LATVIJAS UNIVERSITĀTE
DATORIKAS FAKULTĀTE

TELOS VX FUNKCIONĀLO TESTU AUTOMATIZĀCIJA

KVALIFIKĀCIJAS DARBS

Autors: **Uga Sproģis**
Studenta apliecības Nr.: us16005
Darba vadītājs: M.dat. Matīss Rikters

RĪGA 2018

ANOTĀCIJA

Šis kvalifikācijas darbs izstrādāts saistībā ar programmēšanas praksi Latvijas Universitātes Matemātikas un informātikas institūtā. Šis darbs sniedz ieskatu VX VoIP radioapraides telefona iekārtas funkcionalitātes un šīs sistēmas tīmekļa vietnes lietotāja saskarnes testu automatizācijā. Darbā aprakstītā programmatūra ir šīs iekārtas funkcionalitātes testu automatizācijas skripti.

Darba mērķis ir atvieglināt testēšanas procesu. Testu automatizācija nodrošina ātrāku, vienkāršāku un efektīvāku testēšanas procesu salīdzinājumā ar manuālo testēšanu. Šo testu lielākais ieguvums ir iespēja tos viegli atkārtot un paplašināt, lai veiktu uzdevumus, kas nav iespējami ar manuālo testēšanu, kā arī tie nodrošina to, ka jebkādu funkciju izmaiņu gadījumā var ātri un ērti pārbaudīt vai konkrētās izmaiņas nav izmainījušas kādas citas funkcijas darbību un ļauj ātrāk atrast, identificēt un salabot jebkādas šāda veida problēmas.

Gala rezultātā ir nodrošināta VX VoIP radioapraides telefona iekārtas funkcionalitātes un šīs sistēmas tīmekļa vietnes lietotāja saskarnes testu automatizācija, kas būtiski samazina laika patēriņu iekārtas izstrādes procesa testēšanas fāzē un viennozīmigi pašam uzņēmumam arī izmaksas.

Atslēgvārdi: Automatizēšana, testēšana, radio studijas vadības sistēma, radio apraides iekārta.

ABSTRACT

TELOS VX FUNCTIONALITY TEST AUTOMATION

This qualification thesis was developed in association with programming practice internship in the Institute of Mathematics and Computer Sciences, University of Latvia. This thesis provides insight in the testing automation of the VX VoIP broadcast phone systems functionality and its websites user interface. In this thesis, the described software is the mentioned systems functionalities test automation scripts.

The aim of this thesis is to facilitate the testing process. Test automation provides a faster, simpler and more efficient testing process than manual testing. The greatest benefit of these tests is the ability to easily repeat and extend them to perform tasks that are not possible with manual testing, also that tests after said changes provides a quick and handy way to check if the changes have not had some unexpected and unwanted effect on other function and allows for a faster identifying and fixing of any problems of such nature.

In the end, the test automation for the VX VoIP broadcast phone systems functionality and its websites user interface is ensured, which significantly reduces the time for the testing phase of the equipment development and the cost to the company itself.

Keywords: Automation, testing, radio studio management system, radio broadcasting equipment.

APZĪMĒJUMU SARAKSTS

Apzīmējums	Skaidrojums
Telos VX	Uzņēmuma "Telos Alliance" izstrādāta VoIP radioapraides telefona iekārta.
VoIP	Voice over Internet Protocol - balss pārraide ar interneta protokolu.
Skripts	Instrukciju virkne, kas nosaka, kā programmai jāveic kāda specifiska procedūra.
SIP	Sesijas inicializācijas protokols - standarta signalizācijas protokols, multivides komunikāciju sesiju izveidošanai starp diviem vai vairākiem lietotājiem IP tīklā.
HTML	Hypertext Markup Language - hiperteksta iezīmēšanas valoda, kas, izmantojot speciālus kodus, nosaka hiperteksta dokumenta jeb tīmekļa lappuses atveidojumu pārlūkprogrammas logā.
XML	eXtensible Markup Language - programmēšanas valodas SGML apakškopa, kas speciāli izstrādāta darbam ar tīmekļa dokumentiem. Valoda XML nodrošina globālā tīmekļa izstrādātajiem iespēju radīt savas personiskās birkas (kodus), lai nodrošinātu tādu funkciju izpildi, ko nevar nodrošināt ar valodas HTML starpniecību.

IEVADS

Telos VX[1] ir viena no uzņēmuma “Telos Alliance” izstrādātajām VoIP radioapraides telefona iekārtām. Šī ir pati pirmā VoIP sarunu šovu sistēma, kas ir īpaši izstrādāta un izveidota apraidei. Kvalifikācijas darbā ir aprakstīta Telos VX sistēmas funcionālo testu un šīs sistēmas tīmekļa vietnes lietotāja saskarnes testu automatizācija. Tēma bija izvēlēta pamatojoties uz darba autora prakses vietas plānotajiem darbiem.

Darba tēma ir aktuāla, jo testēšanas automatizācija ir būtiska programmatūras izstrādes procesa testēšanas fāzes sastāvdaļa, saistībā ar to ka tā palīdz testētājam veikt ātru un drošu testēšanu, nepieļaujot kļūdas, kas var rasties no cilvēciskā faktora, un ka arī tas pašam uzņēmumam samazina izmaksas, salīdzinājumā ja testēšana tiktu veikta manuāli.

Kvalifikācijas darbs ir strukturēts 7 nodaļās. Pirmajā nodaļā ir aprakstīta izstrādājamo automatizēto testu skriptu programmatūras prasību specifikācija, otrajā nodaļā ir aprakstīts programmatūras projektējuma apraksts, trešajā – testēšanas dokumentācija. Ceturtā nodaļa satur darba organizācijas aprakstu, piektajā ir aprakstīts tas, kā darbā tika aprakstīti projektā pielietotie kvalitātes nodrošināšanas mehānismi, bet sestajā nodaļā ir aprakstīta konfigurācijas pārvaldība. Septītā nodaļa satur darbietilpības novērtējumu.

1. PROGRAMMATŪRAS PRASĪBU SPECIFIKĀCIJA

1.1. Ievads

1.1.1. Nolūks

Programmatūras prasību specifikācija (PPS) ir paredzēta izstrādājamās Telos VX funkcionālo testu automatizācijas programmatūras prasību aprakstīšanai. Atbilstoši programmas prasību specifikācijai tiks izstrādāts arī programmas projektējuma apraksts (PPA) un funkcionējošs programmkods. Šis dokuments ir paredzēts programmatūras pasūtītājiem, lai varētu precīzi noformulēt nepieciešamās prasības pret programmatūru, un programmatūras izstrādātājiem.

1.1.2. Darbības sfēra

Telos VX[1] funkcionālās testēšanas automatizācijas programmatūras mērķis ir paātrināt un atvieglot testēšanas procesu Telos VX funkcionalitātes un tīmekļa vietnes lietotāja saskarnes darbību pārbaudē, kas nodrošinās vienkāršāku un ātrāku iespējamo kļūdu atrašanu testētājiem un to atrisināšanu iekārtas izstrādātājiem.

1.1.3. Definīcijas, akronīmi un saīsinājumi

- Iekārta - Telos VX
- Sistēma - radio studijas vadības sistēma, kur viena no šīs sistēmas sastāvdaļām ir Telos VX
- Lietotājs - persona vai personas, kas izmantos izstrādāto programmatūru, lai testētu iekārtu;
- Programmatūra – testēšanas automatizācijas skripti

1.1.4. Saistība ar citiem dokumentiem

Dokuments tika izstrādāts vadoties pēc LVS 68:1996 „Programmatūras prasību specifikācijas ceļvedis” standarta prasībām.

1.1.5. Pārskats

Šajā dokumentā iekļauts vispārējais apraksts, kurā ir aprakstītas produkta funkcijas, lietotāja raksturiezīmes, vispārējie ierobežojumi un pieņēmumi un atkarības, konkrēto prasību apraksts, kurā ir aprakstītas funkcionālās un nefunkcionālās prasības. Nefunkcionālās funkcijās ietilpsts veikspējas un uzturamības prasības.

1.2. VISPĀRĒJAIS APRAKSTS

1.2.1. Produkta perspektīva

Testējamā iekārta ir viena no radio studijas apraides sistēmas sastāvdaļām. Izstrādājamā testēšanas programmatūra ir neatkarīga un pašpietiekama.

1.2.2. Produkta funkcijas

Dotajai programmatūrai ir jānodrošina sekojoša funkcionalitāte:

- Iekārtas tīmekļa vietnes lietotāju saskarnes iestatījumu uzstādīšana un iestatīto vērtību pārbaudīšana;
- Jāpārbauda vai veicot noteiktas darbības ar iekārtu, tiek izdoti attiecīgie iekārtas paziņojumi.

1.2.3. Lietotāja raksturiezīmes

Programmatūras lietotājs ir šīs iekārtas testētājs. Testētājam ir jābūt zināšanām par testējamo sistēmu, jāpārzina izmantotā programmēšanas valoda un jāprot interpretēt iegūtie rezultāti. Vajadzības gadījumā jāprot rediģēt programmatūras pirmkods.

1.2.4. Vispārējie ierobežojumi

Programmatūras izstrādei ir nepieciešams tiešs pieslēgums testējamajai iekārtai. Nepieciešams, lai iekārta jau būtu uzstādīta lietošanai. Izstrādājamai programmatūrai ir jābūt pieejamai tikai tās lietotājiem sistēmas iekšējā tīklā, kurš nav pieejams no ārpuses.

1.2.5. Pieņēmumi un atkarības

Tiek pieņemts, ka lietotājam ir zināšanas par VX iekārtas darbību un interpretatora veidoto rezultātu analīzi balstoties uz testa scenāriju. Tiek pieņemts, ka lietotājam ir tieša pieja VX, kā arī lietotājam ir pieejams interpretators vai iespēja to izmantot.

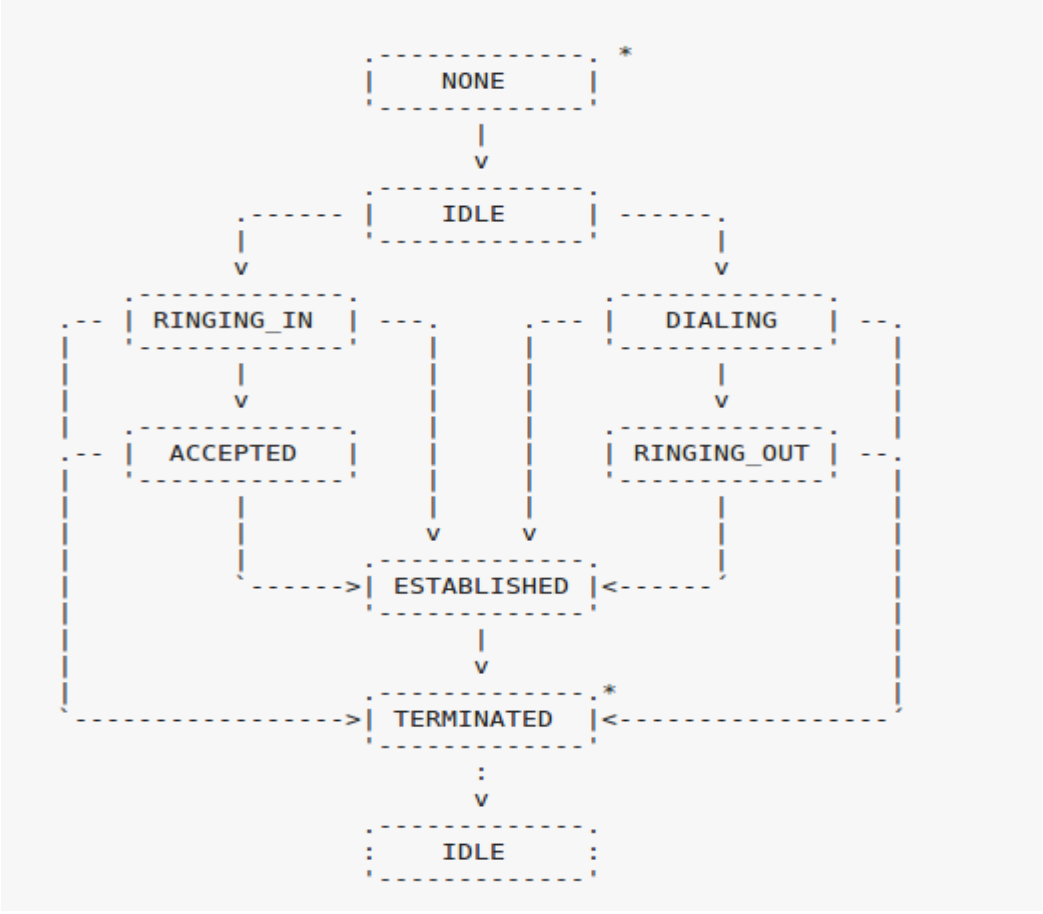
1.3. KONKRĒTĀS PRASĪBAS

1.3.1. Funkcionālās prasības

Funkcionālās prasības programmatūras izstrādātājam no darba devēja un programmatūras pasūtītāja tika apkopotas lietotāju stāstu formātā.

1.3.1.1. LS_1 Zvanu funkcionalitātes automatizācija

Programmatūras pasūtītājs vēlas, lai tiktu izstrādāta programmatūra, kas ļautu tās lietotājam veikt pilnīgu iekārtas zvanu funkcionalitātes pārbaudi. Šī prasība izpaužas, kā nepieciešamība, lai lietotājs izmantojot izstrādāto programmatūru var pārbaudīt no iedotās shēmas (Skat. 1.3.1.1. att.)[3] katru zvanu stāvokļu ceļu uz vienas studiju šovu telefonlīnijas.



1.3.1.1.1. att. **Telos VX zvanu stāvokļu diagramma**

Akceptēšanas kritēriji – lietotājs izmantojot izstrādāto programmatūru var veikt iekārtas zvanu pilnīgu zvanu funkcionalitātes pārbaudi.

1.3.1.2. LS_2 Zvanu stāvokļu automatizētu pārbaužu veikšana uz vairākām studiju šovu līnijām

Programmatūras pasūtītājs vēlas, lai 1.3.1.1. nodaļā minētā programmatūra tiktu papildināta, tā, lai ļautu tās lietotājam veikt iepriekš minēto zvanu stāvokļu pārbaudes uz vairākām studiju šovu telefonlīnijām vienlaicīgi, kas nodrošinātu zvanu funkcionalitātes veikspējas testēšanu.

Akceptēšanas kritēriji – lietotājs izmantojot papildināto programmatūru var veikt iekārtas zvanu stāvokļu pārbaudes uz vairākām studiju šovu līnijām vienlaicīgi.

1.3.1.3. LS_3 Zvanu stāvokļu automatizētu pārbaužu veikšana uz nejauši izvēlētām studiju šovu līnijām

Programmatūras pasūtītājs vēlas, lai 1.3.1.1. nodaļā minētā programmatūra tiktu papildināta, tā, lai ļautu tās lietotājam veikt iepriekš minēto zvanu stāvokļu pārbaudes uz nejauši izvēlētām vairākām studiju šovu telefonlīnijām vienlaicīgi vai arī tikai uz vienas līnijas.

Akceptēšanas kritēriji – lietotājs izmantojot papildināto programmatūru var veikt iekārtas zvanu stāvokļu pārbaudes uz nejauši izvēlētām vairākām studiju šovu telefonlīnijām vienlaicīgi vai arī tikai uz vienas telefonlīnijas.

1.3.1.4. LS_4 Studiju konfigurāciju automatizācija

Programmatūras pasūtītājs vēlas, lai tiktu izstrādāta programmatūra, kas ļautu tās lietotājam veikt iekārtas studiju konfigurācijas automatizēšanu caur tīmekļa vietnes lietotāja saskarni. Papildus prasība šai programmatūrai daļai būtu spējai tos apvienot ar zvanu funkcionalitātes programmatūru.

Akceptēšanas kritēriji – lietotājs izmantojot izstrādāto programmatūru var veikt iekārtas studiju konfigurācijas automatizēšanu caur tīmekļa vietnes lietotāja saskarni.

1.3.2. Nefunkcionālās prasības

1.3.2.1. Veikspējas prasības

Lai iespējami paātrinātu testēšanas procesu un lieki nenoslogotu tīklu, testa skriptiem jāizpildās iespējami īsākā laikā.

1.3.2.2. Uzturamības prasības

Programmatūrai ir jābūt viegli modificējamai un struktūrēti sadalītai mazos apakšskriptos, lai kļūdu gadījumos spētu programmatūras izstrādātājs noteikt izraisīto kļūdu vainas un tās atrisināt. Programmatūras kodam ir jābūt komentētam tā, lai atklātas kļūdas gadījumā būtu viegli noteikt, tieši kurā vietā kļūda ir meklējama. Programmatūras koda komentāriem ir jābūt angļu valodā, jo izstrādātāju komanda ir starptautiska un pieņemtā saziņas valoda ir angļu.

2. PROGRAMMATŪRAS PROJEKTĒJUMA APRAKSTS

2.1. Programmtūras uzbūve

Atbilstoši PPS izvirzītajām prasībām programmatūrai jābūt strukturēti dalītai vairākās mapēs, lai pirmkods būtu sakārtots un organizēts. Kods ir jāorganizē sekojoši – pirmkārt, visi testu scenāriji atrodas vienā mapē, kuri pēc tam tiek dalīti apakšmapēs atkarībā no testu scenāriju kopīgajām iezīmēm, lai būtu ērtāk tos atrast un izmantot. Otrkārt, lai nepieciešamības gadījumā varētu noteikt kurā funkcionalitātes testēšanas posmā ir kļūme, jāizveido katrai zvanu stāvokļu pārbaudei savs skripts un šiem skriptiem ir jāatrodas atsevišķā mapē. Treškārt VX[1] studiju konfigurāciju skriptiem un SIPp[4] konfigurāciju XML datnēm arī jāatrodas atsevišķi katrām savā mapē. Tas ir nepieciešams, lai nodrošinātu daudz ātrāku pārbaudi, piemēram, pārtestējot tikai atsevišķas testu scenāriju komponentes, mainoties kādai no iekārtas funkcijām, kas var ietekmēt tikai atsevišķas testa scenārija sastāvdaļas.

Svarīgi nodrošināt, lai katrs testa scenārijs būtu neatkarīgs viens no otra, t.i. nedrīkst rasties situācija, kad iepriekšējam testa scenārijam beidzoties nākošais nedarbojas, jo iekārta nav pareizi iestatīta. Vislabāk to nodrošināt, katru testa scenāriju uzsākot no noteikta, universāli izmantojama sākuma stāvokļa un scenārija beigās atgriezt iekārtu atpakaļ šajā stāvoklī.

Programmatūrai ir jābūt sadalītai divos moduļos. Pirmais modulis ir studiju konfigurāciju modulis, kas ir nepieciešams, lai atvieglotu testa scenāriju startēšanu. Šis modulis nodrošina, ka iekārta tiek nokonfigurēta skripta iekšienē. Bez šīs konfigurācijas daļas, studiju konfigurēšana būtu jāveic manuāli, kas var būt laikietilpīgs process. Otrais modulis ir pārbaudes daļas modulis. Šis modulis nodrošina Telos VX funkcionalitātes pārbaudi. Šiem abiem moduļiem būtu jābūt pēc iespējas neatkarīgiem vienam no otra, lai tos varētu lietot atsevišķi, piemēram izmantot konfigurācijas moduli iekārtas citas funkcionalitātes testēšanai.

2.2. Programmatūras izstrādes vides un tās tehniskais risinājums

Programmatūra var tikt izstrādāta vai nu uz Windows 10, Linux Mint 18.3 vai Ubuntu Linux 18.04 operētājsistēmas un programmkoda var tikt izmantots jebkurš teksta redaktors, kas ir specializēti izstrādāts programmatūru pirmkodu rakstīšanai un rediģēšanai. Uz attiecīgā datora jābūt arī installētam SIP[4] telefona lietotnes "SIPp" 3.5.1. vai jaunākai versijai un automatizēto testu skriptu interpretatora 1.1.6. vai jaunākai versijai[2].

Izstrādājamās programmatūras pirmkods ir jāraksta programmēšanas valodā, kura ir paredzēta minētajam automatizēto testu skriptu interpretatoram. Šī ir īpaši izveidota specializēta valoda, izveidota uz programmēšanas valodas Ruby bāzes, kurā ērti lietojamā veidā iespējams automatizēti pārbaudīt gala programmatūras un aparatūras darbību. Speciāli šai valodai programmatūras pirmkodam ir jābūt rakstītam teksta datnēs ar paplašinājumu "lwscen". Šī valoda nemitīgi attīstās sniedzot jaunu funkcionalitāti, pieaugot testēšanas veidu izvēlei un ir izveidota pēc iespējas vienkāršāk, skatoties no lietošanas skatupunkta, lai šo valodu spētu lietot cilvēki ar minimālu programmēšanas pieredzi un lai tiktu nodrošināta laba lasāmība[2].

Valodas galvenās funkcijas, lai veiktu nepieciešamās programmatūras izstrādi:

- Izmantot vienkāršas programmēšanas valodu kontrukcijas – mainīgos, ciklus, komentārus
- Izmantot dažādas šīs valodas iebūvētās palīgfunkcijas piem. aizture
- Savienojuma izveide ar iekārtu
- Programmatūras/aparatūras stāvokļa pārbaude
- Palaist programmas vai testa apakšskriptus no termināļa
- Darboties ar HTML formām
- Izmantot HTTP GET un PUT metodes
- Darboties ar XML failiem

SIPp[4] ir atklātā pirmkoda programmatūra, kas radīta SIP protokola ierīču testēšanai. Šī programma spēj atbildēt uz ienākošiem zvaniem vai veikt zvanus, kas ir nepieciešams, lai varētu veikt pilnīgu Telos VX zvanu funkcionalitātes automatizēto testēšanu.

2.3. Studiju konfigurācijas moduļa uzbūve

Šis modulis sastāv no Telos VX[1] studijas konfigurācijas daļas skriptiem, kuri domāti, lai uzstādītu lietotāja izvēlētas vērtības kādai no Telos VX iekārtas studijām caur iekārtas tīmekļa vietnes lietotāja saskarni.

Šiem skriptiem ir jābūt izstrādātiem sekojoši - pirmkārt tiem ir iespēju robežās jābūt pilnīgiem – visai konfigurācijai jānotiek no skripta, jo process var aizņemt ļoti daudz laika, ja jāveic manuāla konfigurācija. Otrkārt, ir jāizveido skripts, kas spēj uzstādīt studiju atpakaļ sākotnējā, noklusētā stāvoklī, lai spētu veikt vairākkārtēju skriptu testu scenāriju palaišanu, bez nekādas manuālas konfigurācijas veikšanas. Treškārt, šiem skriptiem jābūt nodalītiem tā, ka katrs skripts veic vienas HTML formas konfigurāciju, lai spētu ātri noteikt, kurā no HTML formām ir ieviesušies kļūda, uzstādot vērtības. Ceturtkārt, skriptiem ir jābūt strukturētiem tā, ka skripti sākas ar lietotāja izvēlēto vērtību uzstādīšanu, aiz kuras uzreiz seko šo vērtību saglabāšana un pēc tam šo vērtību lauku pārbaude. Tas ir nepieciešams, lai pārliecinātos vai izvēlētās vērtības ir patiešām uzstādītas un lai ātrāk atrastu kļūdas konfigurācijā.

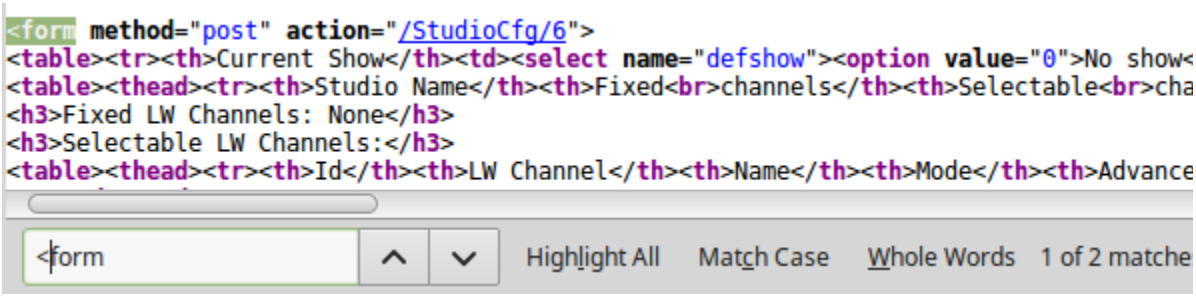
2.3.1. Nepieciešamās HTML formu informācijas noskaidrošana

Lai zinātu, kuru Telos VX[1] studiju ir nepieciešams konfigurēt, ir jāzina šīs studijas unikālais identifikātors jeb ID. To var noteikt, atverot interneta pārlūku un ievadot pārlūka meklētājā iekārtas IP adresi. Kad parlūks atver iekārtas tīmekļa vietni ir jāveic lietotāja authentifikācija. Pēc tam ir jāuzspiež uz sadaļas “Studios”. Pēc šīs darbības parādās saraksts ar izveidotajām studijām un atliek vien izvēlēties kādu no pieejamajām studijām uzkličinot uz kādas no tām ar peles klišķi. Studijas ID var nolasīt no pārlūka meklētāja, kurš atrodas pēc simbolu virknes “/StudioCfg/”. 2.3.1.1. attēlā ir redzams, ka izvēlētās studijas ID ir 6.

2.3.1.1. att. **Telos VX studijas ID noteikšana**

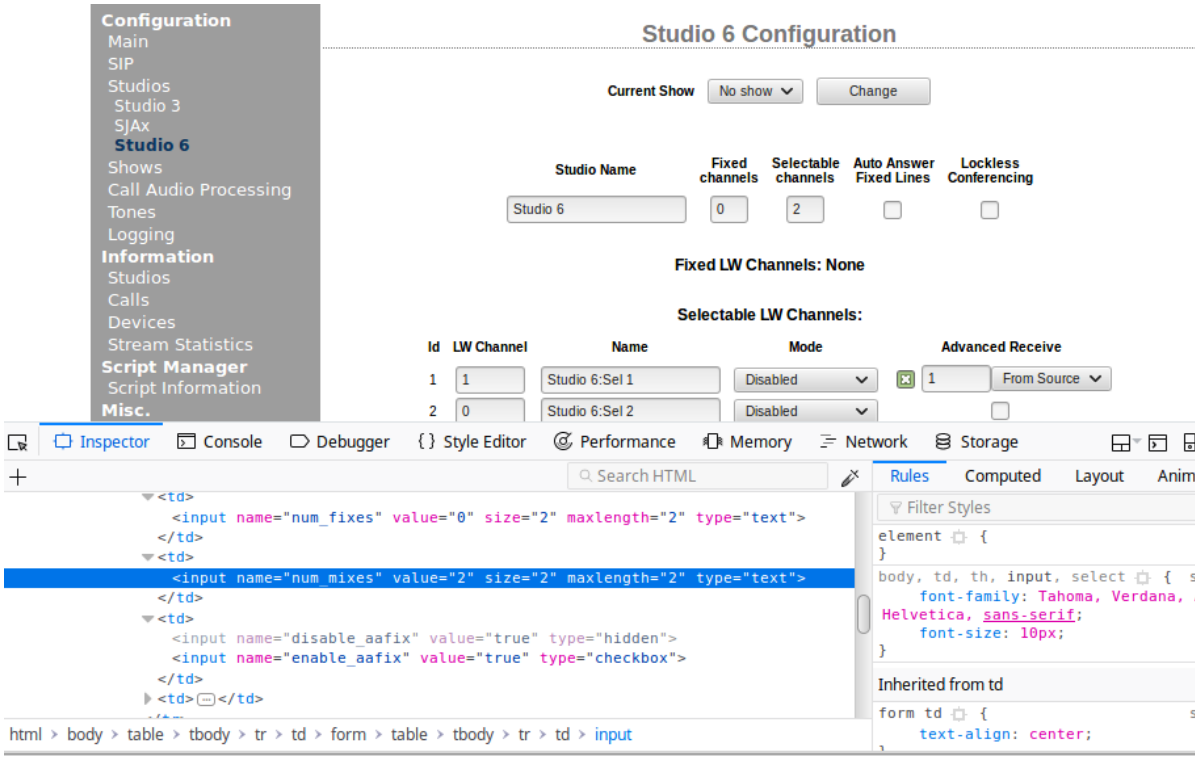
Katras studijas tīmekļa lapa sastāv no vienas vai vairākām lauku grupām (form). Ja lapa sastāv novairāk kā vienas lauku grupas, tad, lai ar testa skriptu palīdzību piekļūtu noteiktam laukam, ir vispirms jānoskaidro, kurā grupā šis lauks atrodas. Tas ir jādara manuāli. Jāatrodas

ir iepriekš aprakstītās studiju konfigurācijas lapā un jāuzklikšķina uz tās nospiežot peles labo taustiņu, un no izvēlnes, kura parādās, izvēlas “View page source”. Atveras lapa, kurā ir redzams attiecīgās tīmekļa lapas pirmkods. Tad lieto klaviatūras komandu ctrl+f, kas izsauc meklēšanas lauku, kurš parādās lapas labajā augšējā stūrī. Meklēšanas laukā ievada “<form”(bez pēdiņām) un uz klaviatūras nospiež pogu “Enter”. Meklēšanas logā būs redzams, cik lauku grupas(forms) ir atrastas. Ja ir atrastas vairākas lauku grupas, tad lapas pirmkodā ir jāatrod un jāizskaita, kurā tieši lauku grupā atrodas testējamais lauks (to dara meklējot lauka nosaukumu, kurš ir minēts tīmekļa lapas lietotāja saskarnē. Ja ir atrasta tikai viena lauku grupa, tas nozīmē, ka visiem lapas laukiem var piekļūt ar vienu lauku grupu. Numerācija lauku grupām sākas ar 0. 2.3.1.2. attēlā ir redzams kā ar teksta meklētāju ir atrasta viena no divām HTML formām.



2.3.1.2. att. **HTML formu skaita noteikšana**

Lai varētu piekļūt HTML formu laukiem, ir nepieciešams noskaidrot lauku HTML nosaukumus. To dara izvēlētās studijas tīmekļa lapu un, uzklikšķinot uz interesējošā lauka ar peles labo taustiņu, izvēlas “Inspect”. Atveras logs, kurā redzams HTML kods, kurš attiecas uz izvēlēto lauku un tas ir iezīmēts zilā krāsā(Skat. 2.3.1.3. attēls). 2.3.1.3. attēlā ir redzams, ka iezīmētais HTML formas lauks ir paredzēts teksta ievadei, kas spēj saturēt maksimums līdz 2 simboliem un tā nosaukums ir “num_mixes” un tā vērtība ir 2. Tīmekļa vietnes lietotāju saskarnē tas atbilst HTML formas ievades laukam ar nosaukumu “Selectable channels”.



2.3.1.3. att. **HTML formu elementu noteikšana**

2.3.2. Studiju konfigurācija izmantojot REST API

Ņemot vērā, ka dažiem HTML formu elementiem nav iespējams piekļūt izmantojot 2.2. nodaļā aprakstīto programmēšanas valodu, pastāv alternatīva Telos VX studiju livewire kanālu konfigurāciju veikt izmantojot REST API. Katrai izveidotai studijai ir pieejama livewire kanālu konfigurācijas datne, kura ir aprakstīta XML datnes formātā. Šīm XML datnēm ir iespējams piekļūt atverot jebkuru Telos VX studiju konfigurācijas lapu un url meklētājā vietnes beigās pierakstīt "/config.xml". Izmantojot REST API var šo XML datņu saturu mainīt, lai konfigurētu Telos VX studijas fiksētos vai izvēlējamos livewire kanālus, kas ļauj veikt to automatizāciju no testu skriptu izstrādes interpretatora. 2.3.2.1. tabulā ir redzami Telos VX studiju konfigurāciju XML datnē lietotie tagi un to skaidrojums.

2. 3.2.1. Tabula **- Studiju konfigurēšanas XML failu tagi un to skaidrojums**

XML tagu nosaukums	Tagu skaidrojums

<ch_fix> ... </ch_fix>	Tags norāda fiksēto livewire kanālu sadaļu studijas konfigurācijas tīmekļa vietnē.
<ch_sel> ... </ch_sel>	Tags norāda izvēlējamo livewire kanālu sadaļu studijas konfigurācijas tīmekļa vietnē.
<channel> ... </channel>	Tags norādi viena livewire kanāla sekciju studijas konfigurācijas tīmekļa vietnē.
<sendnumber> ... </sendnumber>	Tags norāda livewire sūtītāja kanāla numuru.
<recvnumber> ... </recvnumber>	Tags norāda livewire saņēmēja kanāla numuru.
<mode> ... </mode>	Tags norāda kādā režīmā livewire kanālā plūst skaņas kvalitāte.
<disabled> ... </disabled>	Tags norāda vai skaņas kvalitātes režīms ir iespējots vai nē.

2.4. Pārbaudes daļas moduļa uzbūve

Šis modulis sastāv no Telos VX zvanu stāvokļu pārbaudes skriptiem, kuri domāti, lai šīs programmatūras lietotājs spētu veikt pilnīgu šīs iekārtas zvanu funkcionalitātes testēšanu.

Modulim jābūt strukturētam tā, ka katrai zvanu stāvokļu pārbaudei ir izveidots savs skripts, lai kļūdu gadījumā būtu vieglāk un ātrāk noteikt, kurā stāvokļu pārbaudes skriptā ir problēma meklējama un tiem jābūt izveidotiem tā, ka tos var izmantot atkārtoti citem testu scenārijiem.

2.4.1. VX LWCP protokols

LiveWire control protocol (LWCP) ir uz tekstu balstīts, orientēts uz ziņojumiem protokols komunikācijai starp dažādām ierīcēm studijas vidē. Pamata protokola vienība ir ziņa, kas sastāv no darbības, kam seko izvēlējams objekts un nulle vai vairāk atribūtu. Ar šo protokolu ir iespējams likt Telos VX veikt dažādas darbības un šos procesus var automatizēt

izmantojot 2.2. nodaļā minēto interpretatoru. Sākotnēji, lai izprastu kā šis protokols darbojas, tas ir manuāli jāizpēta. Tabulā 2.4.1.1. ir redzamas nepieciešamās LWCP protokola komandas, lai veiktu zvanu stāvokļu pārbaudi izmantojot interpretatoru. Atbiltoši vienai no PPS izvirzītajām funkcionālajām prasībām VX spēj pēc šī protokola, saņemt un veikt vairākus zvanus, ja vien tie ir katrs atsevišķi saslēgti studijas šovā. To var izdarīt izmantojot komandu „lock studio.line#<telefonlīnijas id>", un lai līnijas tiktu atslēgtas jāizmanto komanda „unlock studio.line#<telefonlīnijas id>".[3]

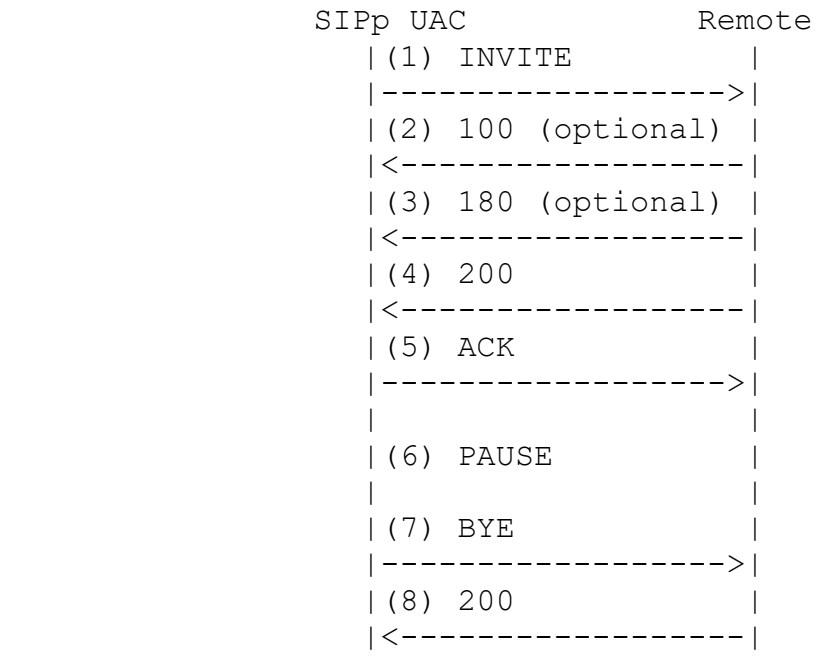
2. 4.1.1. Tabula - **LWCP komandas un to skaidrojums**

LWCP komanda	Komandas skaidrojums
telnet <iekārtas ip adrese> 20518	Savienojuma izveidošana ar iekārtu. 20518 ir noklusētais ports
login cc user = "user", password = ""	Pieslēgšanās pie iekārtas. "user" un "" ir noklusētās vērtības.
select studio id = <studijas id>	Tiek izvēlēta studija
select_show studio id = <šova id>	Tiek izvēlēts studijai šovs
get studio.line#<telefonlīnijas id> callstate	Nosaka izvēlētās telefonlīnijas zvanu stāvokli
take studio.line#<telefonlīnijas id>	Tiek pacelts izvēlētā telefonlīnija ja uz to zvana
drop studio.line#<telefonlīnijas id>	Tiek nomesta izvēlētās telefonlīnijas zvans
call studio.line#<telefonlīnijas id> number="telefona lietotnes numurs"	Tiek veikts zvans no izvēlētās telefonlīnijas uz norādīto telefonnumuru.

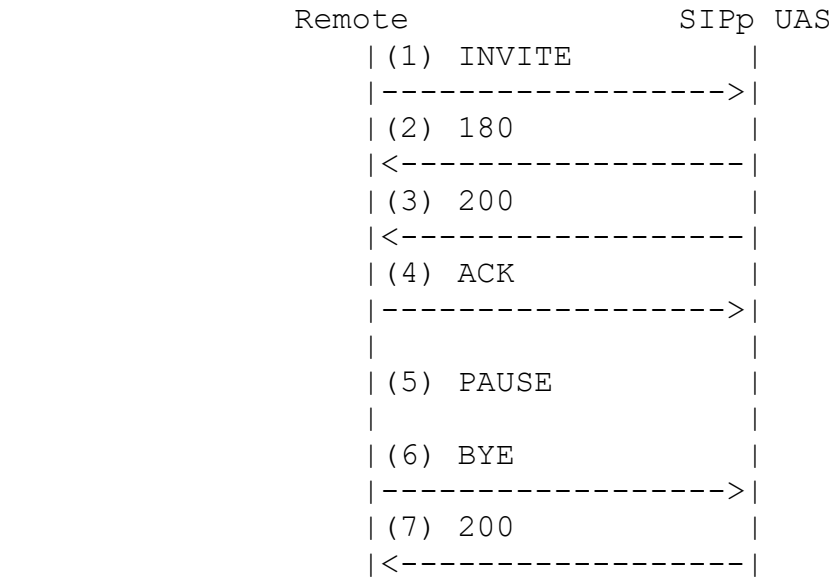
2.4.2. SIPp lietošana

Lai SIPp izmantotu tam, analogi kā interpretatoram, ir nepieciešams padot datni, kurā ir aprakstītas komandas, kas norāda kādas darbības šai lietotnei ir jāveic. Šai datnei ir jābūt ar paplašinājumu „.xml". Būtiski, ka būtu jāizveido 2 xml datnes, jo SIPp spēj katrā instancē būt vai nu kā zvanu ģenerators vai zvanu atbildētājs tikai, nevis abi vienlaikus. Kā arī svarīgi, SIPp xml datnēm jābūt savietojamām ar testu scenāriju skriptiem, proti, tos var automatizēt un

izmantot citos testu scenārijos veicot minmālas izmaiņas tajos tikai pamainot parametru vērtības, kas tie padotas, kad SIPp tiek izsaukts. Attēlos 2.4.2.1. un 2.4.2.2. ir redzami noklusētie SIPp zvanu scenāriji.[4]



2.4.2.1. att**. SIPp noklusētajā zvanu ģenerēšanas stāvoklī**



2.4.2.2. att. **SIPp noklusētajā zvanu atbildēšanas režīmā**

3.TESTĒŠANAS DOKUMENTĀCIJA

Programmatūras testēšana tiek veikta manuāli, laižot testa scenāriju skriptus caur interpretātoru, salīdzinot to sniegtos rezultātus ar paredzētāmajiem rezultātiem. Skripta palaišana ir aprakstīta interpretatora valodas manuālī. Kad skripts tiek palaists pastāv divas iespējamības, ka vai nu skripts nepalaižas tad terminālī ir izlasāms paziņojums par sastapto problēmu un ir jāveic vajadzīgās darbības, lai to novērstu, vai ka skripts palaižas un terminālī ir redzamas pārbaudes kas dalās OK zaļā krāsā vai NOK sarkanā krāsā, kā arī ir redzami izvades komandas no skripta, kas ļauj lietotājam redzēt, kādas darbības šis skripts veic (skat. 3.1. att.) Paralēli skriptam palaižoties, interpretators ģenerē datnes ar paplašinājumiem „.log" un „.rez". „.rez" datnē ir redzama tā pati izvade, kas ir redzama no termināļa, kamēr „.log" datne sniedz plašāku ieskatu, kas notiek, kad skripts tiek palaists caur interpretatoru, piemēram, kādi paziņojumi tiek izvadīti no iekārtas socketa jeb ligzdas pie kā interpretators ir pieslēdzies. Šīs datnes atvieglina kļūdu atrašanu, neveiksmju gadījumā.[3]

```
------------------------TEST TERMINATED callstate-----------------------------
Running: 'ruby "/home/viesis/Desktop/interpret/main.rb"...
Check if studio line:4 callstate is TERMINATED
  54: OK  TERMINATED
Check if studio line:5 callstate is TERMINATED
  54: OK  TERMINATED
Check if studio line:6 callstate is TERMINATED
  54: OK  TERMINATED
Check if studio line:7 callstate is TERMINATED
  54: OK  TERMINATED
Check if studio line:8 callstate is TERMINATED
  54: NOK TERMINATED
Completed: ../callstates/TERMINATED_UNLOCK.lwscen; result:false
------------------------TEST IDLE callstate-----------------------------------
Running: 'ruby "/home/viesis/Desktop/interpret/main.rb"...
Check if studio line:4 is in callstate IDLE
  65: OK  IDLE
Check if studio line:5 is in callstate IDLE
  65: OK  IDLE
Check if studio line:6 is in callstate IDLE
  65: OK  IDLE
Check if studio line:7 is in callstate IDLE
  65: OK  IDLE
```

3. 1. att. **Skripta palaišana no termināļa**

Ja testu scenārijos ir paziņojumi par kļūdām, tad tiek noteikts kurā scenārija daļā kļūda eksistē. Tā ir vai nu studiju konfigurēšanas vai zvanu stāvokļu pārbaužu daļā. Ja kļūda ir studiju konfigurēšana daļā, tad izmantojot interpretatora valodas iespēju, skriptā zvanu stāvokļu daļa tiek aizkomentēta un atverot Telos VX tīmekļa vietnes studiju sadaļu un

izvēloties attiecīgo studiju, kas tika izmantota, lai palaistu skriptu(skat. 2.3.1. nodaļu), tiek noteikts, kurā HTML formas elementā ir ieviesušies kļūda. Tad tiek manuāli pārbaudīts ierakstot attiecīgo vērtību un nospiežot attiecīgo konfigurācijas saglabāšanas pogu. Ja lapā tiek saglabāta neadekvāta vērtība, tas nozīmē, ka testējamajā programmatūrā ir atrasta kļūda. Ja tiek saglabāta pareiza vērtība, tad tas nozīmē, ka ir problēma ar pašu skriptu vai interpretatoru. Ja kļūda ir zvanu stāvokļu pārbaužu daļā, tad studiju konfigurēšanas daļa tiek aizkomentēta skriptam un manuāli tiek attiecīgajai studijai iestatīta attiecīgās vērtības. Tālāk tiek atvērts jauns terminālis un izmantojot telnetu tiek veikta pieslēgšanās pie iekārtas un laižot skriptu vēlreiz tiek rūpīgi novērots kādus paziņojumus izdod sockets jeb ligzda terminālļa logā (skat. 2.4.1. att.). Šajos kļūdu gadījumos problēmas var tikt saistītas ar SIPp telefona lietotnes funkcionalitāti, pašu skriptu vai interpretatoru, vai arī iekārtas funkcionalitātē.

Programmatūras testēšana notika abpusēji - gan no darba autora, gan no programmatūras pasūtītāja puses. Kad tika izstrādāts strādājošs testu scenāriju kopums, skripti tika notestēti, lai nebūtu nekādu kļūdu un nodoti programmatūras pasūtītājiem. Tabulā 3.1. ir redzams testu scenāriju testēšanas žurnāls, kurā katrs scenārijs tika testēts savā pēdējā izstrādes versijā. Ja testa scenārijs izgāja cauri bez kļūdām, tad tam ir ielikts statuss OK, ja nav tad tam ir ielikts statuss NOK, un tas tika pārrakstīts un testēts citā dienā.

3.1.Tabula -**Testēšanas žurnāls**

Testa scenārijs	**Datums**	**Statuss**	**Piezīme**
individual_line_test_case_1.lwscen	07.05.2018	OK	
individual_line_test_case_2.lwscen	07.05.2018	OK	
individual_line_test_case_3.lwscen	07.05.2018	OK	
Individual_line_test_case_4.lwscen	07.05.2018	OK	
individual_line_test_case_5.lwscen	07.05.2018	OK	
individual_line_test_case_6.lwscen	07.05.2018	OK	
individual_line_test_case_7.lwscen	07.05.2018	OK	
individual_line_test_case_8.lwscen	07.05.2018	OK	

multiple_line_test_case_1.lwscen	24.05.2018	OK	
multiple_line_test_case_2.lwscen	24.05.2018	OK	
multiple_line_test_case_3.lwscen	24.05.2018	OK	
multiple_line_test_case_4.lwscen	24.05.2018	OK	
multiple_line_test_case_5.lwscen	24.05.2018	OK	
multiple_line_test_case_6.lwscen	24.05.2018	OK	
multiple_line_test_case_7.lwscen	24.05.2018	OK	
multiple_line_test_case_8.lwscen	24.05.2018	OK	
random_individual_line_tests.lwscen	24.05.2018	NOK	Vairākas reizes atkārtojot nejauši izvēlētu testa scenāriju, dažas zvanu stāvokļu pārbaudes izdod NOK. Kļūdas tiek saistītas ar nepareizu parametru padošanu SIPp.
random_multiple_line_tests.lwscen	24.05.2018	NOK	Vairākas reizes atkārtojot nejauši izvēlētu testa scenāriju, dažas zvanu stāvokļu pārbaudes izdod NOK. Kļūdas tiek saistītas ar nepareizu

			parametru padošanu SIPp.
random_individual_line_tests.lwscen	25.05.2018	OK	
random_multiple_line_tests.lwscen	25.05.2018	OK	

4. PROJEKTA ORGANIZĀCIJA

Programmatūra tika izstrādāta pēc modificēta ūdenskrituma modeļa. Izstrādes sākumā bija zināmas dažas prasības par to, kādas funkcijas izstrādājamajai programmatūrai ir jāveic. Iteraktīvi tika veidots programmatūras PPS un PPA un veikta kodēšana. Pieaugot prasībām pret izstrādājamo programmatūru, pirmkods tika mainīts vairākas reizes. Pirmkods arī tika modificēts balstoties pēc tīmekļa vietnes Red Mine iesniegtajām sūdzībām. Programmatūras izstrādi būtiski palēnināja automatizētās testēšanas testu skriptu interpretators, saistībā ar to, ka tam nebija nepieciešamās funkcionalitātes, ar kuru varētu noprogrammēt programmatūru atbilstoši izvirzītajām prasībām, kura rezultātā interpretatoram tika pieliktas jaunas iespējas un atbilstoši izvirzītajām prasībām atrast nepieciešamo SIP telefona lietotni, lai automatizētu Telos VX funcionalitātes testēšanu. Sākotnēji prasību apmierināšanai pietika ar Linphone telefona lietotni, bet pieaugot prasībām nolemts bija lietot SIPp telefona lietotni.

5. KVALITĀTES NODROŠINĀŠANA

Programmatūra ir viegli modificējama nepieciešamības gadījumā – pirmkods ir komentēts un strukturēts balstoties testa scenārija aprakstā, un ja scenārijā tiek veiktas izmaiņas, pirmkodā ar teksta redaktora meklēšanas funkciju ir iespējams viegli atrast nepieciešamo pārbaudes soli un to izmainīt balstoties uz jaunajās prasībās.

Izstrādājot programmatūru tika ņemti vērā ieteikumi no izstrādes vietā strādājošiem zinošiem programmētājiem un testētājiem. Katrai programmatūras vienībai, jeb automatizētās testēšanas skriptam, pirmkoda sākumā ir komentāru bloks, kurā ir norādīts:

- Skripta nosaukums, tā izveides datums un autors.
- Kopsavilkums un sīkāks apraksts, ko šis skripts dara.
- Kā šo skriptu palaist no termināļa
- Parametru apraksts, kas tiek padoti palaižot skriptu
- Skripta veidošanas vēsture

6. KONFIGURĀCIJAS PĀRVALDĪBA

Par versiju pārvaldības rīku tika izvēlēts darba devēja iedotais Git un privāta repozitorija glabātuve – Github. Tas nodrošina sistēmas pirmkoda un dokumentācijas drošu glabāšanu un pieejamību. Sistēmas pirmkods repozitorijam tika pievienots, pēc katru neatkarīgu lietojamu skriptu kopuma izveides un pēc katras . Izstrāde notika darba vietā, izmantojot darba vietas datoru. Izstrādātā programmatūra tika rakstīta izmantojot Sublime text 3 teksta redaktoru uz Linux Mint 18.3 operētājsistēmas.

7. DARBIETILPĪBAS NOVĒRTĒJUMS

. Kā orientējošais laiks šī kvalifikācijas darba izveidei bija 3 personmēneši, kas sevī ietver vai nu 480 astronomiskās stundas (60 min) vai 60 darba dienas(8h). Autors izvēlējās darbietilpību novērtēt izmantojot eksperta metodi un darāmos darbus mērīt dienās. Darbietilpības novērtēšanai ņemu vērā optimālo, reālo un pesimistisko dienu skaitu noteiktā darba uzdevuma veikšanai pēc formulas (Optimālais stundu skaits + 4 * Reālistiskais stundu skaits + Pesimistiskais stundu skaits) / 6. Projekta posmu un paveikto darbu darbietilpība parādīta tabulā 7.1.

7.1. Tabula – **Darbietilpības novērtējums**

Darba uzdevums	**Optimistikais novērtējums**	**Pesimistiskais novērtējums**	**Reālistiskais novērtējums**	**Reāli patērētais laiks**
Funkcionalitātes automatizācijas testu skriptu rakstīšana	23	29	25	25.33
Lietotāja saskarnes automatizācijas testu skriptu rakstīšana	6	12	10	9.66
Testēšana	4	14	9	9
Testu skriptu kļūdu labošana no iesniegtajām sūdzībām tīmekļa vietnē Redmine.	5	11	8	8
Dokumentācijas rakstīšana	13	22	17	17.16

Kopā (dienas):	51	88	69	69.16
Personmēneši kopā:	2.55	4.4	3.45	3.458

Darbs tiek izstrādāts bez jebkādas iepriekšējas pieredzes testu automatizācijā, un konkrētās valodas izmantošanā. Šie ir apgrūtinoši faktori, kuri ņemti vērā veicot darbietilpības aprēķinu. Ir iespējams, ka šāda apjoma programmatūras apjoms nozarē pieredzējušam programmētājam būtu novērtējams ar mazāku darba apjomu.

8. SECINĀJUMI

Ir izveidota programmatūra, kas atvieglo testēšanas procesu uzņēmumā. Izstrādājot šo programmatūru, tika iegūts paplašināts priekštats par to, kā notiek programmatūras testēšana un tā izstrāde. Darba izstrāde bija ļoti vērtīga pieredze testu automatizācijas procesā. Tā rezultātā tika apgūti gan testu automatizācijas pamati, gan specifiskas zināšanas aparatūras testēšanā, gan par pašu programmatūras izstrādes dzīves ciklu. Problēmas protams sagādāja, pierast pie programmatūras izstrādes metodoloģijas un apgūst jaunas prasmes un izstrādes rīkus. Kopumā šis darbs palīdzēja autora attīstīt programmētāja "mīkstās prasmes", jeb dokumentācijas rakstīšanu un darbošanas komandā.

9. IZMANTOTĀ LITERATŪRA

1. Telos VX lietošanas manuālis [tiešsaiste][atsauce – 28.05.2018] Pieejams internetā - https://www.telosalliance.com/images/Telos%20Products/VX/Support%20Files/VX%20Manual-2.0.2.pdf
2. Spunde, D.AutomaticTestScriptLanguageManual_1-1-6.odt[atsauce – 28.05.2018]
3. Rukšāns, J.Petrovs, N. - VX LWCP: LiveWire Control Protocol for Phone Systems[atsauce – 28.05.2018]
4. SIPp telefona lietotnes lietošanas manuālis [tiešsaiste][atsauce – 28.05.2018] Pieejams internetā - http://sipp.sourceforge.net/doc/reference.html

10. PIELIKUMI

Programmatūras pirmkoda fragments

```
/*
File: individual_line_test_case_1.lwscen
Summary: Script for placing a outgoing call to VX and then terminating it from VX.
    Authors: Uga Sprogis
      Date: March 13 2018 V.1.0.0
       RUN: ruby .../interpret/main.rb .../individual_line_test_case_1.lwscen -p
VX_ADDRESS: -p USERNAME: -p PASSWORD: -p STUDIO_ID: -p FIXED_LINES: -p
SELECTABLE_LINES: -p SHOW_ID: -p LINE_ID: -p ENABLE_AUTO_FIX: -p
ENABLE_LOCKLESS: -p LOCAL_IP_ADDRESS: -p REPEAT:
                        PARAMETERS:
                                VX_ADDRESS - VX address
                                USERNAME - username, which is needed to log in the VX
engine
                                PASSWORD - password, which is needed to log in the VX
engine
                                STUDIO_ID - VX studio to use, selected by its ID
                                FIXED_LINES - number of how many fixed livewire channels
should be added to the selected
                                                studio
                                SELECTABLE_LINES - number of how many selectable
livewire channels should be added to the
                                                selected studio
                                SHOW_ID - VX studio show to use, selected by its ID
                                LINE_ID - studio show line, to which make the outgoing call
and check callstates
                                ENABLE_AUTO_FIX - enable auto answer fixed lines(Values
can be "checked" or "unchecked")
                                ENABLE_LOCKLESS - enable lockless conferencing(Values
can be  "checked" or "unchecked")
                                LOCAL_IP_ADDRESS - your PC ip address for the SIPp tool
to use, when making the calls
                                                to VX
                                REPEAT - declares how many studio lines to repeat the script

 Description:
        When executing the script, the states should succesfully change in this sequence:
        IDLE --> DIALING --> TERMINATED --> IDLE. Test case script uses only one
        studio and show.
        The line is in avaible callstate (IDLE), then the outbound call is being placed
(DIALING).
         After this, the call is being dropped (TERMINATED) and returning to being available
        callstate (IDLE).
        The number to which the outbound call is being placed, belongs to a open source cross
        platform SIP Phoneapplication which can be runned from the terminal named "SIPp".
```

Revision:

Uga Sprogis April 6 2018 V.1.0.1 - new parameters are added: USERNAME, PASSWORD,
so the login process to the VX engine is more flexible and not hardcoded
and parameter names are changed to get a better unstanding of their meaning.
More comments are added, and some comments are edited, to get a better understanding what the script does.

Uga Sprogis April 12 2018 V.1.0.2 - with some modifications to the interpretor,
changes have been made to ${PASSWORD} parameter usage: now a empty parameter can be passed, if needed.

Uga Sprogis April 17 2018 V.1.0.3 - VX studio configuration and VX studio reset scripts are added, so that the VX studio configuration and default value resetting is not done manually by hand.
Shell scripts are no longer needed due to the improvement of the interpretor to run SIPp as a external program.

Uga Sprogis May 7 2018 V.1.0.4 - scripts are overwritten so they can run only one line at the time This is due to that the scripts are defined for doing one thing which is specified by their file name.

*/

#Variables:

#Declares for how many studio lines to go though with this callstate script
%line=${LINE_ID}
#------------------------------VX studio configuration--

#Before checking studio show lines for needed callstates, a studio is needed to be configured
<< Configuring VX studio ${STUDIO_ID} and adding the show ${SHOW_ID} to it

run ../configuration_setup/VX_studio_configuration.lwscen -p VX_ADDRESS:${VX_ADDRESS} -p STUDIO_ID:${STUDIO_ID} -p USERNAME:${USERNAME} -p PASSWORD:${PASSWORD} -p FIXED_LINES:${FIXED_LINES} -p SELECTABLE_LINES:${SELECTABLE_LINES} -p ENABLE_AUTO_FIX:${ENABLE_AUTO_FIX} -p ENABLE_LOCKLESS:${ENABLE_LOCKLESS} -p SHOW_ID:${SHOW_ID}
#---

<<
<< Testing callstate route: IDLE -> DIALING -> TERMINATED -> IDLE
<<

loop ${REPEAT}

#---
#Running the SIPp aplication for taking calls from VX
<<

```
                    << Setting SIPp in call answering mode
                    run_external sipp -sf
${SCENARIO_PATH}/../SIPp/take_call_from_VX.xml -i ${LOCAL_IP_ADDRESS} -d 0 -
m ${REPEAT}
                    #-----------------------------------------------------------------

                    << ------------------------TEST IDLE callstate-------------------------------
-
                    run ../callstates/IDLE.lwscen -p VX_ADDRESS:${VX_ADDRESS} -p
USERNAME:${USERNAME} -p PASSWORD:${PASSWORD} -p
STUDIO_ID:${STUDIO_ID} -p SHOW_ID:${SHOW_ID} -p LINE_ID:%{line} -p
REPEAT:1

                    << ------------------------TEST DIALING callstate--------------------------
----
                    run ../callstates/DIALING.lwscen -p
VX_ADDRESS:${VX_ADDRESS} -p USERNAME:${USERNAME} -p
PASSWORD:${PASSWORD} -p STUDIO_ID:${STUDIO_ID} -p
SHOW_ID:${SHOW_ID} -p LINE_ID:%{line} -p REPEAT:1 -p
LOCAL_IP_ADDRESS:${LOCAL_IP_ADDRESS}

                    << ------------------------TEST TERMINATED callstate----------------------
------
                    run ../callstates/TERMINATED.lwscen -p
VX_ADDRESS:${VX_ADDRESS} -p USERNAME:${USERNAME} -p
PASSWORD:${PASSWORD} -p STUDIO_ID:${STUDIO_ID} -p
SHOW_ID:${SHOW_ID} -p LINE_ID:%{line} -p REPEAT:1

                    << ------------------------TEST IDLE callstate-------------------------------
-
                    run ../callstates/IDLE.lwscen -p VX_ADDRESS:${VX_ADDRESS} -p
USERNAME:${USERNAME} -p PASSWORD:${PASSWORD} -p
STUDIO_ID:${STUDIO_ID} -p SHOW_ID:${SHOW_ID} -p LINE_ID:%{line} -p
REPEAT:1

                    #The repetation is done in a ascending order
                    %line+1

                    #The script is put on sleep, so SIPp can close itself succesfully
                    <<

                    << Closing SIPp

                    sleep 4

                    <<

          endloop

     #-----------------Resetting VX studio to default values--------------------------------
```

```
        #After the callstate checking is done, the specified studio is reset to default values
        run ../configuration_setup/VX_studio_reset.lwscen -p
VX_ADDRESS:${VX_ADDRESS} -p STUDIO_ID:${STUDIO_ID} -p
USERNAME:${USERNAME} -p PASSWORD:${PASSWORD}
```

Kvalifikācijas darbs „TELOS VX funkcionālo testu automatizācija” izstrādāts Latvijas Universitātes Datorikas fakultātē.

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

Autors: *Uga, Sproģis* _____ ___.05.2018.

Rekomendēju darbu aizstāvēšanai
Darba vadītāj**s**: M. Dat. Matīss Rikters_____ ___**.**05.2018.

Recenzents: M.dat. Jānis Mārtužs

Darbs iesniegts 28.05.2018.
Kvalifikācijas darbu pārbaudījumu komisijas sekretāre: *Darja Solodovņikova*_____

Darbs aizstāvēts kvalifikācijas darbu pārbaudījuma komisijas sēdē
___.06.2018. prot. Nr. _____

Komisijas sekretārs(-e): _____