LATVIJAS UNIVERSITĀTE
DATORIKAS FAKULTĀTE

STARPPLATFORMU MOBILO LIETOTŅU IZSTRĀDE BEZ TĪMEKĻA TEHNOLOĢIJĀM

BAKALAURA DARBS

Autors: **Kristaps Krūmiņš**
Studenta apliecības Nr.: kk13074
Darba vadītājs: asociētais profesors, Dr. dat. Uldis Straujums

RĪGA 2017

ANOTĀCIJA

Bakalaura darbā “Starpplatformu mobilo lietotņu izstrāde bez tīmekļa tehnoloģijām” tiek aprakstītas metrikas un kritēriji, pēc kuriem var veikt ietvaru salīdzināšanu. Vēl darbā tiek veikta 3 ietvaru salīdzināšana, pēc aprakstītajām metrikām un kritērijiem. Darba mērķis ir sniegt pamatotu atbildi uz jautājumu: “Kuru no starpplatformu ietvariem izvēlēties?”.

Darbā aprakstīti mobilo lietotņu izstrādes veidi, metrikas un kritēriji. Tiek sniegts ieskats autora 3 izvēlētajos ietvaros, kuri tiek salīdzināti. Nobeigumā tiek sniegta atbilde uz iepriekš izvirzīto jautājumu.

Atslēgvārdi: starpplatformu ietvari, lietotnes, metrikas, kritēriji.

ABSTRACT

CROSS-PLATFORM MOBILE APPLICATION DEVELOPMENT WITHOUT WEB TECHNOLOGIES

The bachelor work “Cross-platform mobile application development without web technologies” describes metrics and criteria which can be used to compare frameworks. Also in the work author compares 3 frameworks by described metrics and criteria. The aim of work is to provide reasoned answer to question: “Which one of the cross-platform frameworks to choose?”.

Work describes different kinds of mobile application development methods, metrics and criteria. Provides an insight of authors selected 3 frameworks which will be compared. In final author provides an answer for previous given question.

Keywords: cross-platform frameworks, applications, metrics, criteria.

SATURA RĀDĪTĀJS

APZĪMĒJUMU SARAKSTS

API (Application Programming Interface) - iepriekš definētu funkciju kopums, kas tiek pasniegts kā pielikums, kuru iespējams izmantot ārējiem programmatūras produktiem.

CPU (central processing unit) - galvenā vadības ierīce, kas savienota ar speciālajiem procesoriem, galveno atmiņu un kas kontrolē informācijas ievadi un izvadi [1].

CSS (Cascading Style Sheets) – stila lapas valoda, ko lieto, lai aprakstītu izskatu HTML dokumentiem.

HTML (HyperText Markup Language) – valoda, kas, izmantojot speciālus kodus, nosaka hiperteksta dokumenta atveidojumu displeja ekrānā gadījumā, ja tiek lietotas interneta globālā tīmekļa lappuses [1].

HTTP (HyperText Transport Protocol) - Interneta standartprotokols, kas nodrošina informācijas apmaiņu globālajā tīmeklī [1].

HTTPS (Hypertext Transfer Protocol Secure) – HTTP protokola paplašinājums drošai saziņai datortīklā.

JS (JavaScript) – firmas Netscape izveidota valoda, kas ļauj globālā tīmekļa izstrādātājiem veidot interaktīvas vietnes [1].

NFC (near field communication) – komunikācijas tehnoloģija, kas bāzētu uz īsa attāluma (4cm vai mazāk) bezvadu datu apmaiņu.

QML (Qt Meta Language) – lietotāja saskarnes iezīmēšanas valoda.

SDK (software development kit) – palīgprogrammu kopa, kas lietojumprogrammu izstrādātājam atvieglo programmu veidošanu, ievērojot konkrētās to darbības vides īpatnības [1].

Starpplatformu mobilā lietotne – mobilā lietotne, kuras pirmkods ir pilnībā vai gandrīz pilnībā neatkarīgs no mobilās operētājsistēmas.

UI (user interface) – lietotājam redzamā sistēmas daļa.

XAML (Extensible Application Markup Language) – ir uz paplašinātās iezīmēšanas valodas (XML) bāzēta valoda, ko ir izstrādājusi kompānija Microsoft.

IEVADS

Mobilo operētājsistēmu vidū ir novērojama fragmentācija, katra kompānija ir izveidojusi savu metodi lietotņu izstrādei, izmantojot dažādas programmēšanas valodas un SDK [2]. Kādas ir iespējas mobilo lietotņu izstrādē? Šobrīd industrijā eksistē divas mobilo lietotņu izstrādes pieejas – natīvā un hibrīda. Natīvā pieeja ir standarta pieeja, jo to rekomendē operētājsistēmas izstrādātājs, tā izmanto mērķa mobilās operētājsistēmas piedāvātos rīkus un tehnoloģijas. Hibrīda pieeja izmanto ietvaru, līdz ar to tiek izmantoti ietvara rīki un tehnoloģijas. Ietvara viens no mērķiem ir, lai lietotne strādātu uz vairāk kā vienas mobilās operētājsistēmas. Starpplatformu ietvarus var iedalīt divās grupās, tie, kas izmanto tīmekļa tehnoloģijas un tie, kas neizmanto. Ietvari, kas izmanto tīmekļa tehnoloģijas jeb HTML, JS un CSS, ir populārāki par ietvariem, kas neizmanto tīmekļa tehnoloģijas. Tas ir tāpēc, ka programmēšana ar HTML, JS un CSS ir salīdzinoši ātra, un parasti jau ir gatavs tīmekļa vietnes risinājums, kuru ar maziem resursiem var pārveidot un izmantot lietotnē. Bet mobilajām lietotnēm, kas izstrādātas ar šāda veida ietvariem, ir arī savi trūkumi, viens no galvenajiem ir ātrdarbība. Savukārt lietotnes, kas izstrādātas ar ietvaru, kas realizē starpplatformu atbalstu neizmantojot tīmekļa dzini, piedāvā lielāku ātrdarbību. Tas arī ir iemesls, kāpēc autors izvēlējās apskatīt ietvarus, kas neizmanto tīmekļa tehnoloģijas.

Tēma ir aktuāla, jo mobilo operētājsistēmu tirgū nav viena izteikta līdera, līdz ar to izstrādātāji ir ieinteresēti, kā ar mazākām pūlēm var atbalstīt vairākas operētājsistēmas. Starpplatformu ietvari samazina izstrādes laiku un resursu daudzumu, kas nepieciešams, lai izstrādātu lietotni uz vairākām platformām.

Darba mērķis ir sniegt pamatotu atbildi uz jautājumu: “Kuru no starpplatformu ietvariem izvēlēties?” Lai spētu atbildēt uz šo jautājumu, darba gaitā tika izvirzīti sekojošie uzdevumi:

1. izpētīt programmatūras mērīšanas metodes;
2. izpētīt līdzīgos darbus šajā tēmā;
3. izvirzīt un aprakstīt metrikas un kritērijus, pēc kuriem veikt ietvaru salīdzināšanu;
4. noskaidrot 3 populārākos starpplatformu ietvarus, kas atbilst izvirzītajām prasībām;
5. veikt izvēlēto ietvaru salīdzināšanu.

Darbā tiek izmantotas sekojošās pētniecības metodes:

- salīdzināšanas metode;
- analītiskā metode;
- eksperimenta metode;

Darbs pamatojas uz ārvalstīs izdotām grāmatām, zinātniskām publikācijām, standartiem un Internetā pieejamiem materiāliem.

Darbs sastāv no ievada, pamatdaļas ar 4 nodaļām un 10 apakšnodaļām, kas sevī ietver vēl 25 apakšnodaļas, kuras iekļauj vēl 2 apakšnodaļas.

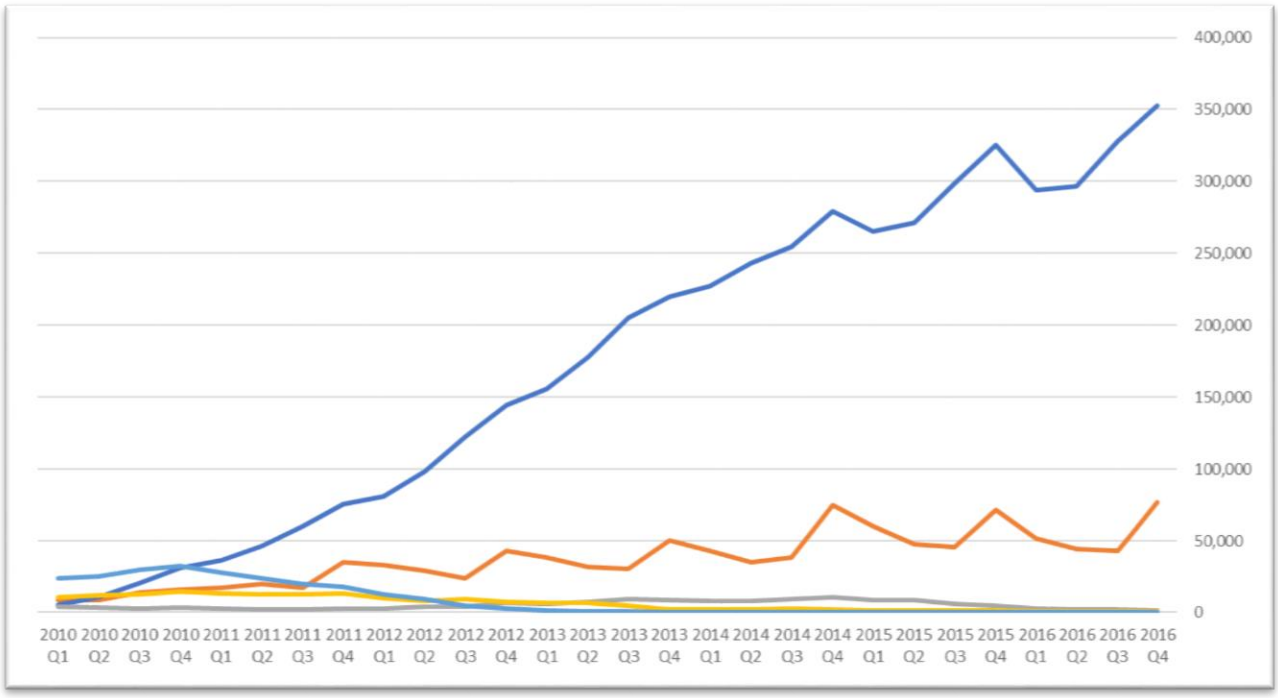
Darbā iekļauti 27 informācijas avoti, 19 tabulas, 9 attēli un 4 pielikumi.

1. MOBILO LIETOTŅU IZSTRĀDES VEIDI

Nodaļā tiek aprakstīts pašreizējais stāvoklis mobilo operētājsistēmu tirgū un sniegts ieskats mobilo lietotņu izstrādes veidos.

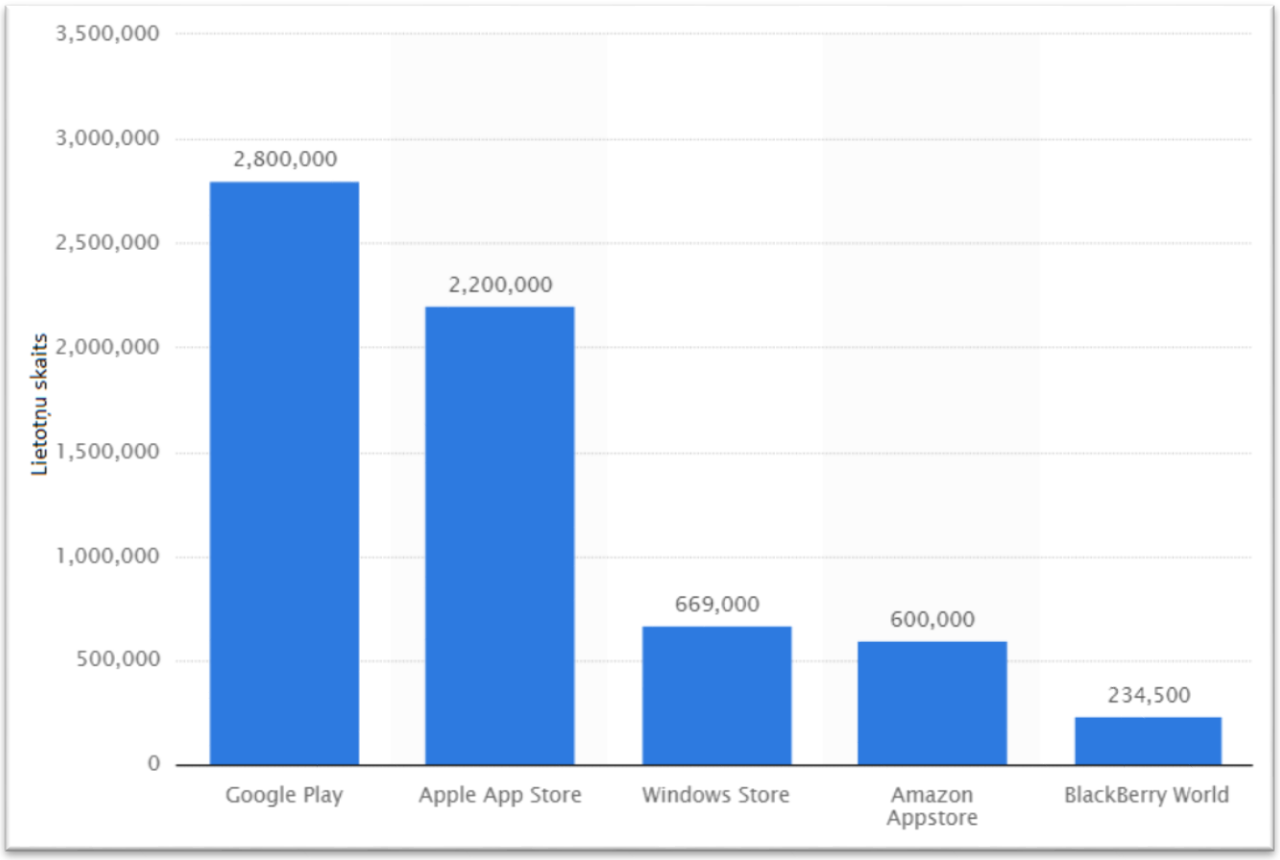
1.1. Mobilās operētājsistēmas

Ar katru gadu pārdoto viedtālruņu skaits palielinās (1.1. att.), līdz ar to pieprasījums pēc mobilajām lietotnēm arī palielinās. Domājot par mobilās lietotnes izstrādi, svarīgs faktors ir mobilā operētājsistēma, uz kuras darbosies lietotne. Eksistē vairākas mobilās operētājsistēmas, no kurām divas populārākās ir Android un iOS (1.1. att.) [3].



1.1. att. **Pārdoto viedtālruņu skaits tūkstošos pēc mobilajām operētājsistēmām**

Lietotņu izplatīšanas vidēs ir redzama līdzīga statistika, Google Play (izplata Android lietotnes) ir pirmajā vietā, kam seko Apple App Store (izplata iOS lietotnes) (1.2. att.) [4].



1.2. att. **Lietotņu skaits populārākajos lietotņu izplatīšanas vidēs**

Tā kā priekš Android izstrādāta mobilā lietotne nestrādās uz iOS un otrādi, tad rodas jautājums, ko darīt, lai lietotne strādātu uz vairāk, kā vienas mobilās operētājsistēmas. Viens variants ir, veidot vairākas lietotnes, katru priekš savas operētājsistēmas, un otrs variants ir, izmantot kādu no starpplatformu risinājumiem.

1.2. Starpplatformu ietvaru dažādība

Darbā apskatīta starpplatformu pieeja, kuras viens no mērķiem ir samazināt pirmkoda daudzumu, kas nepieciešams, lai izstrādātu lietotni, priekš vairākām mobilajām operētājsistēmām. Lai realizētu šo pieeju, tiek izmantots kāds no starpplatformu ietvariem. Salīdzinājumā ar pieeju, kad katrai operētājsistēmai tiek veidota sava lietotne, jeb natīvo pieeju, starpplatformu pieejai ir savas priekšrocības un trūkumi. Tā kā vairāk par 70% no programmatūras izstrādes aizņem testēšana un uzturēšana, tad ir vērts padomāt par starpplatformu pieeju vairāku lietotņu gadījumā [5]. Šīs pieejas priekšrocība ir vairāk kā vienas operētājsistēmas atbalsts. Atkarībā no izvēlētā starpplatformu ietvara, ir iespējams no viena pirmkoda izveidot izpildāmo datni vairākām

operētājsistēmām. Otra priekšrocība ir programmēšanas valodas izvēle. Eksistē vairāki ietvari ar dažādām programmēšanas valodām, līdz ar to ir iespējams izvēlēties izdevīgāko. Trūkums ir ātrdarbības zudums, kas atkarībā no ietvara var būt niecīgs vai pat jūtami liels.

Starpplatformu ietvarus var iedalīt divās grupās, tie, kas izmanto tīmekļa tehnoloģijas, un kas neizmanto. Ietvari, kas izmanto tīmekļa tehnoloģijas jeb HTML, JS un CSS, pārsvarā izmanto tīmekļa pārlūkprogrammas dzini kā, piemēram, Adobe PhoneGap, Sencha, Ionic, Onsen UI. Šajā gadījumā ātrdarbība ir jūtami lēnāka par natīvo lietotni. Eksistē arī ietvari, kas izmanto HTML, JS un CSS, bet neizmanto tīmekļa dzini kā, piemēram, Axway Appcelerator, React Native, Flutter, RubyMotion. Šie ietvari sola ātrdarbību līdzīgu kā natīvai lietotnei, bet tie nav vienīgie. Visi ietvari, kas neizmanto tīmekļa tehnoloģijas, arī piedāvā ātrdarbību, kas līdzinās natīvai lietotnei. Darbā tiks salīdzināti populārākie šāda veida ietvari, kas nav domāti spēļu izstrādei.

2. METRIKAS UN KRITĒRIJI

Viens no veidiem, kā veikt ietvaru salīdzināšanu, ir no sākuma tos nomērīt un pēc tam salīdzināt rezultātus pēc kritērijiem. Mērījums ir mēra noteikšanas procedūra. IEEE standartu vārdnīca priekš programmatūras inženierijas terminiem definē metriku kā kvantitatīvu mēru, kas parāda, kādā pakāpē sistēmai, komponentei vai procesam ir spēkā dotais atribūts [5]. Bet programmatūrai ne visu mēs spējam nomērīt, līdz ar to ir jāmeklē citi veidi, kā iegūt datus uz kuriem balstīties. Viens no piemēriem, ko nevar nomērīt, ir lietotāja saskarne, tāpēc tiek meklēti citi veidi, kā iegūt datus, pēc kuriem vadīties. Lietotāja saskarni vislabāk ir mērīt pēc aptaujām, kas tad arī ir vēl viens veids, kā veikt salīdzināšanu.

2.1. Problēmas programmatūras mērīšanā

Viena no programmatūras mērīšanas problēmām ir, ka programmatūras inženierija ir ļoti sarežģīts process, kas rada sarežģītu produktu [6]. Mērīšanu arī apgrūtina tas, ka katrs projekts ir unikāls. Vēl viena problēma ir, ka ar skaitliskām vērtībām nevar raksturot kā nomērīt, piemēram, programmatūras estētiskumu. Tomēr eksistē vairākas metrikas pēc kurām var mēģināt mērīt programmatūras, viena no tām ir pirmkoda rindiņu skaits. Tā ir populāra metrika, jo pirmkoda rindiņu skaitu ir viegli iegūt, bet mazāks rindiņu skaits ne vienmēr ir labāks, jo tad pasliktinās koda lasāmība.

2.2. Metrikas

Šajā apakšnodaļā autors apraksta 4 metrikas – CPU lietojums, kopīgais kods, ātrdarbība un lietotnes datnes izmērs. Tās tiks izmantotas mobilo lietotņu salīdzināšanai, kas izstrādātas ar katru no ievēlētajiem ietvariem. Ir vērts pieminēt, ka metrikas autors nav smēlies no kāda avota.

2.2.1. Uz pirmkodu balstītas metrikas

Pirmkoda rindiņu skaita metrika tiek izmantota, lai mērītu programmas izmēru, saskaitot visas pirmkoda rindiņas. To izmanto arī, lai noteiktu aptuveno darbietilpību un uzturēšanas izmaksas. Izmantojot šo metriku salīdzināšanā, tā būs noderīga vienīgi, ja rindiņu skaits, kas atšķiras, ir ļoti liels, piemēram, programma ar 3000 rindiņām un 20000 rindiņām. Šajā gadījumā šī metrika nav noderīga, jo mums neinteresē ietvara pirmkoda rindiņu skaits, un izstrādātās lietotnes pirmkoda rindiņu skaits ir atkarīgs no izstrādes programmēšanas valodas, kas katram

ietvaram var būt cita. Salīdzinot programmatūras, kurās izstrāde notiek vienā un tajā pašā programmēšanas valodā, tādas metrikas kā ciklomātsikā sarežģītība, uzturēšanas indekss un pirmkoda rindiņu skaits, būs noderīgas. Visual Studio piedāvā izstrādātajam projektam veikt šādus mērījumus [7].

2.2.2. CPU lietojums

Viens no viedtālruņa galvenajiem baterijas patērētājiem ir CPU jo vairāk tas tiek nodarbināts, jo ātrāk izlādējas baterija. Svarīgi, lai ietvars baterijas jaudu izmanto efektīvi. Jaudas vidējais patēriņš un vidējā CPU noslodze tiek mērīta ar Trepn Profiler, tā ir Android lietotne, kas pieejama Google Play veikalā [8].

2.2.3. Kopīgais pirmkods

Tā kā starpplatformu ietvaru galvenais mērķis ir samazināt pirmkoda daudzumu, kas nepieciešams, lai kompilētu to uz vairākām platformām, tad svarīgi ir noskaidrot, cik daudz no pirmkoda ir kopīgs vairākām platformām. Kopīgo pirmkodu procentos var iegūt dalot kopīgo pirmkoda rindiņu skaitu ar kopējo rindiņu skaitu. Ja nebūs platformu specifisks pirmkods, tad, respektīvi, kopīgais kods būs 100%. Pirmkoda rindiņu skaits tiek iegūts izmantojot cloc programmu [9].

2.2.4. Ātrdarbība

Lietotāji sagaida, ka lietotne strādās reālā laikā, tas ir, tā momentāni reaģēs uz lietotāja ievadi. Autors ir izvēlējies šādas ātrdarbības mērīšanas metodes:

- ielādes laika mērīšana;
- aktivitāšu pārslēgšanās laika mērīšana.

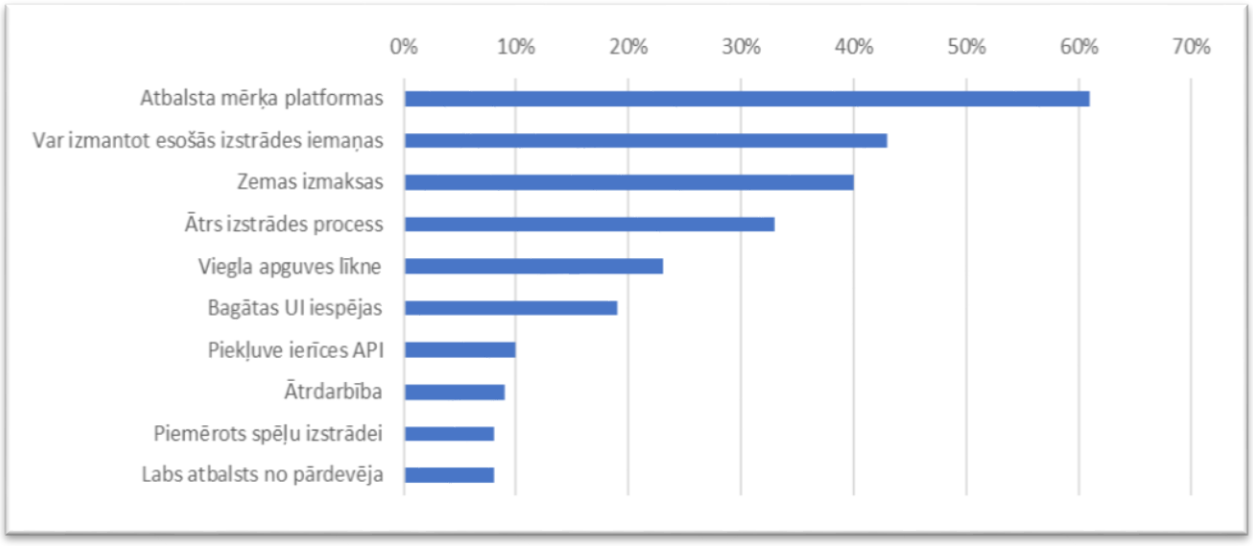
2.2.5. Lietotnes datnes izmērs

Lielāks lietotnes datnes izmērs nozīmē, ka lietotnes lejupielāde notiks ilgāk un tā aizņems vairāk atmiņu uz viedtālruņa. Labāks ietvars ir tas, kurš rada mazāku virstēriņa (overhead) izmēru, kas rodas konvertācijas laikā.

2.3. Kritēriji

Katrai mobilajai lietotnei ir savs mērķis, līdz ar to, lai izvelētos piemērotāko ietvaru, ir jāskatās uz īpašībām, kas palīdzēs šos mērķus sasniegt. Pēc Appcelerator pētījuma datiem primārie 3 lietotnes mērķi ir peļņas gūšana (60,4%), klientu lojalitātes palielināšana (45,9%) un zīmola atpazīstamības palielināšana (36,9%) [10]. Peļņas gūšanai būs svarīga darbinieku apmācības ilgums, ietvara izmaksas un produktivitāte. Savukārt uz klientu lojalitāti balstītām lietotnēm, būs svarīga to kvalitāte un darbība ilgtermiņā. Bet zīmola atpazīstamības palielināšana, vislabāk izdosies, ja lietotne izcelsies no pārējām, līdz ar to iespēja veidot unikālu dizainu ir svarīga.

Lietotni izstrādās programmētāji, tāpēc ir būtiski ņemt vērā to, ko viņi sagaida no laba ietvara. Vision mobile 2012. gadā veica aptauju par izstrādātāju iemesliem rīka izvēlē, kurā piedalījās vairāk kā 50 respondenti (2.1. att.) [11].



2.1. att. **Izstrādātāju iemesli ietvara izvēlē**

Galvenais ietvara izvēles iemesls ir atbalsts pēc visām mērķa platformām (61%), kas bija raksturīgi tajā laikā, jo nebija divu izteiktu līderu. Šobrīd ietvars, kas neatbalsta Android un iOS, nav konkurētspējīgs. Vēl pēc aptaujas datiem var redzēt, ka izstrādātāji dod priekšroku tiem ietvariem, kas ļauj izmantot esošās izstrādes iemaņas (43%). 4. vietā ir ātrs izstrādes process (33%) un 5. vietā ir viegla apguves līkne (23%), kas tikai pastiprina faktu, ka izstrādātāji vēlas pēc iespējas mazāk veltīt laiku jaunu lietu apguvei un fokusēties uz ātru rezultātu.

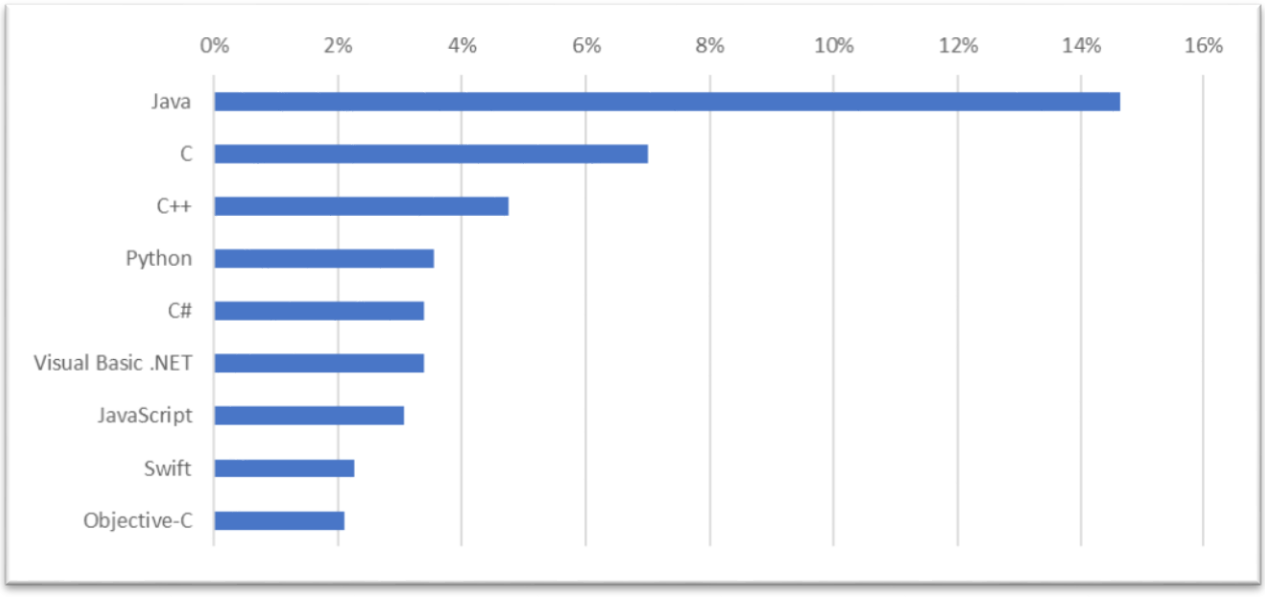
Šajā apakšnodaļā autors ir uzskaitījis vairākus kritērijus, kuru izvēle balstās uz H. Heitkötter, S. Hanschke un T. A. Majchrzak darba par starpplatformu izstrādes izvērtējumu [12]. Vairākums gadījumu mums ir svarīgi, lai papildus ietvara SDK, būtu tādas lietas kā integrētā izstrādes vide, vizuālais saskarnes izveides rīks un testēšanas rīki, tāpēc arī tika izvēlēti šādi kritēriji.

2.3.1. Apguves līkne

Jo ātrāk ir iespējams apgūt ietvaru, jo ātrāk ir iespējams sākt iegūt rezultātus. Apguves līkne ir patērētā laika attiecība pret apgūto funkcionalitātes daudzumu. Pārsvarā tiek salīdzināts nepieciešamais laiks pamat funkcionalitātes apgūšanai. Šī kritērija izvērtēšana, darbā notiek pēc programmēšanas valodas popularitātes un dokumentācijas dažādības.

2.3.1.1. Programmēšanas valoda

Kā jau tika minēts iepriekš, programmētāji dod priekšroku esošo iemaņu izmantošanai, nevis jaunu iemaņu apgūšanai, viens no šiem faktoriem ir programmēšanas valoda. Ja kompānijai jau ir programmētāji, tad vislabāk būtu izvēlēties ietvaru, kas izmanto to programmēšanas valodu, ar ko programmētājiem ir pieredze. Ja priekš mobilās lietotnes izstrādes ir plānots meklēt jaunus programmētājus, tad noderīgi ir ņemt vērā programmēšanas valodu popularitāti, jo tas ietekmēs to, cik viegli būs atrast vajadzīgo programmētāju. TIOBE kompānijas dati par populārākajām programmēšanas valodām redzami 2.2. att. [13].



2.2. att. **Populārākās programmēšanas valodas**

Kā redzams, izteikts līderis ir Java (14,6%), kam seko C (7%) un C++ (4,7%) programmēšanas valodas. Skatoties uz natīvo izstrādi, tad Android programmētājus ir salīdzinoši viegli atrast, jo Android programmēšanas valoda ir Java, savukārt iOS programmētājus būs daudz grūtāk, jo Swift un Objective-C programmētāju ir mazāk kā 3%.

2.3.1.2. Dokumentācija un atbalsts

Ietvars bez dokumentācijas ir praktiski nelietojams, jo izstrādātājam nav skaidrs, kā var paveikt vajadzīgos uzdevumus. Dokumentācijai ir jāapraksta izstrādes vides uzstādīšana un jāsniedz piemēri pamata funkcionalitātes realizēšanai. Ne vienmēr viss būs aprakstīts dokumentācijā, tāpēc ir labi, ja var saņemt atbildes no produkta izstrādātāja. Ja ietvaru uztur kopiena, tad svarīgi ir, lai tā ir aktīva, jo tad ir lielāka iespējamība saņemt atbildes uz konkrētiem jautājumiem.

Novērtēšana tiks veikta pēc tā, cik dažādi veidi būs pieejami dokumentācijai, piemēram, dokumentētas parauga lietotnes, rakstveida pamācības, video pamācības un diskusijas.

2.3.2. Lietotāja saskarnes izveide

Izstrādājot mobilo lietotni, samērā daudz laika var tikt veltīts lietotāja saskarnes izveidei, tāpēc grafisks rīks, kas ļauj ar peles palīdzību paņemt vajadzīgo elementu (pogu, teksta lauku, datuma lauku) un pievienot to vajadzīgajā vietā, ir krietns atvieglojums. Ir labi, ja rīks piedāvā saskarni apskatīt uz vairākiem ierīču ekrāna izmēriem. Šādus redaktorus sauc par WYSIWYG.

2.3.3. Izmaksas un licence

Mobilās lietotnes izstrādes budžets nosaka to, cik mēs varam tērēt izmaksās par ietvaru, ar kuru tiks izstrādātas lietotnes. Pēc research2guidance datiem vidējais mobilo lietotņu budžets ir $37000 un tieši starpplatformu lietotņu budžets ir $25000 [14]. Ja tiek izstrādāta tikai viena lietotne, tad nebūs izdevīgi izvēlēties ietvaru, kuram ir dārga licence. Jo vairāk lietotnes tiek plānots izstrādāt, jo vairāk līdzekļu var atvēlēt ietvara iegādei. Acīmredzami ir tas, ka tiks meklēti veidi kā pēc iespējas vairāk samazināt izmaksas, bet vairākums gadījumu labāk ir samaksāt par dārgāku produktu, lai iegūtu ilgtermiņa atbalstu.

Dažkārt ir svarīgi, lai projekta izstrādē visi izmantotie rīki ir atvērtā pirmkoda, jo tas, ka šobrīd rīks ir bezmaksas, nenozīmē, ka tas būs bezmaksas arī turpmāk. Šajā gadījumā derēs tikai tie ietvari, kuru pirmkods ir publiski pieejams.

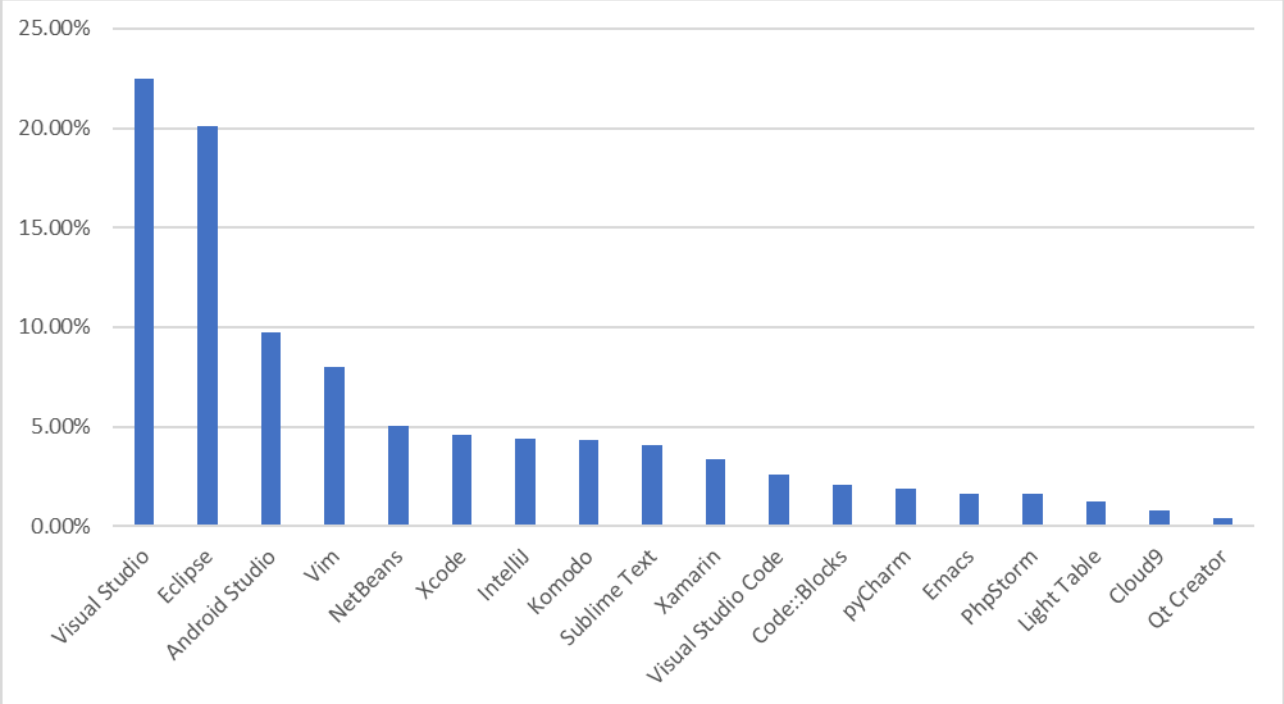
2.3.4. Piekļuve ierīcei

Piekļuve ierīcei jeb piekļuve natīvām API nozīmē, piemēram, piekļuvi kamerai, ģeolokācijai (geolocation), akselerometram, žiroskopam un kompasam. Lielāks atbalstīto API skaits nāk tikai par labu.

2.3.5. Integrētā izstrādes vide

Programmētāja vissvarīgākais rīks izstrādes procesā ir integrētā izstrādes vide, jo pie tās tiks pavadīts lielākais laiks. Eksistē vairākas izstrādes vides un katra no tām piedāvā daudz dažādu funkciju, kas paātrina un uzlabo izstrādes procesu. Galvenās funkcijas, kas būtu jāatbalsta modernam izstrādes rīkam, ir automātiska koda fragmentu ģenerācija, automātiskā pabeigšana (auto-complete) un atkļūdotājs.

Būtisks rādītājs ir integrētās izstrādes vides popularitāte, jo pirmkārt, tas liecina par kvalitāti un otrkārt, tas parāda, cik liela ir iespējamība, ka programmētājs zinās šo izstrādes vidi. Populārākās integrētās izstrādes vides pēc tā, cik bieži tas tiek meklētas Google meklētājā, ir redzamas 2.3. att. [15].



2.3. att. **Populārākās integrētās izstrādes vides**

Kā redzams, tad divi izteikti līderi ir Visual Studio un Eclipse.

2.3.6. Izstrādes ātrums

Izstrādes ātrums sevī iekļauj izstrādi, testēšanu un produkta palaišanu produkcijā. Tas ir atkarīgs no vairākiem faktoriem kā, piemēram, programmēšanas valodas, ietvara iespējām, apguves līknes un dokumentācijas daudzuma.

2.3.7. Natīvs UI

Viens no starpplatformu lielajiem izaicinājumiem ir panākt to, ka lietotne izskatās kā natīva uz katras no atbalstītajām platformām. Tas nav viegli panākams, jo iOS elementi nav publiski pieejami. Vislabākais rezultāts ir, ja starpplatformu lietotni nevar atšķirt no natīvās, kā arī jo vairāk natīvi UI elementi tiek atbalstīti, jo labāk.

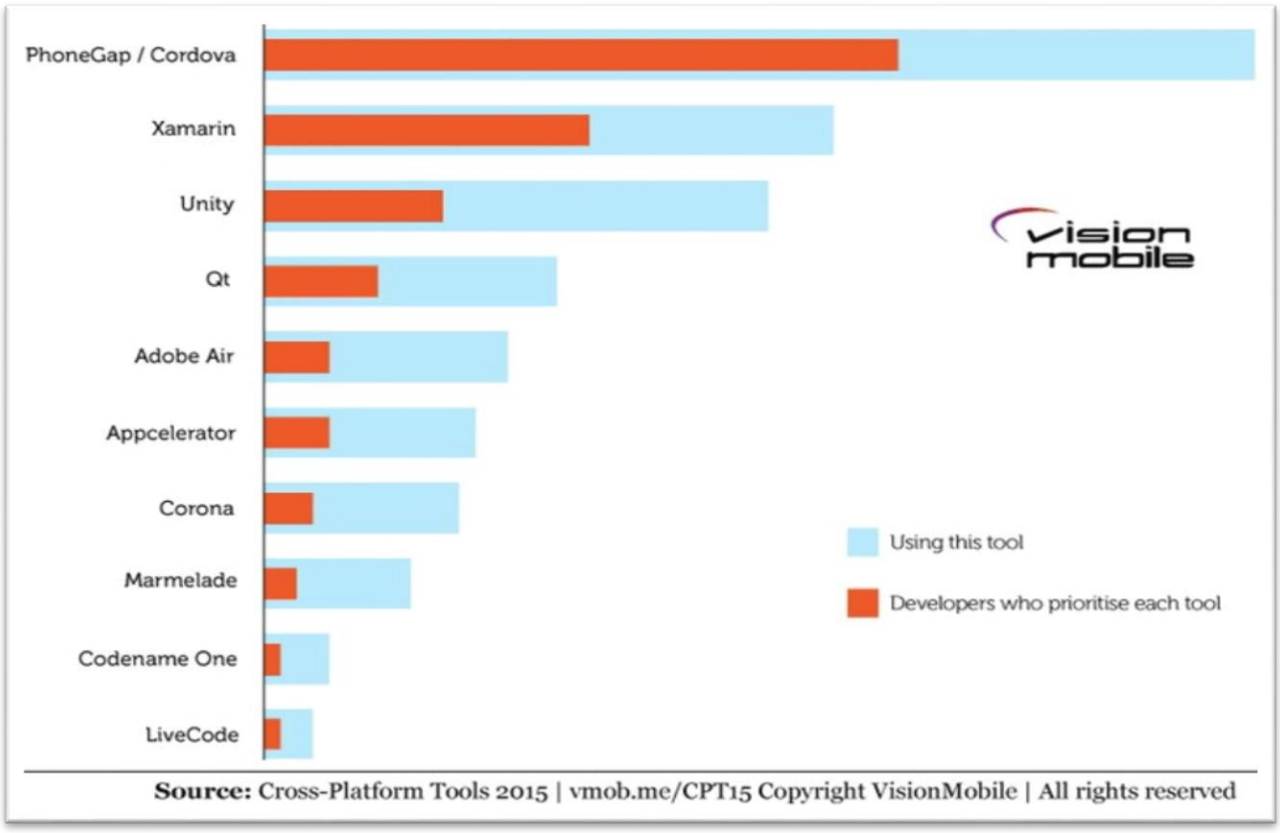
3. IZVĒLĒTIE IETVARI

Pirms autors izvēlējās ietvarus tika izvirzītas sekojošās 3 prasības:

1. ietvaram ir jāatbalsta vismaz Android un iOS operētājsistēmas;
2. mobilās lietotnes izstrādē netiek izmantotas tīmekļa tehnoloģijas;
3. ietvars nav domāts spēļu veidošanai.

Pirmā prasība izriet no tā, ka pasaulē vairāk kā 95% viedtālruņu ir Android vai iOS operētājsistēma. Otrā prasība seko no autora personīgās pieredzes, ka ar tīmekļa tehnoloģijām izstrādāta lietotne ir jūtami lēnāka un vēlmes izmēģināt izveidot starpplatformu lietotni, kas strādā līdzvērtīgi natīvai bez kompromisiem. Iepazīstoties ar vairākiem starpplatformu risinājumiem, autors uzzināja, ka eksistē ietvari, kas JS kodu pārveido par natīvo kodu, tas nozīmē, ka lietotne strādās bez tīmekļa dziņa un ātrdarbība ir ļoti tuvu natīvai lietotnei. Līdz ar to šajā gadījumā varēja izvēlēties ietvarus, kas neizmanto tikai tīmekļa dzini, piemēram, WebKit. Divi no šāda veida ietvariem ir Flutter un React Native. Otrs iemesls šai prasībai ir, ka eksistē ļoti daudzi starpplatformu ietvari un tos visus apskatīt prasītu daudz laika. Trešā prasība seko no tā, ka ietvars, kas domāts spēļu izstrādei, būtiski atšķirsies no parastajiem.

No populārākajiem ietvariem tika izvēlēti Xamarin, Qt, Codename One (3.1. att.).



3.1. att. **Populārākie starpplatformu ietvari**

PhoneGap un Appcelerator ietvari neatbilst prasībām, jo izmanto JS programmēšanas valodu, savukārt Unity, Corona un Marmelade ir spēļu ietvari. Adobe Air jau vairāk kā 4 gadus nav izlaidusi atjauninājumus.

3.1. Xamarin ietvars

Izveidots 2011. gadā un 2016. gadā pārgājis Microsoft īpašumā. Xamarin SDK pirmkods ir pieejams zem MIT licences. Pēc Xamarin datiem vairāk nekā 1,4 miljoni izstrādātāji ir izmantojuši Xamarin produktus [16]. Ietvars atbalsta 3 mobilās operētājsistēmas: Android, iOS un Windows Phone. Izstrādes programmēšanas valoda ir C#, ar to var veidot arī lietotāja saskarni, bet tas nav vienīgais veids. Tiek piedāvātas divas pieejas lietotnes saskarnes izveidei, viena ir izmantojot viņu formas un otra izmantojot mērķa platformas API. Neizmantojot viņu formas ir praktiski neiespējami sasniegt līdz 90% kopīga koda, kas nepieciešams, lai izstrādātu mobilās lietotnes uz vairākām platformām. Iemesls tam ir, ka lietotāja saskarne katrai operētājsistēmai ir jāveido sava. Šo pieeju jāizvēlas, ja īpašs lietotnes dizains ir svarīgāks par kopīgo pirmkodu. Izmantojot viņu

piedāvātās formas ir iespējams izveidot vienkāršu dizainu, kas strādās uz visām trim platformām izmantojot natīvās komponentes. Šeit ir ierobežojums uz to, cik daudz elementu ir pieejams. Papildus ietvaram tiek piedāvāti dažāda veida produkti, kas palīdz lietotņu izstrādē, bet ir vērts pieminēt, ka tie visi ir maksas produkti.

Xamarin University piedāvā tiešsaistes kursus un pamācības priekš mobilo lietotņu izstrādes ar Xamarin. Ir iespēja iegūt Xamarin sertifikātu C# mobilo lietotņu izstrādē. Šis produkts maksā $83,25 mēnesī, ir pieejama izmēģinājuma versija uz 30 dienām.

Piedāvātā izstrādes vide ir Visual Studio, kas ir pieejama uz Mac un Windows platformām. Visual Studio Community versiju var lejupielādēt bez maksas, bet šai versijai ir ierobežojumi, daži no tiem ir, ka drīkst izmantot tikai mazas izstrādes komandas un produktam jābūt atvērtā pirmkoda. Professional un Enterprise versiju var abonēt uz gadu par $539 un $2999 respektīvi, kā arī abiem ir pieejama izmēģinājuma versija uz 30 dienām.

Tiek piedāvāti arī testēšanas risinājumi. Xamarin Test Recorder ir rīks, kas automatizē lietotnes saskarnes testēšanu. Ar to ir iespējams ierakstīt lietotāja pieskārienus, tādā veidā simulējot navigāciju pa lietotni. Xamarin Test Cloud ir tiešsaistes rīks, kas izpilda izveidotos vienībtestus un lietotāja saskarnes testus uz vairāk kā 2000 ierīcēm, sniedzot pārskatu par testēšanas rezultātiem. Šī pakalpojuma mēneša maksa ir no $99 līdz $799 mēnesī.

3.2. Qt ietvars

Qt ietvars atbalsta starpplatformu programmatūras izveidi gan mobilajām ierīcēm, gan datoriem. Izveidots 2011. gadā un pēdējā stabilā versija izlaista šī gada 23. janvārī. Ietvara pirmkods ir pieejams zem GPL un LGPL licences. Qt kompānija darbojās 8 valstīs un nodarbina vairāk kā 200 darbinieku [17]. Atbalstītās platformas ir Windows, Linux/X11, macOS, Android, iOS un Windows Phone. Izstrādes programmēšanas valoda ir C++ un lietotnes saskarnes veidošana notiek izmantojot QML. Integrētā izstrādes vide ir Qt Creator, kas darbojās uz Windows, macOS un Linux platformām. Tiek iekļauta arī Qt Quick programmatūra, kas domāta lietotāja saskarnes veidošanai.

Bezmaksas versijai ir ierobežojumi uz izstrādātās lietotnes lietošanu komerciālos nolūkos. Komerciālā versija vienam izstrādātājam, ja abonē uz gadu, izmaksā $295 mēnesī. Tā iekļauj sevī oficiālu atbalstu no Qt kompānijas. Tiek piedāvāta atlaide tiem uzņēmumiem, kas klasificējas kā jaunuzņēmums (start-up), mēneša maksa šajā gadījumā ir $79.

3.3. Codename One ietvars

Izveidots 2012. gadā un pēdējā stabilā versija ir izlaista šī gada 16. janvārī. Codename One ietvara pirmkods ir pieejams zem GPL licences. Pēc Codename One datiem aptuveni 40 tūkstoši izstrādātāju lieto šo produktu [18]. Ietvars atbalsta 4 mobilās operētājsistēmas: Android, iOS, Windows Phone un Blackberry. Izstrādes programmēšanas valoda ir Java.

Atbalstītās izstrādes vides ir NetBeans, Eclipse un IntelliJ IDEA, jo tām ir pieejams Codename One spraudnis. Eclipse vidē spraudnis lejupielādēts 6510 reizes, NetBeans vidē 175595 reizes un IntelliJ vidē 22539 reizes, kas kopā veido 204644 lejupielādes. Bezmaksas versijā mobilās lietotnes kompilāciju ir iespējams veikt tikai caur viņu mākoni, kam ir savi plusi un mīnusi. Šāda veida risinājums ļauj veikt lietotņu izstrādi arī no Linux platformas, bet līdz ar to izstrādātāji ir atkarīgi no mākoņa pakalpojuma. Lai veiktu kompilāciju uz lokālās iekārtas, no sākuma vajag abonēt Enterprise versiju, kas gadā izmaksā $3799, un pēc tam veikt vēl nepieciešamo rīku uzstādīšanu.

Lietotnes saskarnes veidošana notiek, galvenokārt, izmantojot programmēšanas valodu Java, bet ir pieejams arī vizuālais rīks saskarnes veidošanai.

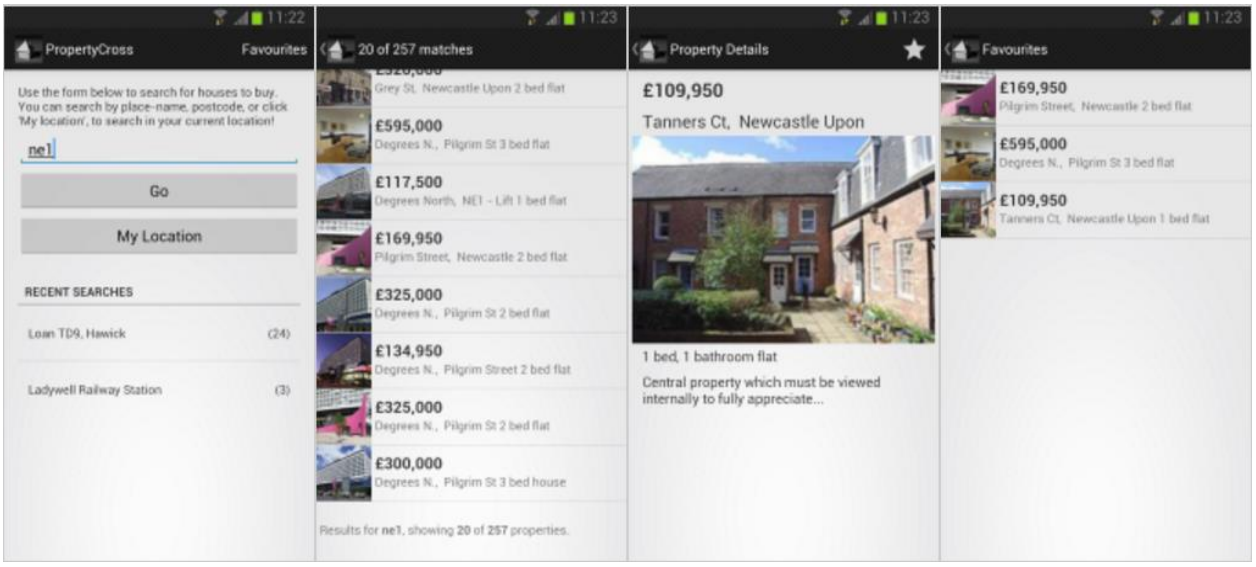
4. IETVARU SALĪDZINĀJUMS

Sākumā ietvari tiek salīdzināti pēc metrikām kā, piemēram, ātrdarbība un kopīgais pirmkods. Pēc tam tie tiek salīdzināti pēc kritērijiem, balstoties uz analītiski hierarhisko procesu, ko izstrādājis Thomas L. Saaty [19].

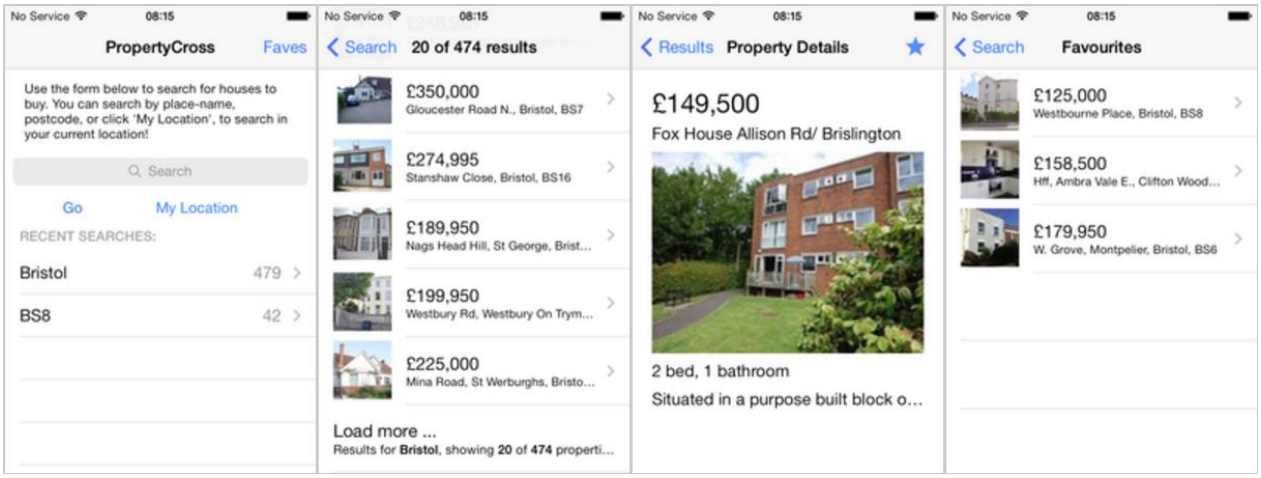
4.1. Pēc metrikām

Autors veic ietvaru salīdzināšanu pēc 4 metrikām – CPU lietojums, kopīgais kods, ātrdarbība un lietotnes datnes izmērs. Mērījumu veikšanai tiek izmantots Nexus5X viedtālrunis un mērījumi tiek analizēti pēc bāzes stāvokļa, kas šajā gadījumā ir natīvās lietotnes mērījumi. Tiek izvērtēts arī kopējais ietvaru sniegums pēc visām 4 metrikām.

Darba sākumā autors domāja ar katru no izvēlētajiem ietvariem izveidot nelielu lietotni, lai varētu veikt salīdzināšanu, bet darba izstrādes laikā autors atrada atvērtā pirmkoda projektu - Property Cross [20]. Projekts piedāvā jau ar vairākiem starpplatformu ietvariem izstrādātas lietotnes tieši, lai veiktu salīdzināšanu. Ar visiem trim autora izvēlētajiem ietvariem ir izstrādāta netriviāla lietotne, kurai ir vairāki skati, lapu navigācija, ģeolokācija, datu glabāšana un Interneta pieprasījumi. Ar lietotni ir iespējams meklēt Anglijas īpašumus, kuri ir pārdošanā. Android natīvās lietotnes skati redzami 4.1. att., turpretī iOS natīvās lietotnes skati redzami 4.2. att. [20].



4.1. att. **Android natīvā lietotne – PropertyCross**



4.2. att. **iOS natīvā lietotne – PropertyCross**

Visu lietotņu pirmkods ir pieejams GitHub repozitorijā [21]. Kā redzams pēc vēstures, tad lietotnes Codename One pirmkods pēdējo reizi labots pirms 9 mēnešiem, Qt un iOS pirms 2 gadiem, Xamarin pirms 3 gadiem un Android natīvā lietotne pirms 4 gadiem. Visām lietotnēm, izņemot Codename One, bija problēmas ar Nestoria API [22]. Nestoria ir Anglijas īpašumu meklēšanas dzinis. Galvenā problēma tam, ka lietotnes nestrādāja, ir saistīta ar to, ka Nestoria pārgāja no HTTP uz HTTPS pieprasījumiem un pārtrauca atbalstīt HTTP pieprasījumus. Autors veica nepieciešamo labojumu, bet dažām lietotnēm, lai tās sāktu strādāt, ar to nepietika. Qt neiekļauj sevī OpenSSL atbalstu, jo ir problēmas ar licenci dažās valstīs. Lai lietotnes HTTPS pieprasījumi veiksmīgi izpildītos, nācās pievienot OpenSSL bibliotēku. Xamarin lietotnei līdz veiksmīga pieprasījuma apstrādei pietrūka pareiza tipu konvertācija. Visi labojumi ar pull pieprasījumu tika augšupielādēti GitHub repozitorijā, kuri 26. maijā tika apstiprināti.

4.1.1. CPU lietojums

Katra lietotne tika mērīta 3 minūtes, kuru laikā tika izmantota visa pieejamā funkcionalitāte:

- meklēti īpašumi pēc pilsētas nosaukuma;
- pieprasīta nākamā saraksta lapa;
- īpašuma pievienošana, dzēšana un apskatīšana no favorītu lapas;
- pieprasīta īpašuma detalizēta informācija;
- meklēšana no nesen veiktajiem meklēšanas pieprasījumiem.

Mērījumi tiek veikti ar Trepn Profiler lietotni, kas aprēķināja vidējo baterijas patēriņu milivatos un vidējo CPU noslodzi procentos. Iegūtie mērījumi ir apkopoti 4.1. tabulā.

4.1. tabula

Vidējā CPU noslodze un baterijas patēriņš

Ietvars	Vidējais baterijas patēriņš (mW)	Vidējā CPU noslodze (%)
Natīva lietotne	495	19
Xamarin	319	14
Qt	505	19
Codename One	411	17

Pirms tika veikti mērījumi autors domāja, ka viszemākais jaudas patēriņš būs natīvai lietotnei. Kā redzams pēc mērījumiem, tad viszemākais jaudas patēriņš ir Xamarin lietotnei (319mW), to var skaidrot ar to, ka Xamarin iekļauj sevī papildus optimizācijas stratēģijas [23]. Savukārt natīvās lietotnes gadījumā papildus optimizācijas stratēģijas ir jāveido pašam. Zemāku jaudas patēriņu par natīvo lietotni uzrāda arī Codename One lietotne, bet pārsvars nav liels, līdz ar to atšķirība visticamāk ir saistīta ar realizāciju.

Pēc šīs metrikas uzvarētājs ir Xamarin, kam seko Codename One, un visbeidzot Qt.

4.1.2. Kopīgais pirmkods

Programmas cloc iegūtie dati katrai lietotnei redzami 1., 2. un 3. pielikumā. Šo datu apkopojums ir redzams 4.2. tabulā. Codename One lietotnes pirmkoda rindiņu skaits būtiski atšķiras no pārējo 2 lietotņu pirmkoda rindiņu skaita. Tas varētu būt skaidrojams ar to, ka Codename One lietotne izveidota pirms 9 mēnešiem, turpretī Qt pirms 2 gadiem un Xamarin pirms 3 gadiem. Vēl viens faktors, kas to varētu ietekmēt, ir natīvais izskats, jo Codename One lietotne ir vienīgā no šīm 3, kas neizskatās natīva.

4.2. tabula

Kopīgais pirmkods

Ietvars	Kopējais pirmkoda rindiņu skaits	Kopīgais pirmkoda rindiņu skaits	Kopīgais pirmkods (%)
Natīva lietotne	2422	0	0
Xamarin	3576	1617	45,2
Qt	1962	1953	99,5
Codename One	405	405	100

Rezultāti atbilst tam, ko sola ietvaru izstrādātāji. Xamarin piedāvā lietotāja saskarni taisīt katrai platformai atsevišķi, izmantojot Xamarin.Andorid un Xamarin.iOS, līdz ar to noteikti būs salīdzinoši daudz platformu specifiska pirmkoda. Savukārt Qt un Codename One vairāk fokusējās uz vienu pirmkoda bāzi vairākām platformām.

Pēc šīs metrikas uzvarētājs ir Codename One, kam ļoti tuvu ir Qt, kas velta 5 pirmkoda rindiņas Android izskata konfigurācijai un 4 rindiņas iOS.

4.1.3. Ātrdarbība

Ielādes laika fiksēšanai tiek izmantots LogCat Displayed atribūts [24]. Mērījumi tiek atkārtoti trīs reizes un tiek aprēķināta to vidējā vērtība. Ielādes laiki katrai lietotnei redzami 4. pielikumā. Vidējais aktivitāšu pārslēgšanās laiks tiek iegūts no favorītu, īpašuma saraksta un īpašuma detalizētas informācijas skatu ielādes laikiem. Mērījumi apkopoti 4.3. tabulā.

4.3. tabula

Lietotņu ielādes laiki

Ietvars	Vidējais ielādes laiks (ms)	Vidējais aktivitāšu pārslēgšanās laiks (ms)
Natīva lietotne	253	112
Xamarin	1416	123
Qt	348	133
Codename One	289	136

Ātrākais ielādes laiks ir natīvai lietotnei, savukārt būtiski lielāks ielādes laiks ir tikai Xamarin lietotnei. Vidējais aktivitāšu pārslēgšanās laiks arī ir viszemākais natīvai lietotnei, bet pārējām lietotnēm tas nav būtiski lielāks.

Vidējā vieta pēc abām kategorijām visiem 3 ietvariem ir vienāda – 2. vieta, līdz ar to viena uzvarētāja šajā kategorijā nav.

4.1.4. Lietotnes datnes izmērs

Lietotnes datnes izmērs tiek nolasīts no datnes, kas tiek izveidota pēc Android kompilācijas. Pēc lietotnes instalācijas izmantotā viedtālruņa iekšējā atmiņa tiek nolasīta no lietotnes informācijas, kas atrodama Android operētājsistēmā. Iegūtie dati apkopoti 4.4. tabulā.

4.4. tabula

Lietotņu datnes izmēri un izmantotā iekšējā atmiņa

Ietvars	Lietotnes datnes izmērs (MB)	Pēc instalācijas izmantotā viedtālruņa iekšējā atmiņa (MB)
Natīva lietotne	1	2,6
Xamarin	10	16
Qt	12	43
Codename One	2,8	8

Kā redzams 4.4. tabulā, tad visi ietvari būtiski palielina lietotnes datnes izmēru, tas skaidrojams ar nepieciešamajām bibliotēkām, ko tie iekļauj. Pēc šis metrikas uzvarētājs ir Codename One ietvars.

4.1.5. Kopējais sniegums

Kopējais sniegums tiek iegūts saskaitot punktu skaitu, ko veido iegūtā vieta pēc katras no metrikām. Viens punkts tiek piešķirts par 1. vietu un trīs punkti par 3. vietu, līdz ar to jo mazāks kopējais punktu skaits, jo labāk. Rezultāti apkopoti 4.5. tabulā.

4.5. tabula

Kopējais sniegums pēc metrikām

Ietvars	CPU lietojums	Kopīgais kods	Ātrdarbība	Lietotnes datnes izmērs	Kopā
Xamarin	1. vieta	3. vieta	2. vieta	2. vieta	8
Qt	3. vieta	2. vieta	2. vieta	3. vieta	10
Codename One	2. vieta	1. vieta	2. vieta	1. vieta	6

Salīdzinot ietvarus pēc metrikām, labāko rezultātu uzrāda Codename One ietvars, kam seko Xamarin un Qt.

4.2. Pēc kritērijiem

Pēc ietvaru salīdzināšanas ar metrikām tie tiek salīdzināti pēc analītiski hierarhiskā procesa. Šī analītiskā metode balstās uz ekspertu izstrādātu prioritāšu skalu un salīdzināšana tiek veikta pēc

tā, cik ļoti viens elements dominē pār otru. Tiek izmantota Thomas L. Saaty piedāvātā skala, kas raksturo, cik reizes viens elements dominē pār otru, attiecībā pret izvēlēto kritēriju (4.6. tabula) [19].

4.6. tabula

Fundamentāla absolūto skaitļu skala

Svarīguma intensitāte	**Definīcija**	**Paskaidrojums**
1	Vienlīdz svarīgi	Divi elementi sasniedz mērķi vienlīdzīgi
2	Vājš pārsvars	
3	Mērens pārsvars	Pieredze un novērojumi nedaudz par labu vienam elementam
4	Starp mērenu un spēcīgu	
5	Spēcīgs pārsvars	Pieredze un novērojumi daudz par labu vienam elementam
6	Starp spēcīgu un ļoti spēcīgu	
7	Ļoti spēcīgs pārsvars	Pārākums pierādīts praksē
8	Ļoti, ļoti spēcīgs pārsvars	
9	Ārkārtīgi svarīgs	Augstākais iespējamais apliecinājums

Prioritāšu skala uz kuru autors balstās ir izveidota intervējot 6 izstrādātājus ar vairāk kā 10 gadu stāžu nozarē (4.7. tabula) [25]. Tabulā redzams, ka viss augstākais kopējais vērtējums tiek dots iespējām.

4.7. tabula

Kritēriju pārisks (pairwise) salīdzinājums

	Iespējas	**Izstrādes ātrums**	**Natīvs UI**	**Apguves līkne**	**Piekļuve ierīcei**
Iespējas	1	3	7	5	5
Izstrādes ātrums	0,333	1	4	3	2
Natīvs UI	0,143	0,25	1	0,5	0,5
Apguves līkne	0,2	0,333	4	1	2
Piekļuve ierīcei	0,2	0,5	2	0,5	1

Autora personīgā pieredze mobilo lietotņu izstrādes industrijā ir 2 gadi, kuras laikā ir strādāts pie natīvās Android un iOS lietotnes izstrādes, kā arī pie hibrīda lietotnes varianta. Skatoties uz prioritāšu skalu, autors vienīgi mainītu veiktspējas svarīgumu uz lielāku.

4.2.1. Apguves līkne

Novērtēšana ir veikta apvienojot ietvara programmēšanas valodas popularitātes un ietvara dokumentācijas rezultātus. Ietvaru programmēšanas valodu savstarpējās salīdzināšanas piemērs - salīdzinot C# un C++ attiecīgie popularitātes procenti ir 0,034 * 100 % / (0,034 + 0,047) = 42% un 100% - 42% = 58%. Pēc fundamentālo skaitļu skalas 100% atbilst 9 un 50% atbilst 1. Rezultāti pēc programmēšanas valodām ir apkopoti 4.8. tabulā.

4.8. tabula

Ietvaru salīdzinājums pēc programmēšanas valodām

Ietvars	**Xamarin**	**Qt**	**Codename One**
Xamarin	1	0,43	0,17
Qt	2,28	1	0,2
Codename One	5,96	5	1

Dokumentācijas novērtēšana tiek veikta pēc tās dažādības. Autors dokumentāciju ir iedalījis šādās kategorijās: dokumentētas parauga lietotnes, rakstveida pamācības, video pamācības un

diskusijas. Ja kategoriju skaits abiem ietvariem ir vienāds, tad svarīguma intensitāte ir 1, ja skaits atšķiras pa 4, tad svarīguma intensitāte ir 9. Xamarin University un Xamarin Forums sevī iekļauj visas kategorijas un papildus tam ir pieejami arī testi. Qt arī sevī iekļauj visas kategorijas, kur dažas no tām izpaužas kā tīmekļsemināri (webinars). Lai gan Codename One arī piedāvā visas minētās kategorijas, ir vērts pieminēt, ka to kvalitāte ir zemāka un to daudzums ir mazāks. Rezultāti pēc dokumentācijas dažādības ir apkopoti 4.9. tabulā.

4.9. tabula

Ietvaru salīdzinājums pēc dokumentācijas dažādības

Ietvars	Xamarin	Qt	Codename One
Xamarin	1	1	1
Qt	1	1	1
Codename One	1	1	1

Tā kā dokumentācijas dažādība visiem trim ietvariem ir vienādā, tad apguvēs līknes rezultāti ir redzami 4.8. tabulā. Šī kritērija uzvarētājs ir Codename One ietvars.

4.2.2. Lietotāja saskarnes izveide

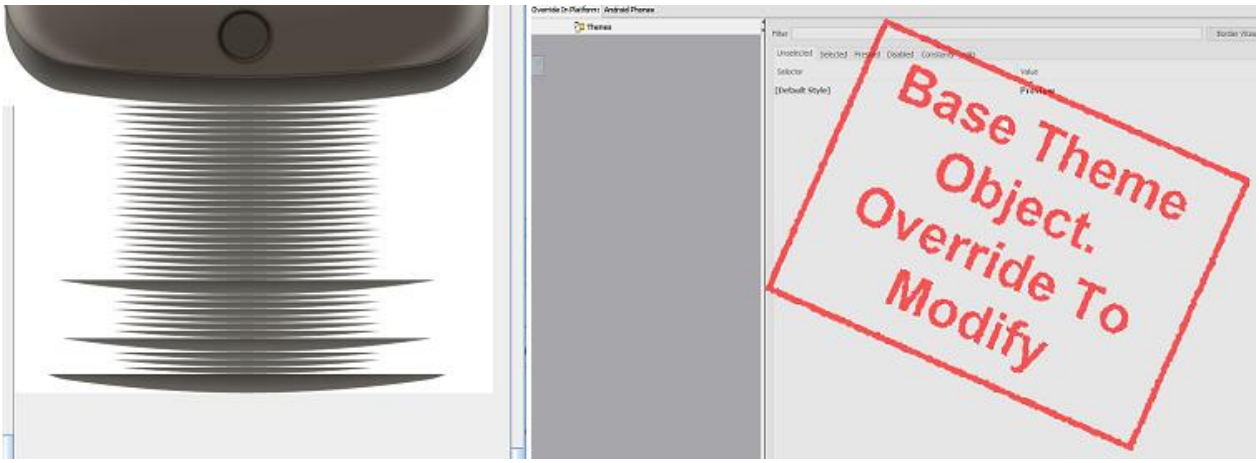
Visiem izvelētajiem ietvariem ir grafisks lietotāja saskarnes izveides rīks. Autors izmēģina katru no tiem un sniedz vērtējumu, balstoties uz saviem personīgajiem iespaidiem.

Qt ietvaram šī rīka nosaukums ir Qt Designer, tas tiek instalēts kopā ar integrēto izstrādes vidi. Visas apskatītās funkcijas strādā, tas ir, nebija novērojama kļūda rīka darbībā. Pietrūkst iespēja izmantot iepriekš definētus viedtālruņu ekrāna izmērus, lai pārbaudītu, kā izveidotā forma izskatās uz dažādiem ekrāniem.

Arī Xamarin ietvaram šis rīks tiek instalēts kopā ar integrēto izstrādes vidi, patiesībā pat divi, jo ir viens priekš Android un otrs priekš iOS. Tā kā rīks ir specializēts uz konkrēto platformu, tad rīka iespējas tuvinās Android Studio iespējām, kas ir Andorid standarta izstrādes rīks. Ir iespēja izvēlēties starp iepriekš definētiem viedtālruņa ekrāna izmēriem un Android versijām, kas ļauj visprecīzāk redzēt, kā lietotne izskatīsies uz konkrētā telefona, lietojot natīvas komponentes. Elementu klāsts no kuriem var izvēlēties ir plašs.

Vislielākās problēmas sagādā Codename One vizuālais saskarnes izstrādes rīks. Veidojot skatus ar šo rīku, tie tiek saglabāti vienā lielā failā, kuru var atvērt tikai ar šo rīku, līdz ar to nav

iespējams veikt korekcijas caur parastu teksta redaktoru. Autors saskata šeit problēmas ar versijas kontroles rīku, jo būs praktiski neiespējami izsekot līdzi skatu izstrādes gaitai. Izmēģinot dažādas rīka piedāvātās iespējas, tika novēroti defekti vai neprofesionāla pieeja. Pārvietojot emulatora loga saturu, var novērot vizuālu defektu, kas redzams 4.3. att. kreisajā pusē. Savukārt izvēloties iespēju "pārrakstīt platformā" parādās 4.4. att. labajā puse redzamais teksts. Nav skaidrs, vai tas ir kļūdas paziņojums, un šķiet dīvaini, ka zem šī teksta redzamos elementus var rediģēt.



4.3. att. **Codename One vizuālā izstrādes rīka īpatnības**

Autora vērtējumi, salīdzinot ietvarus savā starpā pēc vizuālā izstrādes rīka, ir apkopoti 4.10. tabulā.

4.10. tabula

Ietvaru salīdzinājums pēc vizuālā lietotāja saskarnes izstrādes rīka

Ietvars	**Xamarin**	**Qt**	**Codename One**
Xamarin	1	3	7
Qt	0,33	1	4
Codename One	0,14	0,25	1

Codename One saskarnes izstrādes rīks būtiski atpaliek no saviem konkurentiem gan iespēju, gan kvalitātes ziņā.

4.2.3. Izmaksas un licence

Visiem trim ietvariem SDK ir atvērtā pirmkoda, vienīgi atšķiras licences tips. Ietvaru licences apkopotas 4.11. tabulā. GPL licence ir stingrākā par MIT licenci, jo GPL licence nosaka, ka veiktajām modifikācijām jābūt atvērtā pirmkoda, kurpretim MIT ļauj veidot aizvērtā pirmkoda projektu.

4.11. tabula

Ietvaru licences

Ietvars	Licence
Xamarin	MIT
Qt	GPL un LGPL
Codename One	GPL

Visām trim kompānijām nav viens kopīgs peļņas gūšanas modelis. Autors uzskaita katra ietvara viena izstrādātāja izmaksas gadā, kas ļauj izstrādāto projektu izmantot komerciālos nolūkos. Xamarin ietvara izmantošanas izmaksas sastāda Microsoft Visual Studio iegāde, $539 par Professional versiju un $2999 par Enterprise versiju. Qt ietvara izmaksas arī sastāda integrētās izstrādes vides iegāde, kas maksā $3540. Savukārt Codename One piedāvā bezmaksas izstrādi arī komerciālos nolūkos, bet ar ierobežojumiem. Gandrīz visu iespēju izmantošana maksā $879 un pilnā pakete maksā $3799. Ietvaru izmaksu apkopojums ir redzams 4.12. tabulā.

4.12. tabula

Ietvaru izmaksas

Ietvars	Gada maksa vienam izstrādātājam
Xamarin	$539/$2999
Qt	$3540
Codename One	$0/$228/$879/$3799

Vienīgais ietvars, kas piedāvā bezmaksas izstrādi komerciālos nolūkos ir Codename One, bet pēc šī plāna nevar izstrādāt kārtīgu lietotni, jo mēnesī maksimālais iOS kompilāciju skaits ir 12. Salīdzinot pēc cenām, kas dod pilnu piekļuvi visām iespējām, Xamarin piedāvājums ir lētākais.

4.2.4. Piekļuve ierīcei

Novērtēšana veikta pēc atbalstīto API skaita, autors izvēlējies 8 API, kas redzami 4.13. tabulā [26]. Ar plusiņu apzīmēts API, ko atbalsta ietvars.

4.13. tabula

Ietvaru atbalstītie platformas API

API nosaukums	Xamarin	Qt	Codename One
Sensori	+	+	+
Bluetooth	+	+	+
Kalendārs	+	+	+
Kamera	+	+	+
Kontakti		+	+
Faili	+	+	+
NFC	+	+	
Paziņojumi	+	+	+

Visus izvēlētos API atbalsta Qt, bet Xamarin un Codename One pietrūkst viena API atbalsta. Ja API skaits abiem ietvariem ir vienāds, tad svarīguma intensitāte ir 1, ja skaits atšķiras par 8, tad svarīguma intensitāte ir 9. Rezultāti pēc atbalstīto API skaita ir apkopoti 4.14. tabulā.

4.14. tabula

Ietvaru salīdzinājums pēc atbalstīto API skaita

Ietvars	Xamarin	Qt	Codename One
Xamarin	1	0,5	1
Qt	2	1	2
Codename One	1	0,5	1

Pēc šī kritērija uzvarētājs ir Qt ietvars.

4.2.5. Integrētā izstrādes vide

Visiem ietvariem ir integrētās izstrādes vides, dažiem vairāk pazīstams, dažiem mazāk pazīstamas, jo ir specifiskas tieši ietvaram. Novērtēšana tiek veikta balstoties uz izstrādes vides popularitāti. Visas galvenās funkcijas, kas minētas 2.3.5. nodaļā, katra no izstrādes vidēm atbalsta.

Xamarin izstrādes vide ir Microsoft Visual Studio, kas samērā daudziem programmētājiem, ir pazīstama un noteikti gandrīz katrs programmētājs ir dzirdējis par tādu. Izstrādes videi ir labs profilētājs, kas palīdz analizēt izstrādātās programmas darbību, piemēram, parāda funkcijas, kuru izpilde aizņem vislielāko laiku. Ir iespēja instalēt kādu no vairākiem paplašinājumiem, kas ir pieejami veikalā.

Qt izstrādes vide ir Qt Creator, tā ir specifiska, jo tā paredzēta tikai Qt projektu izstrādei. Tikai retais programmētājs būs lietojis šo vidi. Arī šai videi ir profilētājs. Dokumentācija par izstrādes vidi ir daudz un tā ir iekļauta vidē iekšā. Lielākais šīs vides plus ir, ka tā darbojās uz Windows, iOS un Linux platformām. Bet tā nav vienīga vide ar kuru var izstrādāt Qt projektus, otra piedāvāta ir Visual Studio, tai ir izstrādāts Qt paplašinājums.

Codename One atbalstītās izstrādes vides ir Netbeans, Eclipse un IntelliJ IDEA. Pozitīvi ir vērtējams tas, ka var izvelēties starp vairākām populārām izstrādes vidēm.

Vērtības tiek aprēķinātas līdzīgi kā 4.2.1. nodaļā, veicot ietvaru salīdzināšanu, pēc programmēšanas valodām. Rezultāti apkopoti 4.15. tabulā

4.15. tabula

Ietvaru salīdzinājums pēc integrētās izstrādes vides

Ietvars	Xamarin	Qt	Codename One
Xamarin	1	6,76	1,48
Qt	0,15	1	0,16
Codename One	0,67	6,44	1

Qt izstrādes vide būtiski zaudē saviem konkurentiem, tas varētu būt iemesls, kāpēc Qt piedāvā savu paplašinājumu Visual Studio (2013. un 2015. gada versijas) videi.

4.2.6. Izstrādes ātrums

Novērtēšana tiek veikta apkopojot vairākus kritērijus: apguves līkni, lietotāja saskarnes izveidi, integrētās izstrādes vidi un kopīgo pirmkodu.

Kopīga pirmkoda novērtējumu, pēc fundamentālo absolūto skaitļus skalas, iegūst, balstoties uz 4.1.2 apakšnodaļā veiktajiem mērījumiem, pēc kopīgā pirmkoda metrikas. Rezultāti apkopoti 4.16. tabulā.

4.16. tabula

Ietvaru salīdzinājums pēc kopīgā pirmkoda

Ietvars	Xamarin	Qt	Codename One
Xamarin	1	0,25	0,25
Qt	4,04	1	1
Codename One	4,04	1	1

Tā kā Qt un Codename One rezultāti, pēc kopīga pirmkoda metrikas, ir ļoti tuvi, tad šeit to vērtības ir vienādotas.

Gala vērtība tiek noteikta, pēc visu kritēriju vidējo vērtību attiecības. Tālāk tiek parādīts aprēķina piemērs. Skatoties, cik Xamarin ietvars dominē pār Qt, saskaita katra kritērija vērības iegūstot 0,43 + 3 + 6,76 + 0,25 = 10,44. Vidējā vērtība ir 10,44 / 4 = 2,61. Tā pat iegūst vidējo vērtību skatoties, cik Qt ietvars dominē pār Xamarin. Vidējā vērtība ir (2,28 + 0,33 + 0,15 + 4,04) / 4 = 1,7. Tagad, tā pat kā iepriekš, tiek aprēķināta savstarpējā attiecība. Kopējais novērtējums pēc izstrādes ātruma, ir apkopots 4.17. tabulā.

4.17. tabula

Ietvaru salīdzinājums pēc izstrādes ātruma

Ietvars	Xamarin	Qt	Codename One
Xamarin	1	1,76	0,56
Qt	0,57	1	0,24
Codename One	1,8	4,2	1

Codename One ir izteikts līderis, kam seko Xamarin un Qt.

4.2.7. Natīvs UI

Ar Xamarin var iegūt ļoti labu natīvu izskatu un sajūtu, jo katrai platformai tiek taisīta sava saskarne. Qt nav natīvā iOS izskata, bet ir Android Material Design stils, līdz ar to Qt izstrādāta

lietotne izskatās natīva tikai uz Android platformām [27]. Codename One piedāvā natīvu izskatu, bet rezultāts nav tāds kādu gribētos, tas ir, ir redzamas atšķirības un pat daži defekti. Autors instalēja viņu piedāvāto natīvas lietotnes piemēru uz Android viedtālruņa un novēroja sekojošās atšķirības. Teksta lauka pagaidu (placeholder) teksts ir izbīdīts vairāk pa kreisi, līdz ar to teksta kursors atrodas tieši virsū pirmajam pagaidu teksta burtam. Pogas iezīmēšanas krāsa arī ir cita.

Autors, balstoties uz saviem novērojumiem, ietvaru salīdzināšanas rezultātus pēc natīvā UI, ir apkopojis 4.18. tabulā.

4.18. tabula

Ietvaru salīdzinājums pēc natīva UI

Ietvars	Xamarin	Qt	Codename One
Xamarin	1	3	5
Qt	0,33	1	2
Codename One	0,2	0,5	1

Autors ietvarus pēc natīva UI iespējām ir ierindojis šādā secībā: Xamarin, Qt un Codename One.

4.2.8. Kopējais sniegums

Ietvaru savstarpējais sniegums pēc visiem kritērijiem, tiek aprēķināts balstoties uz kritēriju vērtībām, kas pareizinātas ar koeficientiem, kas iegūti no 4.7. tabulas. Kritērija koeficients ir visu 4.7. tabulā punktu attiecība pret kritērija punktu summu. Visu punktu summa ir 48,753, līdz ar to iespēju kritērija koeficients ir 48,753 / 25 = 0,5129. Līdzīgi tiek iegūta arī kritēriju vērtību attiecība, tiek saskaitīts kopējais punktu skaits pēc kritērija, ar kuru dala ietvara kopējo punktu skaitu pēc kritērija. Piemēram, pēc natīva UI kritērija, Xamarin punktu skaits ir 3 + 5 = 8, tas tiek dalīts ar kopējo punktu skaitu, kas ir 3 + 5 + 0,33 + 2 + 0,2 + 0,5 = 11,03, rezultāta iegūstot 0,7253. Iespējas ir ietvara iespējas, kas ir vidējā vērtība, pēc lietotāja saskarnes izveides un integrētās izstrādes vides kritērija. Kopējā snieguma rezultāti apkopoti 4.19. tabulā.

4.19. tabula

Ietvaru vērtējums pēc visiem kritērijiem

Ietvars	**Iespējas**	**Izstrādes ātrums**	**Natīvs UI**	**Apguves līkne**	**Piekļuve ierīcei**	**Vērtējums**
(koeficients)	0,5129	0,2318	0,0353	0,1441	0,0759	1
Xamarin	0,5975	0,2541	0,7253	0,0428	0,2143	0,41337
Qt	0,1529	0,0888	0,2112	0,1766	0,5714	0,17525
Codename One	0,2469	0,6571	0,0644	0,7806	0,2143	0,40995

Augstākai vērtējums ir Xamarin ietvaram, kam tur pat līdzas ir Codename One, savukārt pēdējais ir Qt ietvars ar lielāku atstarpi.

REZULTĀTI

Populārākās programmatūras mērīšanas metodes balstās uz pirmkoda rindiņu skaitu vai zarošanos, bet ar šādām metodēm var vienīgi noteikt programmas izmēru un sarežģītību. Grūtāk ir nomērīt programmas kvalitāti, lietderīgumu, efektivitāti un lietotāja saskarni. Šādas lietas parasti novērtē pēc aptaujām.

Eksistē vairāki līdzīgi darbi par šo tēmu, bet neviens no viņiem nav pilnīgs. Trūkst metrikas, pēc kurām var nomērīt dažādus ietvara aspektus, un kritēriji, kas precīzāk novērtē lietotāja saskarni.

No darbā aprakstītajām 4 metrikām, viena nebija sastopama citos autora apskatītajos darbos šajā tēmā. Šī metrika ir lietotnes datnes izmērs. Kā noskaidrojās pēc mērījumiem, tad lietotnes datnes izmēri būtiski atšķīrās. Mērījumu veikšanai izmantotā PropertyCross lietotnes realizācija ar Xamarin, Qt un Android tika salabota un GitHub izmaiņu pull pieprasījums tika apstiprināts 26. maijā.

Trīs populārākie starpplatformu ietvari, kas neizmanto tīmekļa tehnoloģijas, ir Xamarin, Qt un Codename One. Autors uzzināja, ka eksistē ietvari, kas izstrādē izmanto tīmekļa tehnoloģijas un veic natīvu kompilāciju, līdz ar to izslēdzot daudzus mīnusus, ko sagādāja autoram pazīstamā lietotņu izstrāde ar tīmekļa tehnoloģijām.

Pēc iepazīšanās ar katru no izvēlētajiem ietvariem, autors kā labāko uzskatīja Qt ietvaru, kam seko Xamarin un tad Codename One. Salīdzinot ietvarus pēc metrikām, līderis ir Codename One, otrajā vieta ir Xamarin un trešajā vietā ir Qt. Qt uzrāda zemus rezultātus, jo lietotnes datnes izmērs ir liels, un tas varētu būt iemesls, kāpēc CPU lietojums arī ir vislielākais. Savukārt Xamarin vienīgais būtiskais trūkums šajā gadījumā ir mazs kopīgā pirmkoda daudzums. Salīdzinot ietvarus pēc kritērijiem, līderis ir Xamarin, kam seko Codename One un Qt. Xamarin lielākie plusi ir laba integrētā izstrādes vide un labs lietotāja saskarnes izveides rīks. Savukārt Codename One lielākā priekšrocība ir izmantotā programmēšanas valoda. Visbeidzot Qt vienīgā priekšrocība ir piekļuve ierīcei.

Skatoties tikai uz metriku rezultātiem, tad ietvars, kas būtu jāizvēlas, ir Codename One, savukārt, skatoties uz kritēriju rezultātiem, tad būtu jāizvēlas Xamarin ietvars. Bet tie ir ieteikumi, kas balstīti uz ietvara vidējo sniegumu, pēc visiem kritērijiem. Bieži vien projekti ir specifiski un ar dažādām prioritātēm, līdz ar to ir dažādi svari uz konkrētiem kritērijiem.

SECINĀJUMI

Jautājumam: “Kuru no starpplatformu ietvariem izvēlēties?” Ir vairākas atbildes. Katram ietvaram ir savi plusi un mīnusi, lai izvēlētos pareizo ir jāsaprot, kādas ir galvenās projekta vajadzības, un jāizvēlas ietvars, kas vislabāk atbilst šīm vajadzībām. Ja svarīga ir natīva lietotnes saskarne, tad jāskatās, kā ietvars atrisina saskarnes izskatu uz iOS platformas. Ja svarīgs ir izstrādes ātrums, tad jāskatās uz to, cik populāra ir programmēšanas valoda un integrētās izstrādes rīks. Ja kritēriju prioritātes sakrīt ar darbā piedāvātajām, tad jāizvēlas Xamarin ietvars.

Visi darba sākumā izvirzītie uzdevumi tika izpildīti un mērķis tika sasniegts.

Darbu var turpināt dziļāk izpētot programmatūras metrikas un kritērijus, kā arī detalizētāk analizējot lietotņu darbību.

IZMANTOTĀ LITERATŪRA UN AVOTI

1. Latvijas Zinātņu Akadēmijas akadēmiskā terminu datubāze "AkadTerm" [tiešsaiste]. [atsauce 26.05.2017.]. Pieejams Internetā: http://termini.lza.lv/term.php
2. S. Dhillon, un Q. H. Mahmoud, "An Evaluation Framework for Cross-Platform Mobile Application Development Tools," 2013; doi:10.1002/spe.2286.
3. Gartner analīze [tiešsaiste]. [atsauce 26.05.2017.]. Pieejams Internetā: http://www.gartner.com
4. Populārāko lietotņu veikalu 2017. gada marta statistika [tiešsaiste]. [atsauce 26.05.2017.]. Pieejams Internetā: https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores
5. IEEE Std 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology, IEEE, 1990.
6. Horst Zuse, *A Framework of Software Measurement*, Walter de Gruyter, 1998, lpp. 2-7.
7. Pirmkoda metrikas [tiešsaiste]. [atsauce 26.05.2017.]. Pieejams Internetā: https://msdn.microsoft.com/en-us/library/bb385914.aspx
8. Trepn Profiler lietotne [tiešsaiste]. [atsauce 26.05.2017.]. Pieejams Internetā: https://play.google.com/store/apps/details?id=com.quicinc.trepn&hl=lv
9. Programma cloc [tiešsaiste]. [atsauce 26.05.2017.]. Pieejams Internetā: https://github.com/AlDanial/cloc#Quick_Start
10. Appcelerator pētījums [tiešsaiste]. [atsauce 26.05.2017.]. Pieejams Internetā: https://www.appcelerator.com/resource-center/research/2015-mobile-trends-report
11. Vision mobile pētījums par starpplatformu rīkiem [tiešsaiste]. [atsauce 26.05.2017.]. Pieejams Internetā: https://www.slideshare.net/andreasc/vision-mobile-crossplatformdevelopertools2012
12. H. Heitkötter, S. Hanschke, un T. A. Majchrzak, "Evaluating cross-platform development approaches for mobile applications," International Conference on Web Information Systems and Technologies, lpp. 6-7, 2012.
13. TIOBE 2017. gada statistika [tiešsaiste]. [atsauce 26.05.2017.]. Pieejams Internetā: https://www.tiobe.com/tiobe-index

14. Starpplatformu rīku salīdzinājums 2014. gadā [tiešsaiste]. [atsauce 26.05.2017.]. Pieejams Internetā: http://www.research2guidance.com/r2g/Cross-Platform-Tool-Benchmarking-Report-2014.pdf
15. Populārākās integrētās izstrādes vides [tiešsaiste]. [atsauce 27.05.2017.]. Pieejams Internetā: https://pypl.github.io/IDE.html
16. Xamarin partneri [tiešsaiste]. [atsauce 26.05.2017.]. Pieejams Internetā: https://www.xamarin.com/technical-partners
17. Qt kompānija [tiešsaiste]. [atsauce 26.05.2017.]. Pieejams Internetā: https://www.qt.io/company
18. Codename One kompānija [tiešsaiste]. [atsauce 26.05.2017.]. Pieejams Internetā: https://www.codenameone.com
19. T. L. Saaty, "Decision making with the analytic hierarchy process," International J. Services Sciences, vol. 1, no. 1, lpp. 85-86, 2001.
20. Property Cross projekts [tiešsaiste]. [atsauce 26.05.2017.]. Pieejams Internetā: http://propertycross.com
21. Property Cross GitHub lapa [tiešsaiste]. [atsauce 26.05.2017.]. Pieejams Internetā: https://github.com/tastejs/PropertyCross
22. Nestoria API dokumentācija [tiešsaiste]. [atsauce 26.05.2017.]. Pieejams Internetā: https://www.nestoria.co.uk/help/api
23. Xamarin-forms optimizācija [tiešsaiste]. [atsauce 26.05.2017.]. Pieejams Internetā: https://developer.xamarin.com/guides/xamarin-forms/deployment-testing/performance
24. Logcat dokumentācija [tiešsaiste]. [atsauce 26.05.2017.]. Pieejams Internetā: https://developer.android.com/studio/command-line/logcat.html
25. F. Appiah, J. Hayfron-Acquah, J. K. Panford, and F. Twum, "A tool selection framework for cross platform mobile app development," International Journal of Computer Applications, vol. 123, no. 2, 2015.
26. M. Palmieri, I. Singh, and A. Cicchetti, "Comparison of Cross-Platform Mobile Development Tools," 2012; doi:10.1109/ICIN.2012.6376023.
27. Qt lietotnes saskarnes stili [tiešsaiste]. [atsauce 28.05.2017.]. Pieejams Internetā: https://doc.qt.io/qt-5/qtquickcontrols2-styles.html

PIELIKUMI

1. pielikums. Xamarin lietotnes pirmkoda kopīgo un platformu specifisko rindiņu skaits

```
C:\Users\Kristaps\Downloads>cloc.exe C:\Users\Kristaps\Desktop\xamarin\android\PropertyCross
      18 text files.
      18 unique files.
       6 files ignored.

github.com/AlDanial/cloc v 1.72  T=0.50 s (34.0 files/s, 2512.0 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
C#                              16            188             17           1043
XML                              1              0              0              8
-------------------------------------------------------------------------------
SUM:                            17            188             17           1051
-------------------------------------------------------------------------------

C:\Users\Kristaps\Downloads>cloc.exe C:\Users\Kristaps\Desktop\xamarin\common
      27 text files.
      27 unique files.
       6 files ignored.

github.com/AlDanial/cloc v 1.72  T=0.50 s (54.0 files/s, 3834.0 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
C#                              27            176            124           1617
-------------------------------------------------------------------------------
SUM:                            27            176            124           1617
-------------------------------------------------------------------------------

C:\Users\Kristaps\Downloads>cloc.exe C:\Users\Kristaps\Desktop\xamarin\iphone\PropertyCross
      22 text files.
      22 unique files.
       6 files ignored.

github.com/AlDanial/cloc v 1.72  T=0.50 s (34.0 files/s, 2926.0 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
C#                              16            251             75            908
MSBuild script                   1              0              0            229
-------------------------------------------------------------------------------
SUM:                            17            251             75           1137
-------------------------------------------------------------------------------
```

2. pielikums. Qt lietotnes pirmkoda kopīgo un platformu specifisko rindiņu skaits

```
C:\Users\Kristaps\Downloads>cloc.exe C:\Users\Kristaps\Desktop\qt2\PropertyCrossApp\src
       1 text file.
       1 unique file.
       1 file ignored.

github.com/AlDanial/cloc v 1.72  T=0.50 s (2.0 files/s, 126.0 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
C++                              1             10              7             46
-------------------------------------------------------------------------------

C:\Users\Kristaps\Downloads>cloc.exe C:\Users\Kristaps\Desktop\qt2\PropertyCrossApp\qml\+android
       2 text files.
       1 unique file.
       3 files ignored.

github.com/AlDanial/cloc v 1.72  T=0.50 s (2.0 files/s, 16.0 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
QML                              1              3              0              5
-------------------------------------------------------------------------------

C:\Users\Kristaps\Downloads>cloc.exe C:\Users\Kristaps\Desktop\qt2\PropertyCrossApp\qml\+ios
       1 text file.
       1 unique file.
       1 file ignored.

github.com/AlDanial/cloc v 1.72  T=0.50 s (2.0 files/s, 14.0 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
QML                              1              3              0              4
-------------------------------------------------------------------------------

C:\Users\Kristaps\Downloads>cloc.exe C:\Users\Kristaps\Desktop\qt2\PropertyCrossApp\qml
      13 text files.
      12 unique files.
       6 files ignored.

github.com/AlDanial/cloc v 1.72  T=0.50 s (22.0 files/s, 1794.0 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
QML                             11             60             13            824
-------------------------------------------------------------------------------
SUM:                            11             60             13            824
-------------------------------------------------------------------------------

C:\Users\Kristaps\Downloads>cloc.exe C:\Users\Kristaps\Desktop\qt2\PropertyCrossLib
      17 text files.
      16 unique files.
       4 files ignored.

github.com/AlDanial/cloc v 1.72  T=0.50 s (32.0 files/s, 2820.0 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
C++                              7            131             26            748
C/C++ Header                     8             82             73            315
Qt Project                       1              5              1             29
-------------------------------------------------------------------------------
SUM:                            16            218            100           1092
-------------------------------------------------------------------------------
```

3. pielikums. Codename One lietotnes pirmkoda rindiņu skaits

```
C:\Users\Kristaps\Downloads>cloc.exe C:\Users\Kristaps\Desktop\codenameone\src
       1 text file.
       1 unique file.
       4 files ignored.

github.com/AlDanial/cloc v 1.72  T=1.00 s (1.0 files/s, 623.0 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
Java                             1             60            158            405
-------------------------------------------------------------------------------
```

4. pielikums. Lietotņu ielādes laiki

```
Displayed com.propertycross.xamarin.android/md5fba7f0d303769e80e6c6d6a9cb88885a.PropertyCrossView: +1s433ms
Displayed com.propertycross.xamarin.android/md5fba7f0d303769e80e6c6d6a9cb88885a.PropertyCrossView: +1s403ms
Displayed com.propertycross.xamarin.android/md5fba7f0d303769e80e6c6d6a9cb88885a.PropertyCrossView: +1s411ms
Displayed org.propertycross.qt.android/org.qtproject.qt5.android.bindings.QtActivity: +451ms
Displayed org.propertycross.qt.android/org.qtproject.qt5.android.bindings.QtActivity: +284ms
Displayed org.propertycross.qt.android/org.qtproject.qt5.android.bindings.QtActivity: +308ms
Displayed com.propertycross.codename1/.PropertyCrossStub: +406ms
Displayed com.propertycross.codename1/.PropertyCrossStub: +242ms
Displayed com.propertycross.codename1/.PropertyCrossStub: +218ms
Displayed com.propertycross.android/.views.PropertyFinderView: +287ms
Displayed com.propertycross.android/.views.PropertyFinderView: +237ms
Displayed com.propertycross.android/.views.PropertyFinderView: +235ms
Displayed com.propertycross.codename1/.PropertyCrossStub: +306ms
```

Bakalaura darbs „Starpplatformu mobilo lietotņu izstrāde bez tīmekļa tehnoloģijām” 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.

Autors: _____ Kristaps Krūmiņš

Rekomendēju/nerekomendēju darbu aizstāvēšanai

Vadītājs: asociētais profesors, Dr. dat. Uldis Straujums_____ 29.05.2017.

Recenzents: _____

Darbs iesniegts Datorikas fakultātē 29.05.2017.

Dekāna pilnvarotā persona:

vecākā metodiķe Ārija Sproģe_____

Darbs aizstāvēts bakalaura gala pārbaudījuma komisijas sēdē

__.06.2017. prot. Nr. __.

Komisijas sekretāre: _____ _____