Dāžadi testēšanas rīki tika pielietoti autores darba vietā, lai spētu novērtēt to pienesumu reālā izstrādes procesā. Tā kā bakalaura darba autore ikdienā testē gan tīmekļa lietotnes, gan mobilās lietojumprogrammas, tad reālu projektu ietvaros tika pielietoti dažādi mākoņtestēšanas rīki.

3.1. SOASTA Cloud Test

Bakalaura darba ietvaros tika apskatīta Soasta bezmaksas versija - CloudTest Lite.

Soasta ir Kalifornijā bāzēta tehnoloģiju kompānija, kas specializējas mākoņtestēšanā un tiek minēts, kā viens no daudzfunkcionālākajiem mākoņtestēšanas risinājumiem.

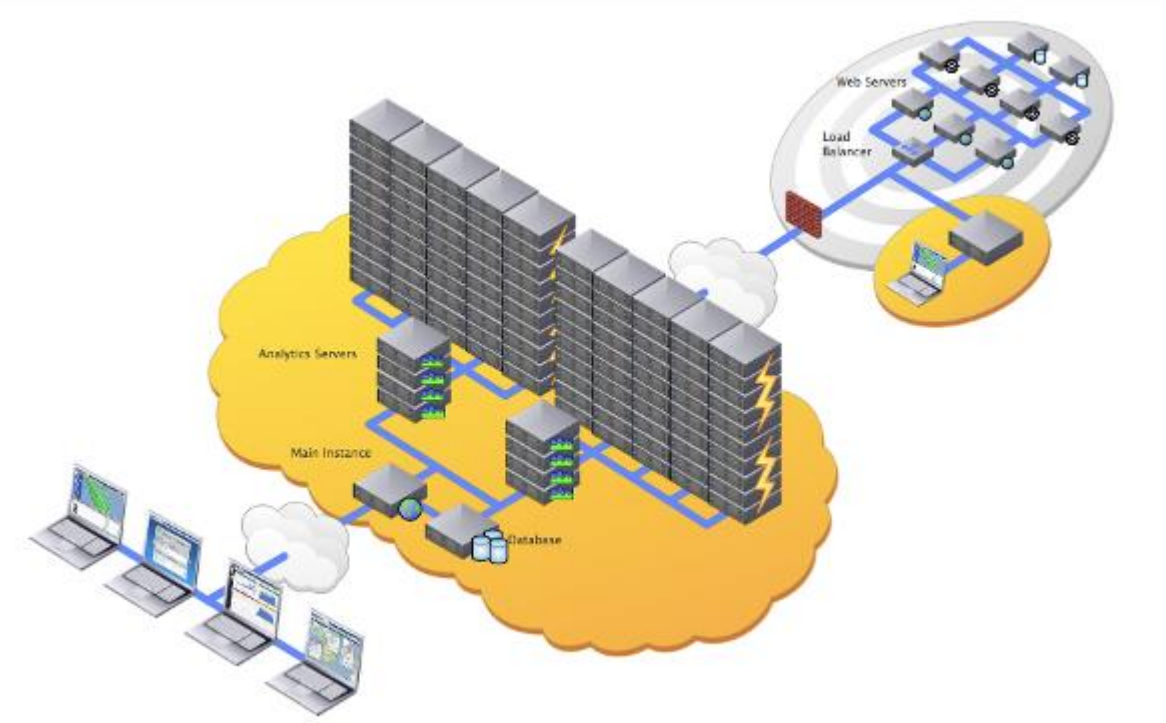
Soasta CloudTest Lite izmanto publiskā mākoņa platformu un visa darbošanās notiek pārlūkprogrammā.

Praktiskā pielietojumā tika apskatīta tīmekļa lapa. Soasta Cloud Test izmanto AJAX tīmeklī bāzētu lietotāju saskarni, kurā testētāji var vadīt un uzraudzīt visu procesu, tajā skaitā slodzes serveru darbības sākšanu, izveidot un palaist testa aģentus no dažādām ģeogrāfiskām vietām, kā arī analizēt atgrieztos testa rezultātus. Soasta CloudTest ir iespējams veikt veiktspējas, slodzes un funkcionālos testus tīmeklī bāzētām lapām - kā jau bezmaksas versijā, ir ierobežojumi, piemēram, slodzes testi tikai ar 100 virtuālajiem lietotājiem, u.c.

Vispasaules testēšanas mākonis nodrošina reālās pasaules sacensību dažādos līmeņos. T.i. Var tikt simulēta miljons lietotāju plūsma no dažādām vietām pasaulē. Vispasaules testēšanas mākonis ļauj starpmākoņu(cross-cloud) testēšanu no pasaules vadošajiem mākoņpakalpojumu sniedzējiem, tādiem kā Amazon, IBM un Microsoft.

3.1.1. Arhitektūra

Soasta CloudTest Lite mākoņapakalpojums testēšanai tiek izmantots pēc pieprasījuma attiecīgajā brīdī. Tās arhitektūra sastāv no Globālās mākoņtestēšanas platformas, kas nodrošina hibrīdmākoņa (cross-cloud) infrastruktūru slodzes radīšanai. Pašā pamatā svarīgas ir atvērtā koda bibliotēkas, kas caurvij visas funkcijas, ko piedāvā šis pakalpojums [17].



3.1. att. **Mākoņtestēšanas pakalpojumu izmaksu ietaupījums** [17]

Soasta CloudTest Lite piedāvā "Programmatūra kā pakalpojums" testēšanas modeli. Kā redzams 3.1. att., Soasta CloudTest Lite ir izvietots mākonī izmantojot dalītu arhitektūru, kas ir papildināta ar testēšanai paredzētām ierīcēm un atrodas aiz ugunsmūra.

3.1.2. Testu veidošana

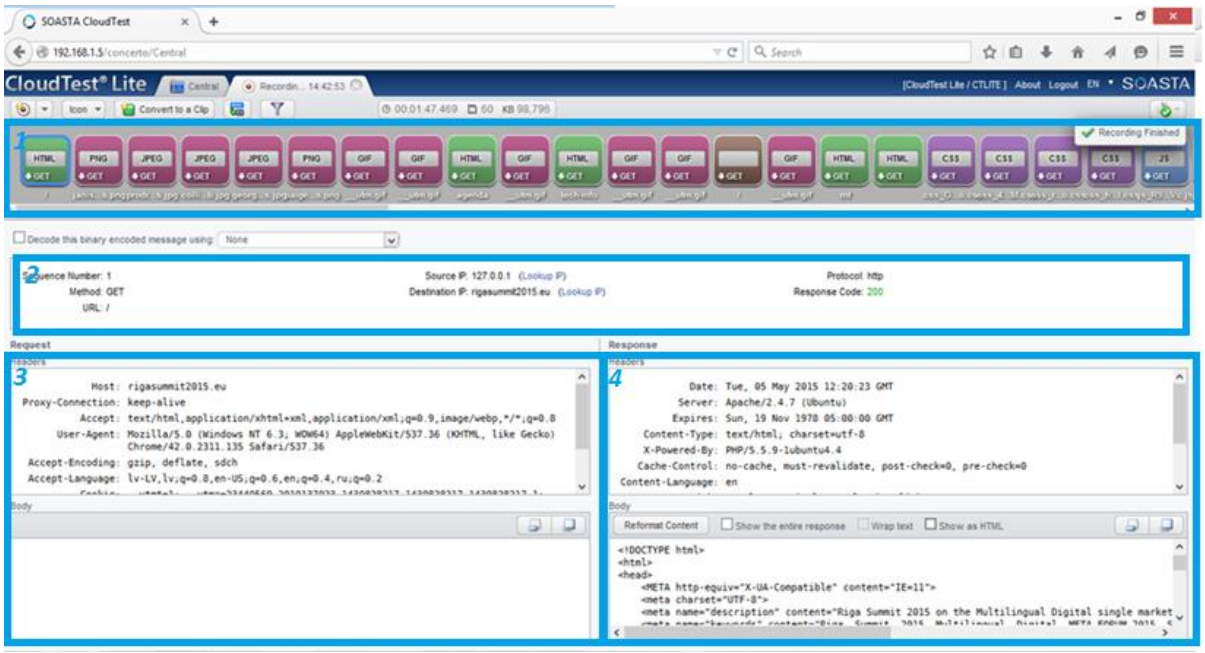
Visi veiktie testi notiek ieraksta veidā - CloudTest ieraksta visu inormāciju, kas iet caur portiem 80 un 443. Tādēļ autorei bija jāaizver visas citas tīmekļa lapas, kas veic asinhromizētus AJAX pieprasījumus, piemēram, kalendārs, kā arī epasta tīmekļa klients.

3.1.3. Veiktspējas testi

Sekojot labajai praksei ierakstā netika veikta visas tīmekļa lapas testēšana, bet tā tika sadalīta pa daļām un izveidoti vienībtesti.

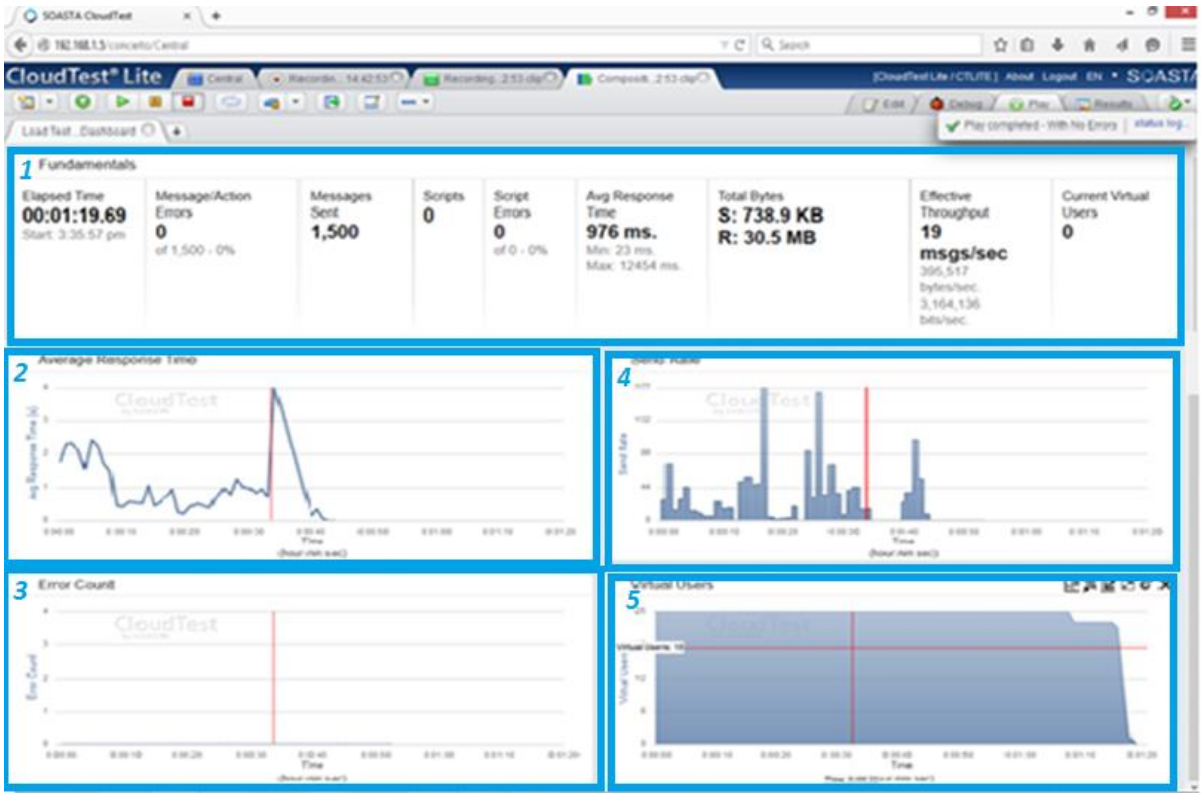
Testi tiek veidoti sastādot klipu, ar kuru turpmāk tiks veikti dažādi testi un tie nav jāatkārto manuāli. Klipu var rediģēt, pārsaukt testpiemērus, iestatīt taimeri un veikt citas dažādas darbības, kas atgriež testētājam noderīgu un vajadzīgu informāciju. Autores konkrētajā gadījumā vienībtesti sastāvēja no pāris darbībām, kā piemēram, tīmekļa vietnes atvēršanu konkrētā pārlūkā, rīkjoslas sadaļu atvēršana (pogu pārbaude), hipersaites pārbaude uz citu tīmekļa vietni, pārlūka aizvēršana.

Kad klips ir izveidots, tālāk to pārveido par kompozīciju, ar ko tālāk tiek veikti slodzes testi - mainīts virtuālo lietotāju skaits, kā arī to, cik reizes klips tiek atkārtots. Šajā gadījumā autore norāda 25 lietotājus un veiktspējas testu. Tāpat ir iespēja norādīt, no kuras vietas pasaulē pieslēgumi tiks veikti.



3.2. att. **Ieraksta rezultāts**

3.2. att. ataino izveidoto klipu. Saskarne ir izveidota intuitīvi saprotama – 3.2. att. nr. 1 attēlo pieprasījumu daudzumus un veidus, kas ir ierakstīti. 3.2. att. nr. 2 attēlo vispārējos tīmekļa vietnes parametrus, 3.2. att. nr. 3 attēlo pieprasījuma datus, 3.2. att. nr. 4 atgrieztos datus.



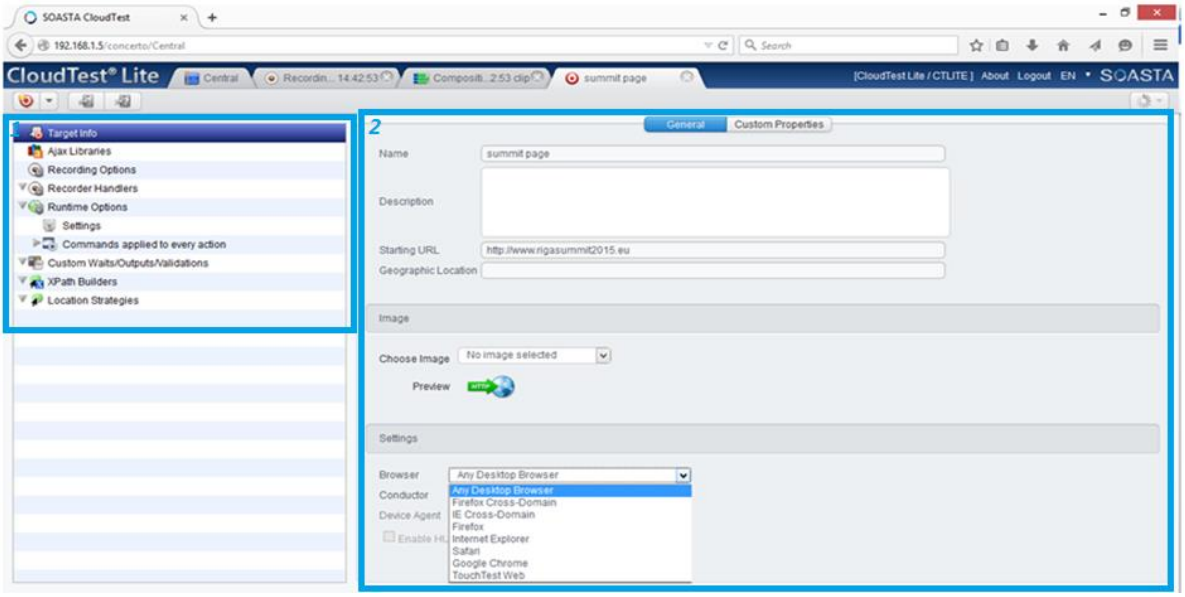
3.3. att. **Rezultāti, kas pamata panelī uzrādās.**

Rezultāti slodzes testiem pamata panelī uzrāda vispārīgos rezultātus (3.3. att. atzīmēts ar nr.1), atbildes laiku (3.3. att. atzīmēts ar nr.2), kļūdu (3.3. att. atzīmēts ar nr.3), kā arī virtuālo lietotāju (3.3. att. atzīmēts ar nr.5) skaitu. Autores veiktais tests tika pabeigts bez kļūdām uz vairākiem tīmekļa pārlūkiem.

Atsevišķā panelī var atlasīt logrīkus, kādi interesē un iegūt samērā detalizētus veiktspējas rezultātus. Papētot logrīku sarakstu sīkāk, darba autore secina, ka to daudzveidība šajā rīkā ir pilnībā apmierinoša.

3.1.4. Funkcionālie testi

Soasta CloudTest funkcionālo testu ieraksti tiek veikti tikai Firefox pārlūkporgrammā izmantojot Firefox papildinājumu windows operētājsistēmā vai Safari pārlūkporgrammu uz Mac operētājsistēmu datoriem.



3.4. att. **Funkcionālo testu saskarne**

Tīmekļa vietnei, kurai tiks veidoti funkcionālie testi, iespējams jau sākumā norādīt vai arī vēlāk pie testa izpildes, kurā pārlūkprogrammā lapa tiks testēta (3.4. att**.** atzīmēts ar nr.2).

Funkcionālajiem testiem tāpat rīkjoslā var norādīt detalizētākas norādes kā, piemēram, objektus pārvilkt (*drag and drop*), veikt aizturi pirms darbības izpildes, sagaidāmos rezultātus, izņēmumus, validāciju, utt.

3.1.5. Testu rezultāti

Pamata panelis rezultātā uzrāda testa stāvokli (Ir kļūdas/Nav kļūdas). Ja ir kļūdas, tad uzrāda kļūdu skaitu un konkrētu tās vietu, atbildes laiku, lietotāju skaitu, kā arī notikumu sarakstu. Atsevišķā panelī var atlasīt logrīkus, kādi interesē un iegūt samērā detalizētus funkcionālo testu rezultātus.

Darba autore bija patīkami pārsteigta, cik daudzfunkcionāls ir šis rīks bezmaksas versijā.

3.1.6. Secinājumi

Izmēģinot praktiski šo rīku, tika secināts, ka samērā viegla ir automatizēto testu veidošana, bet, lai spētu pilnībā izmantot šī rīka piedāvātās iespējas, ir nepieciešams ilgāks laiks, jo testu veidošana šajā rīkā ir laikietilpīgs process.

Ātri ir iespējams palaist izveidotos automatizētos testus dāžādās pārlūkrogrammās. Testu rezultāti tiek atgriezti analīzes skatījumā diagrammas veidā atzīmējot testus sinhronizēti ar laika līniju, kas atviegloja pārskatu un ir ar ilustratīvu nozīmi.

Šī rīka piedāvāto iespēju plašais klāsts spēj apmierināt arī tādu projektu vajadzības, kuriem izvirzītas augstas prasības pret kvalitatīvu un efektīvu testēšanas procesu.

Autorei šis rīks šķiet atbilstošs, jo ir ļoti labi piemērots spējās izstrādes modelim tā iemesla dēļ, ka viegli integrējams sistēmas izmaiņu gadījumos, kā arī gan funkcionālie, gan veiktspējas testi ir veicami šajā platformā.

Vienīgais mīnuss, ko darba autore saskatīja ir tas, ka pie mazākas brīvpiekļuves atmiņas darbošanās ar šo rīku ir ļoti lēna.

3.2. LoadStorm

LoadStorm ir mākoņbāzēts rīks slodzes testu veikšanai, kura pamatā tiek simulēta vairāku virtuālo lietotāju darbošanās veicot norādītu darbību kopumu. Loadstorm darbošānās notiek izmantojot pārlūku, veicot tīmekļa bāzētas lapas darbību ierakstu, kuru pēcāk izmanto automātisko testu veikšanai konfigurējot vidi - tajā skaitā lietotāju skaitu, to lokāciju, u.c. Loadstorm piedāvā saskarni, kur tiek veikta testu ierakstīšana, ierakstīto testu glabāšana un rezultātu analīze.

Tiek piedāvātas divas versijas - Lite un Pro. Lite ir vienkārša, viegli lietojama, bet Pro ir sarežģītāks, uzņēmumu līmeņa rīks ar modernu funkciju kopumu, kas spēj veikt sarežģītu veiktspējas analīzi.

Šajā gadījumā darba autore ir izvēlējusies LoadStorm Pro izmēģinājuma versiju, kura arī sīkāk tiks apskatīta.

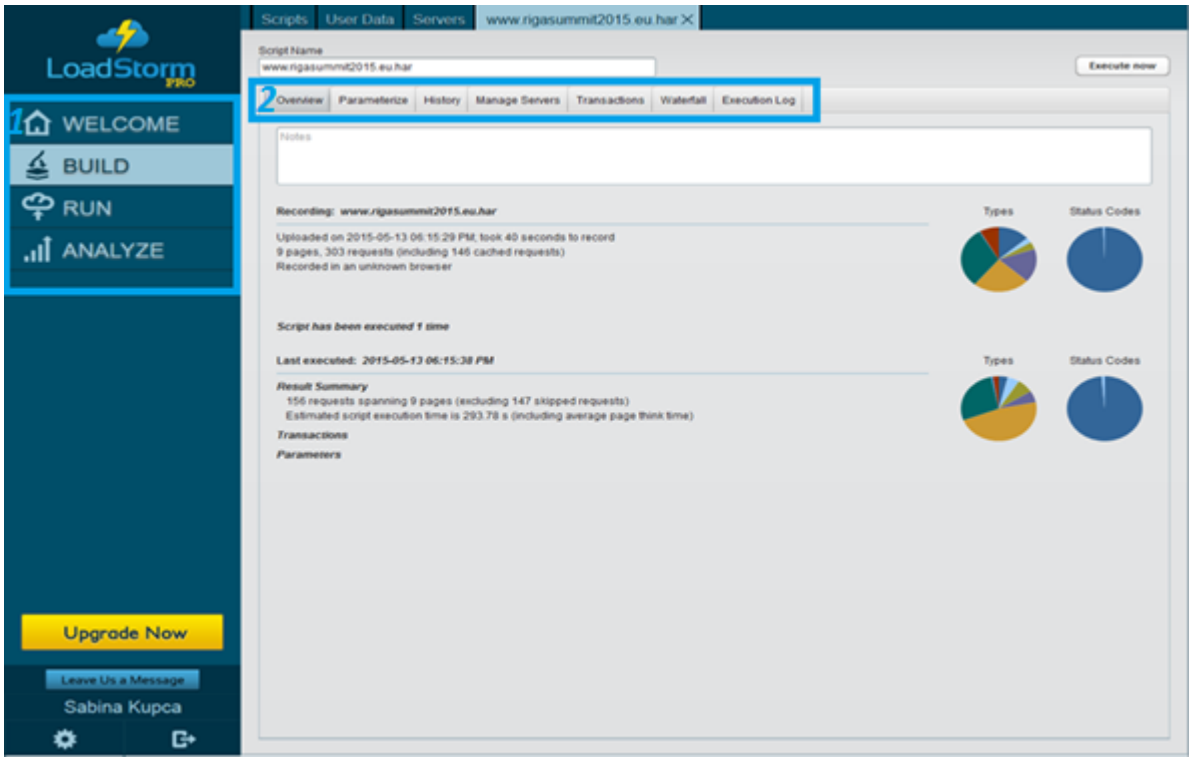
Visa darbošanās tiek iedalīta 3 soļos:

Izveidošana	• Ieraksta lietotāja veiktās darbības un augšuplādē ierakstīto .har formāta failu; • Augšupladē lietotāja datus, kas attiecas uz ierakstu • Izveido skriptu no ierakstiem un norādā kritērijus lietotāja datiem
Palaišana	• Izvēlas skriptus, kuriem tiks veikti slodzes testi; • Norāda slodzes testu palaišanas laiku vai arī var palaist uzreiz;
Analīze	• Slodzes testu izvērtēšana; • Vairāku testu rezultātu salīdzināšana turpat blakus novietojot vienu pie otra; • Testa rezultātu koplietošana;

[18]

Lai veiktu pirmos testus, izvēlas pārlūkprogrammu, ar kuru tiks veikts ieraksts. Salīdzinot ar Soasta Cloud Test rīku, arī šim rīkam ir nepieciešams papildinājums (Firebug 2.0.9). ko arī pievieno pārlūkprogrammai Firefox. Šajā konkrētajā gadījumā darbošanās notiks pārlūkprogrammā Chrome.

Rīka saskarne sastāv no vadības paneļa (3.5. att**.** atzīmēts ar nr.1) un vadības paneļa attiecīgās sadaļas, šajā gadījumā Build sadaļā testa parametri (3.5. att**.** atzīmēts ar nr.2).



3.5. att. **Rīka saskarne**

3.2.1. Testu veidošanas sadaļa

Šajā sadaļā ierakstīto .har failu augšupielādē, maina parametrus un lietotāju datus - laika limitus (ielādes aizture), lietotāja-aģenta detaļas, vēsturi, serveru pārvaldība (3.5. att**.** atzīmēts ar nr.2).

Kā augstāk ir minēts, vispirms darba autore izveido testpiemēru - tiek veikts ieraksts, kuru saglabā .har formāta failā. Tiek izveidots tas pats testpiemērs, kas tika veidots Soasta Cloud test rīka ietvaros - tīmekļa vietnes atvēršana konkrētā pārlūkā, rīkjoslas sadaļu atvēršana (pogu pārbaude), hipersaites pārbaude uz citu tīmekļa vietni, pārlūka aizvēršana.

3.2.2. Palaišanas sadaļa

Slodzes testu plānošana ietver skripta izvēli, testēšanas parametru specificēšanu un starta laika uzstādīšanu.

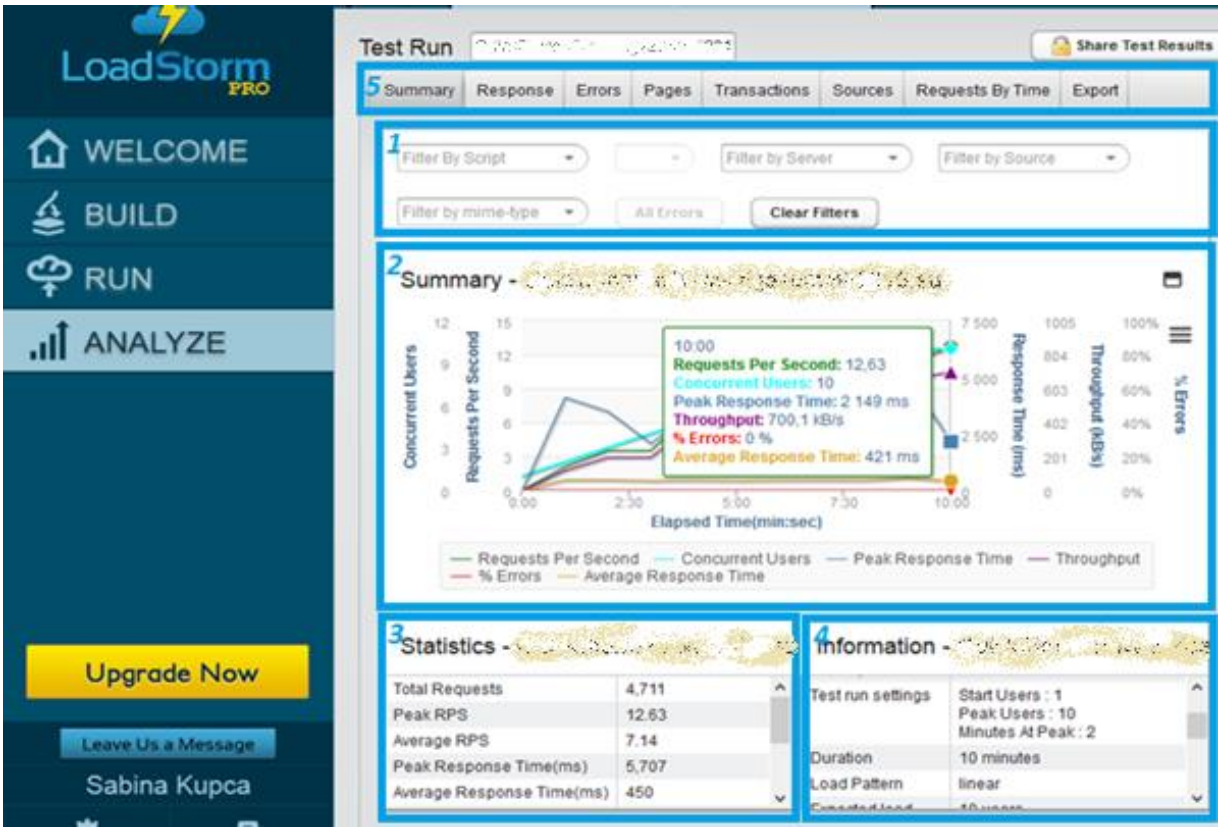
To visu var norādīt Palaišanas sadaļā, kurā atrodas rīkjosla ar vairākām iedaļām:

- *Parameters* – vieta, kur autore norādīja ielādes modeli, ilgumu, virtuālo lietotāju pieslēgumu skaitu;
- *Start Time* - norāda laiku, kad tiks sākta testa izpilde, atkārtošanas laiks un biežums. Autore palaida testpiemēru nekavējoties;
- *Traffic Source* - norāda virtuālo lietotāju pieslēguma vietu, ja lietotāju skaits pārsniedz 100 lietotājus;

Lai konfigurētu testu pret konkrēto serveri, ir nepieciešams to apstiprināt, ievietojot failu saknes direktorijā vai pievienot pārbaudes atzīmi attēlu rādītāja lapā (*index page*). Tā kā šis ir slodzes testēšanas rīks, tas liekas ļoti būtiski, lai novērstu ļaunprātīgu izmantošanu.

3.2.3. Analīzes sadaļa

Šī sadaļa parāda visu veikto testu rezultātus. Pēc grafika var spriest par visbiežāk sastopamajām kļūdām, lapu ielādes ātrumu, kuri no resursiem aizņēma visvairāk laika, u.c.



3.6. att. **Analīzes sadaļa**

Pārskata grafikā detalizēti var redzēt testa rezultātus - kļūdas procentu, vidējo atbildes laiku, caurlaidību, virtuālo lietotāju skaitu un pieprasījumus sekundē (3.6. att**.** atzīmēts ar nr.2). Turpat blakus var filtrēt rezultātus, atlasot vēlamos rezultātus, piemēram, konkrētu serveri (3.6. att**.** atzīmēts ar nr.1). Zem grafika atrodama testa statistika un testa informācija (3.6. att**.** atzīmēts ar nr.3 un nr.4).

Papildu sadaļas, kas atrodamas rīkjoslā, attēlo detalizētāku informāciju attiecīgajos parametros, kā arī testu rezultātu ekspotēšana .csv, .pdf formāta datnēm (3.6. att**.** atzīmēts ar nr.5).

3.2.4. Secinājumi

Liels ieguvums LoadStorm ir tas, ka nav pieciešams ilgs laiks vides uzstādīšanai. Darba uzsākšana ir ļoti vienkārša - reģistrācija. Arī pašu testu uzsākšana ir ātra - atver pārlūkprogrammu, pieslēdzas sistēmai un testēšanu var sākt.

Rīks ir viegli saprotams un darbošanās ar to ir piemērota vienkāršām darbplūsmām, bet ar samērā detalizētiem rezultāta datiem un konfigurējamiem testu veikšanas datiem.

Kā trūkumu var minēt to, ka netiek piedāvāta iespēja veikt funkcionālos testus pārlūkprogrammās, kas autorei šajā gadījumā būtu noderējusi, kaut vai to iemeslu dēļ, lai pārbaudītu darbību uz dažādām pārlūkprogrammām.

Tāpat, ja ir vēlme, var izmantot vairāk kā 50 virtuālos lietotājus vienlaicīgi, bezmaksas versija vairs nederēs. Parasti gan ar šiem 50 lietotājiem būtu jāpietiek, lai servera darbība tiktu traucēta, ja ir tāda iespēja.

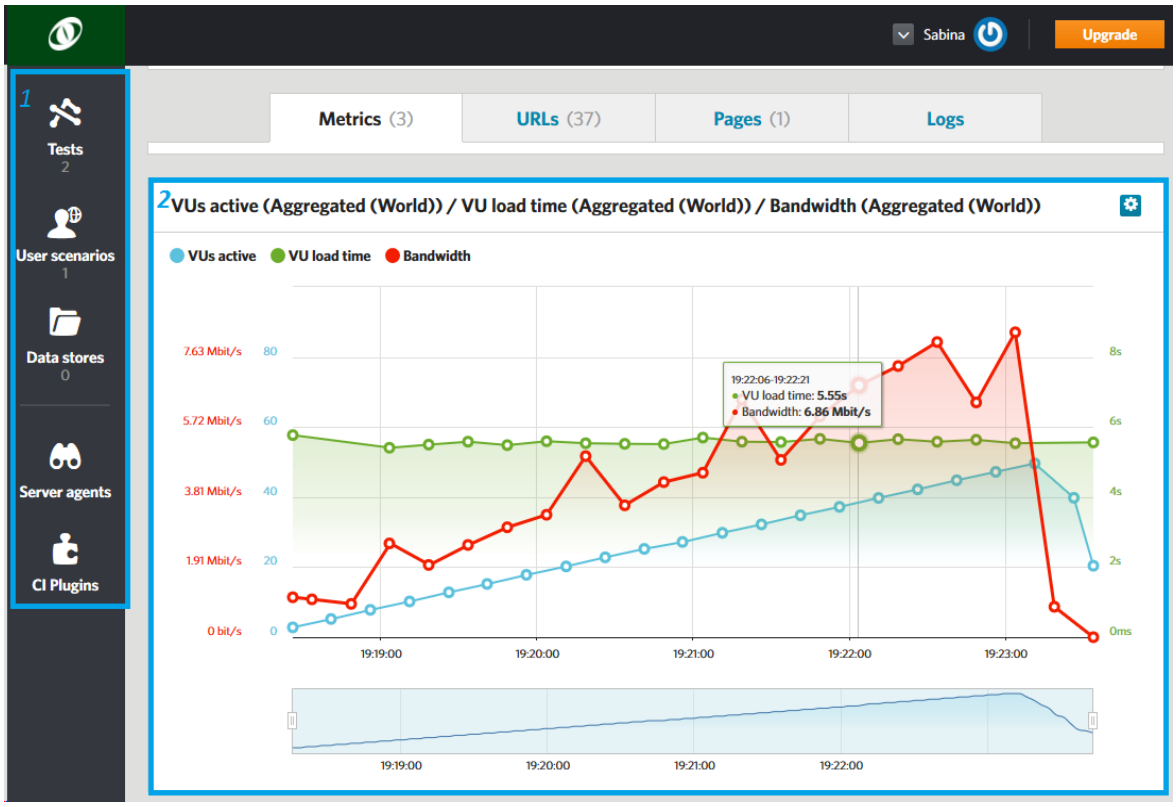
3.3. Load Impact

Load Impact ir vēlviens mākoņbāzēts serviss, kas nodrošina slodzes testus. Testi tiek veikti izmantojot pārlūkprogrammu, kurā norāda virtuālo lietotāju pieslēgumu skaitu un darbības, līdzīgi kā iepriekš apskatītais LoadStorm [8].

Rīks sevi piesaka, kā visbiežāk izmantotais tīmeklī pieejams veiktspējas testiem [19].

3.3.1. Testu veidošana

Šis rīks neveic lietotāju darbības ierakstu, bet gan testi tiek ģenerēti automātiski veidojot vienlaicīgus virtuālo lietotāju pieslēgumus no 10 dažādām pasaules vietām.

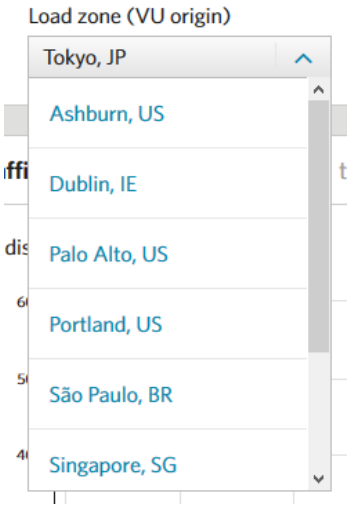


3.7. att. **Testu rezultātu skats**

Rīks ir izkārtots tā, ka kreisajā malā atrodas navigācijas panelis (3.7. att**.** atzīmēts ar nr.1).

Palaižot šo automātisko testu veikšanas rīku, tiek saņemta diezgan detalizēta tīmekļa vietnes analīze. Analīzes skats sniedz samērā plašu informāciju par lietotnes rezultātiem - savienojuma laiku, lejupielādes laiku, u.c Testpiemēra veikšanas brīdī lietotnē iebūvēts servera

aģents ievāc informāciju par centrālā procesora un atmiņas izmantošanu, tīkla ieejas/izejas datiem, kas pēcāk analīzes skatā ir apskatāms (3.7. att**.** atzīmēts ar nr.2).



*3.8. att***. Ģeogrāfiskā reģiona izvēle**

Testpiemēra izpildes laikā Load Impact spēj izveidot pieslēgumus līdz pat 10 ģeogrāfiskiem reģioniem vienlaicīgi, attiecīgi autore tos norāda pirms testpiemēra izpildes (skatīt 3.8. att.*)*.

3.3.2. Secinājumi

Rīks darbojas ļoti līdzīgi kā jau apskatītais LoadStorm. Ja ir nepieciešama vispārēja veiktspējas darbība ar vairākiem virtuālajiem lietotājiem no dažādām pasaules vietām, tad šis rīks ir perfekti tam piemērots. Tomēr, ja sagaida konfigurējamu vidi un pēc tam ar konkrētiem lielumiem veicot testus, tad šis rīks īsti nebūs tam piemērots.

No otras puses skatoties, šis rīks bezmaksas versijā piedāvā vairāk virtuālo lietotāju pieslēgumu testējamai tīmekļa vietnei.

3.4. Secinājumi tīmekļa lietotņu mākoņtestēšanas rīkiem

Praktiski pamēģinot darbā aprakstītos rīkus, katrs no tiem sniedza sava veida noderīgu informāciju. Katrs no šiem rīkiem ir veidots citādāk - darbošanās un sagaidāmais rezultāts atšķiras.

Ņemot vērā tikai šajā sadaļā apskatītos rīkus par noderīgāko šī projekta ietvaros autore atzīst Soasta Cloud Test. Lai gan LoadStorm rīks salīdzinoši ar Soasta neprasa tik lielu konfigurāciju un nav nepieciešama virtuālā mašīna. Tomēr Soasta Cloud Test spēj pārliecināt ar savu sniegumu un plašo funkcionalitāti.

Ja salīdzina LoadStorm ar Load Impact, tad bakalaura darba autoresprāt, veiktspējas rezultāti labāk tika attēloti Loadstorm nekā Load Impact. No otras puses skatoties, Load Impact rīks bezmaksas versijā piedāvā vairāk virtuālajo lietotāju pieslēgumu testējamai tīmekļa vietnei.

Neatkarīgi no rīka, pilnā versija paver daudz plašākas iespējas kā izmēģinājuma vai demo versija, tāpēc darba autore pieļauj, ka šo rīku funkcionalitāte ir plašāka, kā šobrīd ir apskatīta. Tīmeklī lielākoties brīvi piejami ir rīki slodzes un veiktspējas testu veikšanai, kas savā starpā galvenokārt atšķiras ar niansēm un izmaksām.

Praksē pielietojot šos rīkus, rezultātus un analīzi var iegūt salīdzinoši līdzīgus, tātad var secināt, ka rīka izvēle slodzes un veiktspējas testu veikšanai ir plaša un iegūstamie rezultāti vienlīdzīgi.

Diemžēl neviens rīks nespēja piedāvāt risinājumus drošības testēšanai.

4. MĀKOŅTESTĒŠANA MOBILAJĀM IERĪCĒM

Atkarībā no ierīces specifikas, defekti dažādām mobilajām lietojumprogrammām ir novērojami samērā bieži. Lai izvairītos no šādām kļūdam un nodrošinātu lietojumprogrammai uzticamību un kvalitāti, ir nepieciešama lietojumprogrammas testēšana uz dāžādām mobilajām ierīcēm, kas ir dārgi un laikietilpīgi [20], tādēļ tika sperts solis un izvēlēta mākoņtestēšana, jo tā piedāvā dažādas ierīces un to apvienojumu ar dažādām operētājsistēmu versijām.

Darba autore apskatīja pieejamos risinājumus un praktiski pārbaudīja, ko dažādi mākoņtestēšanas rīki spēj piedāvāt reālā projektā. Autorei konkrēta projekta ietvaros bija nepieciešams mākoņtestēšanas rīks, kas spēj atrast raksturīgās kļūdas dažādās mobilajās ierīcēs. Pieaugot ierīču skaitam, tas attiecīgi palielina atrasto kļūdu skaitu.

Mobilo rīku emulatori ir noderīgi, bet nespēj pilnībā aizstāt testēšanu uz reālām ierīcēm, jo nespēj attēlot lietotnes darbību reālā vidē pie reālas aparatūras, tapēc ir pieejami speciāli servisi, kas piedāvā attālināto piekļuvi simtiem reāliem viedtālruņiem un planštdatoriem.

4.1. tabula

Device Clouds

Cloud Service	Supported Platforms			Types of Testing	Delivery Type	
	Android	iOS	Other		Public	Private
Apkudo [68]	+			Stress (automated), New device approval	+	
Appium on Sauce [59]		+		Manual for web applications, Automated	+	+
Cigniti [55]	+	+	+	Automated, Interoperability, Performance, Network	+	
CloudMonkey [57]	+	+		Automated, UI-oriented	+	+
DeviceAnywhere [52]	+	+	+	Manual, Automated, Monitoring, Coverage	+	+
Jamo [67]	+	+	+	Automated		+
Perfecto Mobile [48]	+	+	+	Manual, Automated, Performance, Monitoring	+	
Scirocco Cloud [62]	+			Manual, Automated	+	
SeeTest [56]	+	+	+	Manual, Automated, On a new devices		+
SOASTA [54]	+	+	+	Manual, Automated, Load, Performance, Gesture-based	+	+
TestDroid Cloud [61]	+			Automated, UI-oriented, On a new devices	+	+
UFT Mobile [51]	+	+	+	Automated, Load, Performance, Monitoring		+
Zap-Fix [66]	+	+	+	Automated		+

Apkopojot populārākos mākoņtestēšanas rīkus, uzskatāmi tabulā tiek attēlots to atbalsts dažādām platformām, testēšanas veidiem un piegādes tipam (4.1. tabula) [21].

4.1. tabulā ir apkopoti mākoņtestēšanas pakalpojumu mobilo iekārtu risinājumi. Apkopojums ir veikts pamatojoties uz atbalstīto mobilo platformu skaitu, testēšanas un

mākoņpakalpojuma piegādes veidu. Ar manuālo testēšanu ir domāta tālvadības ierīču darbība, kas tiek nodrošināta izmantojot tīmekļa saskarni. Automatizētie testi ietver funkcionālo, regrestestēšanu un dažāda cita veida automatizētos testus. Publisko un privāto mākoņu veidi ir apskatīti darba teorētiskajā daļā.

Pētījumi un prakse rāda, ka visefektīvāk testēšanu veikt ir uz populārākajām ierīcēm, operētājsistēmām un ierīcēm ar dažādiem raksturīgiem uzstādījumiem [22].

Autorei svarīgi faktori un specifiskas problēmas mobilo ierīču testēšanā ir:

- Atbalsts daudzām aparatūras un programmatūras platformām;
- Lietotnes saskarnes korekta darbība ierīcēs;
- Attēlojamība uz dāžādu ierīču ekrānu izmēriem;
- Funkcionālie testi.

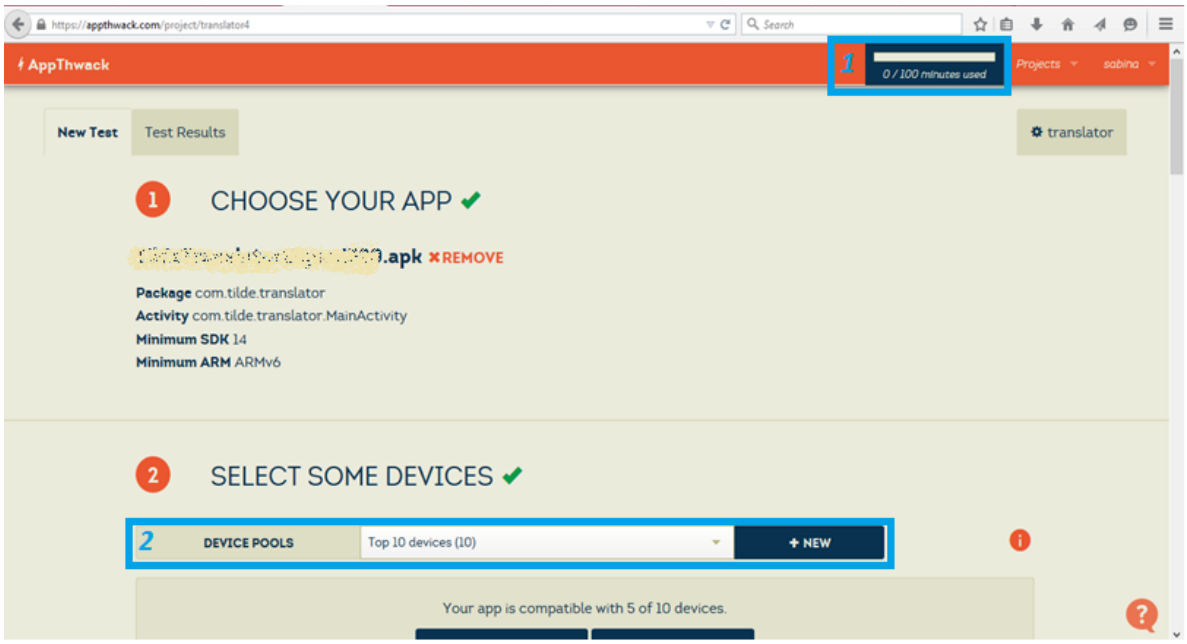
Praktiski tiks apskatīti daži populārākie mākoņtestēšanas rīki, tādi kā Perfecto Mobile, Testdroid, and AppThwack, kas atvieglo testēšanu un ko autore šajā nodaļā apskatīs detalizētāk konkrētas aplikācijas testēšanā.

4.1. AppThwack

AppThwack ir mākoņtestēšnas rīks, kas piedāvā lietotnes testēšanu uz android un iOS operētājsistēmu reālām ierīcēm. Tas automātiski veic testēšanu vienlaicīgi uz 100 reāliem mobilajiem viedtālruņiem un planšetdatoriem.

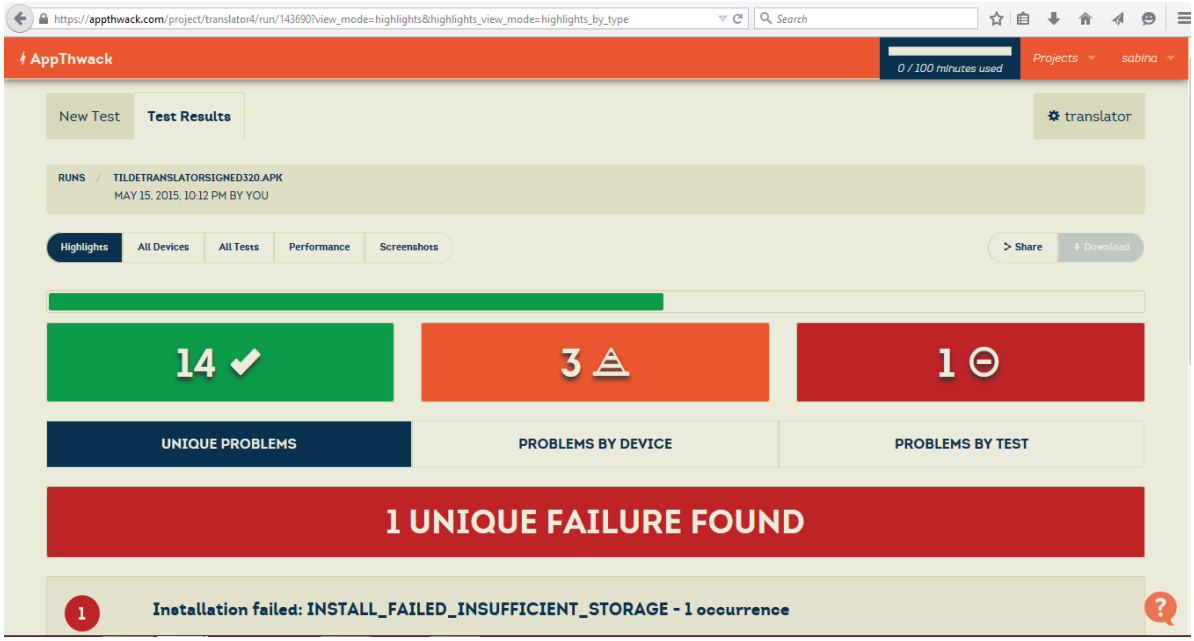
Tā kā darba autore lietoja izmēģinājuma versiju, piedāvātās ierīces uzrādījās krietni samazinātā skaitā, kā arī tika norādīts laika limits – 100 bezmaksas minūtes testu veikšanai (4.1. att**.** atzīmēts ar nr. 1). Pierakstoties šī rīka tīmekļa vietnē, bezmaksas izmēģinājuma versija tiek piedāvāta nedēļas garumā.

Praktiskā darbībā šis rīks tika pārbaudīts ar android lietotnes .apk failu.



4.1. att. **Testu veidošanas skats**

Pieslēdzoties sistēmai tika piedāvāta iespēja izveidot jaunu projektu, kurā veido testpiemēru norādot lietotni un iespējamās ierīces: visas 212 ierīces, top 10 vai top 25 (skatīt 4.1. att**.**). Testu veikšanai tiek piedāvāti automātiskie testi, kas iziet cauri lietotnes pamatfunkcijām.



4.2. att. **Rezultāti detalizētā skatā**

AppThwack testu veikšanas brīdī sāk uzrādīt pirmos kopsavilkumus, kas autorei jau laicīgi spēja sniegt pirmos rezultātus (skatīt 4.2. att.). Kad testpiemērs bija izgājis savus scenārijus, analīzes skatā tika iegūti diezgan detalizēti rezultāti.

Atgrieztie rezultāti autores gadījumā uzrādīja rezultātu ar katru iegūto ekrānu, kas ir pielīdzināts laika vienībai, kādā tas ir izpildījies, kā arī svarīgāko daļu – kļūdas, kas 4.2. att. jau ir redzamas.

Pēc pirmo testpiemēru izlaišanas cauri un pirmo rezultātu analīzes, autore secina, ka pamata testus (instalācija, pamata funkcijas, atinstalācija), šis rīks spēj veikt, bet, diemžēl, šis rīks nepiedāvā drošības prasību testus. Kā arī šis rīks ir vājš automātiskajiem skriptiem, to modificēšanai un citiem sarežģītākiem testpiemēriem.

4.3. att. **Rezultāti detalizētā skatā**

Apskatot sīkāk top 10 ierīces, ko piedāvā AppThwack un salīdzinot ar aktuālajām šobrīd, tiek secināts, ka ir minētas populāras ierīces un, pēc autores domām, piedāvājums pilnībā apmierina. AppThwack piedāvā veiktspējas rezultātus uz izvēlētajām iekārtām, kas lieliski parāda, cik konkrētā lietotne izmanto ierīces atmiņu, kāda ir centrālā procesora noslodze (skatīt 4.3. att.). Autores testējamā lietotnē viss ir normas robežās 4.3. att. - skaidri ir redzams, kādas ierīces ar kādiem procesoriem un operētājsistēmām uzrāda vislabāko sniegumu.

Turpat detalizētā skatā, autore var apskatīt pikseļperfektus ekrānšāviņus, kas sniedz vizuālu pārliecību, ka konkrētos ekrāna izmēros attēlojums nemainās un saglabājas tāds, kāds tiek sagaidīts. Apskatot tos autore apstiprina, ka uz visu ekrāna izmēru ierīcēm, attēlojums ir vienādi proporcionāls.

Rīka aprakstā tiek minēts, ka tas atbalsta populārākos automatizēto testu ietvarus, kas šajā darbā sīkāk netiks apskatīti.

4.1.1. Secinājumi

Autoresprāt, pietiekoši labs rīks, lai pārbaudītu pamata funkcionalitāti, novērtētu lietotnes vājās vietas un atrastu pirmās kļūdas. Iebūvētais testa komplekts neprasa programmkodu, kā arī iekļauj lietotnes instalāciju, lietotnes palaišanu, kā arī aplikācijas izpēti (gudrs meklētājs, kas dinamiski pāriet lietotnes galvenās sadaļas un mijiedarbojas uz programmu, pakļaujot to slodzei).

Rīks gandrīz atbilst visām tā izvirzītajām prasībām. Tas spēja atbalstīt pietiekoši lielu daudzumu ar populārākajām platformām, attēloja datus uz dāžādu ierīču ekrānu izmēriem, vēl jo vairāk uzrādot veiktspējas datus katrai ierīcei. Līdz ar to darba autorei bija nepieciešams novērtēt atrastās kļūdas un manuāli veikt funkcionālos testus lietotnē.

Pie vēlamajiem testiem palika lietotnes darība apstrādājot dažādus paziņojumus (piemēram baterijas uzlādes līmenis, sadarbspēja ar saņemtajām īsziņām un zvaniem), tīkla savienojuma pārrāvuma apstrāde.

Izlabojot atrastās kļūdas, ļoti vienkārša ir testu atkārtota palaišana un atgriezto testa rezultātu salīdzināšana.

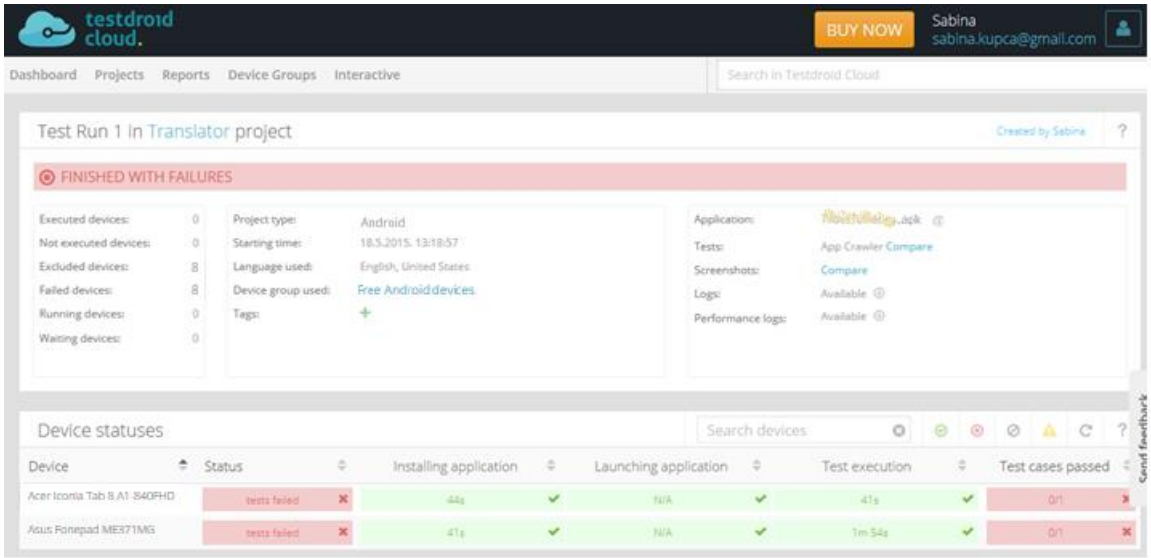
4.2. Testdroid

TestDroid tāpat kā AppThack ir mākoņtestēšanas platforma, lai veiktu lietotāja saskarnes un funkcionālos testus paralēli uz vairākām reālām ierīcēm. Testēšana uz reālām ierīcēm ir vienīgais veids, lai pārliecinātos, ka tiek sniegta lietotājiem vislabākā iespējamā pieredze [23].

TestDroid ierakstītājs (*Recorder*) kā eclipse spraudnis ir piemērots automatizēto testpiemēru veidošanai un ir pieejams bezmaksas versijā samērā nesen - kopš 2014. gada rudens. Līdz ar šo spraudni nav nepieciešams rakstīt testus manuāli, bet gan ar to var veikt darbību kopumu ierakstu. TestDroid ierakstītājs var tikt izmantots, lai veidotu atkārtoti izmantojamus Android JUnit testpiemērus.

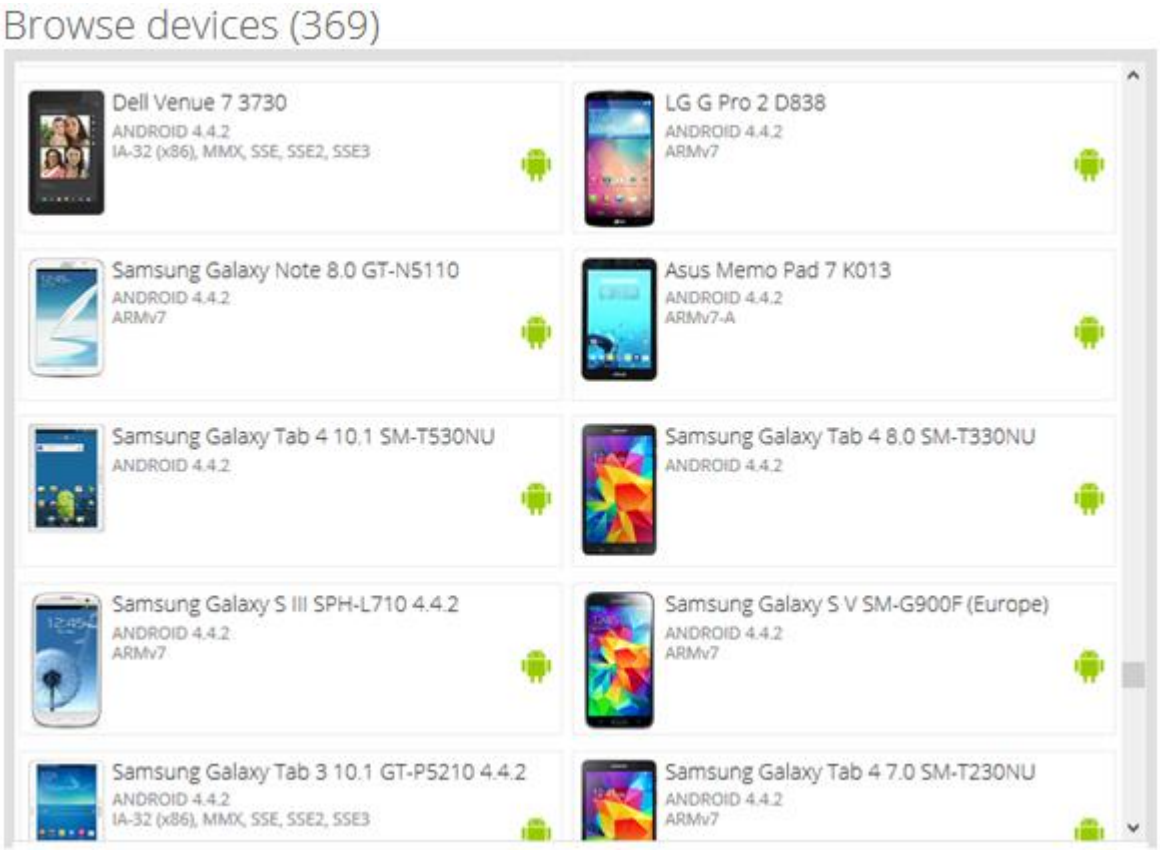
Autore šo rīku pētīja praktiski un secināja, ka darbošanās ir ļoti noderīga un ērta – nav nepieciešams pirmskods, ārējās bibliotēkas, u.c.

Lai pārbaudītu, cik TestDroid labi darbojas praksē un atrod kļūdas, autore ievieto .apk failu ar zināmām kļūdām.



4.4. att. **Testu rezultātu skats**

Rezultāti ir un tie ir diezgan labi – lietotne tiek uzinstalēta, palaista, testpiemērs ir veicis darbu un gala rezultātā sagaidāmais rezultāts nav sakritis ar paredzēto (skatīt 4.4. att.).



4.5. att. **Pieejamās Android ierīces**

Diemžēl bezmaksas versijā nav pieejamas pašas populārākās ierīces, līdz ar to testi, kas veikti uz tām, nav 100% uzticami. Ierīču izvēle ir pieejama tikai maksas versijā un tā ir ievērojama. Tāpat arī var izvēlēties operētājsistēmu versijas, kur attiecīgi tiek attēlotas ierīces uz kurām tās ir pieejamas. Konkrēti ar Android operētājsistēmu ir pieejamas 369 ierīces, kas pēc darba autores domām ir vairāk kā pietiekami (skatīt 4.5. att.).

Tāpat kā iepriekš apskatītajam AppThwack rīkam, arī šeit analīze tiek atgriezta diezgan pietiekama.

4.2.1. Secinājumi

Praktiski izmēģinot šo rīku, ir skaidrs, ka tam ir vairākas priekšrocības. TestDroid piedāvā iespēju izmantot jau piedāvāto testpiemēru, kas ietver aplikācijas instalēšanu, palaišanu, pamatfunkciju veikšanu un atinstalēšanu, ko piedāvāja arī iepriekš aplūkotais rīks. Otra iespēja ir izveidot automatizētos testpiemērus, kas tiek ierakstīti izmantojot Eclipse spraudni.

Beidzoties plānotajiem testiem, lietotājs tiek par to informēts – uz norādīto epasta adresi pienākot epastam ar veiksmīgu vai kļūdainu gala rezultātu.

Rīks spēja atgriezt rezultātu, kas satur ekrānuzņēmumus un ierīces .log failu ar veikto testu rezultātiem. No ekrānuzņēmumiem var spriest par lietotnes saskarnes attēlojumu uz dažādu ierīču ekrāniem. Šajā gadījumā tos apskatot nekādas nepilnības netika atrastas.

Pilnajā versijā ir iespējams izvēlēties, uz kurām ierīcēm veikt testus – vienas, piecām vai jebkurām citām, līdz ar to, šobrīd iegūtie rezultāti ir iegūti uz ierīcēm, kas nav pašas pieprasītākās un nespēj sniegt tik precīzus datus, kā gribētos.

Apskatot .log failu ar veiktajiem testiem un to rezultātiem, maldinošus datus atgrieza par sensoru darbību (priekšējās kameras), lai arī tas nebija šoreiz svarīgi lietotnei.

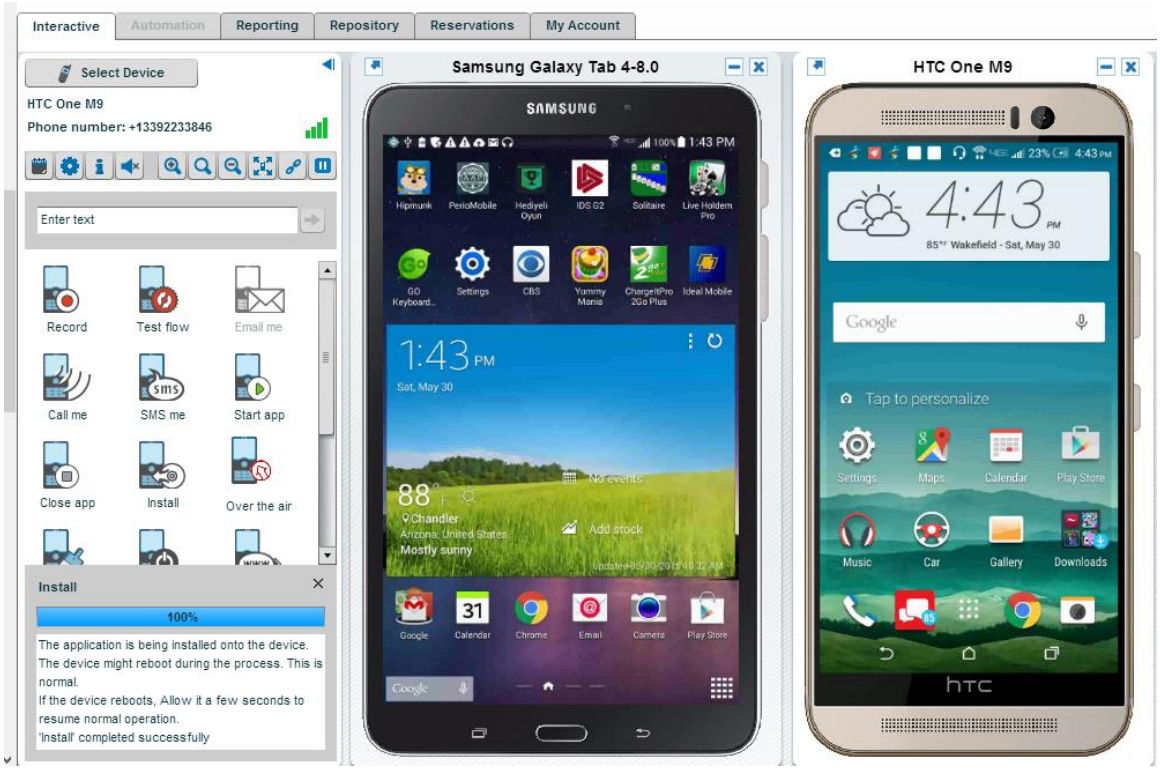
4.3. Perfecto Mobile

Perfecto Mobile ir mazliet citādāks mākoņtestēšanas pakalpojums, kā augstāk apskatītie un ir paredzēts lielākoties funkcionālajiem testiem darbojoties dažādu ierīču saskarnē. Perfecto Mobile izmanto iebūvētu logrīku, kas tiek izmantots vizbiežāk sastopamajām darbībām – lietotnes instalēšanai, palaišanai, tekstu ievadei, u.c.

Perfecto Mobile sniedz pilnīgi atšķirīgu informāciju par lietotni kā iepriekš apskatītie, kā arī atbalsta ne tikai ļoti plašu klāstu ar iOS, Android, taču arī WindowsPhone un BlackBerry ierīcēm.

Salīdzinot ar iepriekš apskatītajiem servisiem, šajā uzsvars tiek likts uz manuālo testu veikšanu. Respektīvi nav automātisko pamata testu. Bezmaksas režīmā tika piedāvāta divu stundu manuālās testēšanas iespēja. Automatizētie un veiktspējas testi tiek piedāvāti tikai maksas versijā.

Saskarne ir vizuāli saprotama un liels uzsvars tiek likts un iespēju darboties reālā ierīcē ar reāliem ierīces uzstādījumiem un pat tos mainīt.



4.6. att. **Saskarne**

Praktiskā darbībā tika uzinstalēta lietotne uz divām ierīcēm - planšetdatora Samsung galaxy tab 4-8.0, kas atbalsta Android 4.4.2 versiju un HTC One M9, kas atbalsta Android 5.0. versiju (skatīt 4.6. att.). Lietotne tika palaista darbībā un veiktās darbības ierakstītas. Tika pamainītas lietotnes, kas atrodas uz ierīces, varēja uzinstalēt jaunas no lokālā datora vai veikala (*Play Store*). Lai gan autores lietotnei tas nebija aktuāli, tas var būt aktuāli cita projekta ietvaros.

Tāpat tika izmantota izdevība pamainīt sistēmas parametrus un ārējos ietekmējošos faktorus, kā zvani, īsziņas paziņojumi, internets pārrāvuma gadījumi, kas palīdzēja veikt nefunkcionālo testēšanas veidu - datorkļūmes novēršanas testēšanu.

Līdz ar visu piedāvāto darbību kopumu lietotnes funkcionālie testi uz šīm divām ierīcēm izgāja bez problēmām. Ja atļautais darbības laiks bezmaksas versijā nebūtu ierobežots, tad testpiemēri tiktu veikti vairāk.

4.3.1. Secinājumi

Mērķis lietotnes testēšanā tika sasniegts un funkcionālie testi ar dažādiem sistēmas iestatījumiem tika veikti.

Salīdzinot ar iepriekš apskatītajiem rīkiem, šajā sistēmā bezmaksas versijā darbošanās bija manuāli ierīces ietvaros, kas ir pieskaitāms pie pozitīvajiem testpiemēriem. Testpiemēra izpildes redzamība sniedz lielāku pārliecību par funkcionalitātes korektu darbību. Kā arī bija iespējams viegli pārbaudīt jaunu būvējumu ar labotām kļūdām, kas tika atklātas funkcionālo testu ietvaros.

4.4. Secinājumi mobilo ierīču mākoņtestēšanas rīkiem

Praktiski pamēģinot darbā aprakstītos rīkus, tika veikts novērtējums un izdarīti secinājumi, cik šie rīki ir bijuši lietderīgi reālā projektā testējot lietotni.

Lielākoties izvirzītie mērķi vairāk vai mazāk, bet tika sasniegti. AppThwack un Perfecto Mobile spēja atbalstīt pietiekoši lielu daudzumu ar populārākajām platformām, attēloja datus uz dāžādu ierīču ekrānu izmēriem, AppThwack vēl jo vairāk uzrādot veiktspējas datus katrai ierīcei. Mazliet sliktākus rādītājus uzrādīja Testdroid nepiedāvājot bezmaksas versijā populārākās ierīces.

Situācijas straujajai attīstībai mobile ierīču tirgū – pie apstākļiem, ka jaunas ierīces nāk klāt straujos tempos, visu rīku ierīču sarakstā visas jaunākās jau bija atrodamas.

AppThwack priekšrocība ir tā, ka rīks atgriež izmantoto ierīces atmiņu un centrālā procesora datus, jo ne Testdroid, ne Perfecto Mobile šādu pārskatāmu attēlojumu nesniedz.

Perfecto Mobile funkcionālie testi apvienojumā ar AppThwack sniegtajiem veiktspējas testiem, veido atbilstošu vidi, lai veiktu lietotnes pilnvērtīgu testēšanu.

REZULTĀTI

Tika izpētīts mobilo ierīču mākoņtestēšanas piedāvājumu klāsts praksē un autores darba vietā veikta sekmīga lietotnes publicēšana veicot testēšanu iesaistot apskatītos rīkus. Tika veikta programmatūras testēšana un sniegts iegūto rezultātu novērtējums un lietderīgums.

Papildu izvirzītajām problēmām, mākoņtestēšanas rīki spēja nodrošināt vienībtestēšanu un regrestestus, kā arī uzrādīt veiktspējas un slodzes testu rezultātus.

Salīdzinot izpētītos rīkus nevar novērtēt visprecīzāko rezultātu, jo no katra rīka tika iegūti derīgi rezultāti. Viens no tiem sniedza vairāk funkcionālo testu rezultātus, savukārt otrs vairāk veiktspējas rezultātus un trešais vispārējo analīzi.

Veicot manuāli Perfecto mobile funkcionālos testus pēc iepriekš izveidota plāna, pirmajās versijās tika atklātas kļūdas, veicot kļūdu labojumu, lietotnes izlaišanas kandidāts testēšanu veica sekmīgi. Šis rīks tika novērtēts kā labs esam funkcionālo testu veikšanai uz ierīcēm, kas nav fiziski pieejamas un piedāvāts šo rīku ieviest darba devējam kā reālu variantu.

SECINĀJUMI

Uzsākot šo darbu tika uzstādīti šādi darba mērķi:

- Iepazīties ar mākoņa testēšanas pieejamajiem servisiem;
- Pārbaudīt, cik labi tie ir reālā programmatūras testēšanā.

Risinot uzstādītos mērķus, darba autore ir nonākusi pie šādiem secinājumiem:

- Mākoņtestēšanas ieviešana ir laikietilpīgs un ļoti sarežģīts process, kuram nepieciešama dažādu rīku izpēte un izvērtējums;
- Eksistē daudz un dažādi mākoņtestēšanas rīki, kuri nodrošina testēšanu gan tīmekļa lietotnēm, gan mobilajām lietojumprogrammām. Piemērotākie risinājumi ir atrodami konkrēti katra uzņēmuma, projekta un sistēmas vajadzībām.
- Mākoņtestēšana ir strauji augoša joma mākoņdatošanā un reāla iespēja samazināt testēšanas izmaksas.
- Priekšrocības mākoņtestēšanā ietver atbalstu ne tikai veiktspējas, slodzes un funkcionālajiem testiem, bet arī nodrošina nepieciešamos aparatūras resursus (t.s. tālvadības viedtālruņus) dažādiem mobilo ierīču testiem.

Autore secina, ka iegūtais rezultāts ir pietiekams, lai secinātu, ka izvirzītie mērķi ir sasniegti. Tīmekļa vietnei tika veikti veiktspējas, slodzes un funkcionālie testi, kas detalizēti un uzskatāmi attēloja veikto testpiemēru rezultātus, tajā skaitā arī dažādām pārlūkprogrammu versijām.

Bakalaura darbā tika mēģināts sniegt pārskatu par jaunākajiem pētījumu rezultātiem, pieejām un programmatūras rīkiem un to pielietojumiem reālā projektā. Lai gan mākoņtestēšana joprojām attīstās, daži interesanti teorētiskie un praktiskie rezultāti ir sasniegti. Nav šaubu, ka šī ir daudzsološa joma, kurai tiks pievērsta uzmanība turpmākajos gados un tā turpinās izvērsties.

Bakalaura darba izstrādes gaitā svarīgu daļu aizņēma izpēte - vai un kā mākoņtestēšana var piedāvāt veiktspējas un funkcionālo testēšanu efektīvāku. Tas palīdz testētājam neraizēties par testēšanas rīkiem, tāpat arī atbrīvo uzņēmumu no programmatūru iegādes, uzstādīšanas, licencēšanas un atjaunināšanas. Testētājs var koncentrēties uz testu izpildi.

Tā kā mākonis ir pieejams nepārtraukti, tad tā lietotāji var koncentrēties uz testu veikšanu, nevis gaidīt uz testēšanas vides sagatavošanu. Mākonis piedāvā veiktspējas un funkcionālo testēšanu kā pakalpojumu, tādējādi palielinot pārredzamību gan testēšanai, gan rezultātiem. Tajā pašā laikā skaitļošanas resursi tiek izmantoti tikai veicot testēšanu un netiek izmantoti pēc pārbaudes, tādējādi ietaupot budžeta izmaksas.

IZMANTOTĀ LITERATŪRA

1. http://en.wikipedia.org/wiki/Cloud_testing
2. Mākoņpakalpojuma definīcija [tiešsaiste] [skatīts: 05.03.2015.] Pieejams: http://www.belatrixsf.com/index.php/whitepaper-cloud-testing-best-practices
3. **Vilkomir**, *S. Cloud Testing: A State-of-the-Art Review [Text]* / S. Vilkomir // Information & Security: An International Journal. – 2012. – V. 28, Is. 2, No. 17 – P. 213 - 222.
4. **Tilley. S.** *Software Testing in the Cloud: Perspectives on an Emerging Discipline* [Text] / **S. Tilley, T. Parveen** // IGI Global. – November 2012.
5. **Tsai, W.** *Testing as a service over cloud* [Text] / **W. Tsai, X. Chen, L. Liu, Y. Zhao, L. Tang, W. Zhao** // Proceedings of the 5th IEEE International Symposium on Service Oriented System Engineering. – 2010. – P. 181 – 188.
6. **Kalliosaari, L.** *Testing in the Cloud: Exploring the Practice* [Text] / **L. Kalliosaari, O. Taipale, K. Smolander** // IEEE Software. – 2012. – V. 29, Is. 2 – P. 46 - 51.
7. **Weidong, F.** *Cloud testing: The next generation test technology* [Text] / **F. Weidong, X. Yong** // Proceedings of the 10th International Conference Electronic Measurement & Instruments (ICEMI). – Chengdu, China, August 16 - 19, 2011. – P. 291 - 295.
8. **Sergiy Vilkomir**. *Cloud testing: a state-of-the-art review* [tiešsaiste] [skatīts: 15.03.2015.] Pieejams: http://it4sec.org/system/files/28.17_Vilkomir.pdf
9. *What exactly is 'cloud testing' anyway?* [tiešsaiste] [skatīts: 05.02.2015.] Pieejams: http://www.networkworld.com/article/2225173/opensource-subnet/what-exactly-is--cloud-testing--anyway-.html
10. **Leah Muthoni Riungu, Ossi Taipale, Kari Smolander,** *"Research Issues for Software Testing in the Cloud,"* Proceedings of the IEEE Second International Conference on Cloud Computing Technology and science, Indianapolis, Indiana USA, 30 Nov. – 3 Dec. 2010, 557-564.
11. *Cloud Computing Trends: 2014 State of the Cloud Survey* [tiešsaiste] [skatīts: 09.04.2015.] Pieejams: http://www.rightscale.com/blog/cloud-industry-insights/cloud-computing-trends-2014-state-cloud-survey

12. **Vinit B. Mohata, Dhananjay M.Dakhane, Ravindra L.Pardhi** *Cloud Based Testing: Need of Testing in Cloud Platforms* [tiešsaiste] [skatīts: 19.05.2015.] Pieejams: http://www.ijaiem.org/Volume2Issue3/IJAIEM-2013-03-31-130.pdf
13. *Getting Started with 'Cloud Testing'* [tiešsaiste] [skatīts: 19.05.2015.] Pieejams: http://www.softwaretestinghelp.com/getting-started-with-cloud-testing/
14. **Eljon Proko & Ilia Ninka** *Global Journal of Computer Science and Technology Cloud & Distributed, Analysis and Strategy for the Performance Testing in Cloud Computing* Pieejams: https://globaljournals.org/GJCST_Volume12/2-Analysis-and-Strategy-for-the-Performance-Testing.pdf
15. **Dr. Rahul Malhotra &Prince Jain** *Testing Techniques and its Challenges in a Cloud Computing Environment.* Pieejams: http://www.thesij.com/papers/CSEA/2013/July-August/CSEA-0103550201.pdf
16. **S.Nachiywappan, S. Justus** *Cloud Testing Tools and Its Challenges: A Comparative Study* Pieejams: http://ac.els-cdn.com/S1877050915005190/1-s2.0-S1877050915005190-main.pdf?_tid=1415e2f8-fed4-11e4-aab1-00000aab0f6b&acdnat=1432114935_af25136f71d10e1dd2f011c382c8c847
17. **G.Gowri, M. Amutha** *Cloud Computing Applications and their Testing Methodology* Pieejams: http://www.ijircce.com/upload/2014/february/7B_Cloud.pdf
18. *LoadStorm user manual* [tiešsaiste] [skatīts: 07.05.2015.] Pieejams: http://d2av97idjaqjyo.cloudfront.net/wp-content/uploads/2014/01/loadstorm_user_manual.pdf
19. *Load Impact tīmekļa vietne* [tiešsaiste] [skatīts: 10.05.2015.] Pieejams: https://loadimpact.com/why-us
20. **Serrgiy Vilkomir and Brandi Amstutz** *Using Combinatorial Approaches for Testing Mobile Applications* Pieejams: http://core.ecu.edu/STRG/publications/Vilkomir-IWCT-2014-Proceedings.Pdf
21. **Tao Zhang, Jerry Gao, Jing Cheng, Tadahiro Uehara** *Compatibility Testing Service for Mobile Applications* Pieejams: http://www.researchgate.net/profile/Jerry_Gao/publication/274370752_Compatibility_Testing_Service_for_Mobile_Applications/links/551bf4d00cf20d5fbde22d8c.pdf
22. **O. Starov, S. Vilkomir** *CLOUD SERVICES AND TOOLS FOR MOBILE TESTING* Pieejams: http://www.irbis-nbuv.gov.ua/cgi-bin/irbis_nbuv/cgiirbis_64.exe?I21DBN=LINK&P21DBN=UJRN&Z21ID=&S21REF=1

0&S21CNR=20&S21STN=1&S21FMT=ASP_meta&C21COM=S&2_S21P03=FILA=&2_S21STR=recs_2013_5_62

23. **Jouko Kaasila, Denzil Ferreira, Vassilis Kostakos, Timo Ojala** *Testdroid: automated remote UI testing on Android* Pieejams:

http://www.mediateam.oulu.fi/publications/pdf/1457.pdf

Bakalaura darbs „Mākoņtestēšanas risinājumi” izstrādāts LU Datorikas fakultātē.

Ar savu parakstu apliecinu, ka pētījums veikts patstāvīgi, izmantoti tikai tajā norādītie informācijas avoti un iesniegtā darba elektroniskā kopija atbilst izdrukai.

Autore: Sabīna Kupča

Rekomendēju darbu aizstāvēšanai

Vadītājs: profesors Dr.dat. Guntis Arnicāns 01.06.2015.

Recenzents: docents Dr.sc.comp. Maksims Kravcevs

Darbs iesniegts Datorikas fakultātē 01.06.2015.

Dekāna pilnvarotā persona: vecākā metodiķe Ārija Sproģe

Darbs aizstāvēts bakalaura gala pārbaudījuma komisijas sēdē

__.06.2015. prot. Nr. , vērtējums

Komisijas sekretāre: