# Identifikacija ponovno iskoristivih artefakata pri procesu razvoja mobilnih aplikacija

**Zlatko Stapić**
Sveučilište u Zagrebu, Fakultet organizacije i informatike Varaždin
Laboratorij za primijenjeno softversko inženjerstvo
Pavlinska 2, Varaždin, Croatia
Zlatko.stapic@foi.unizg.hr

***Sažetak***. *Razvoj mobilnih aplikacija je izazovan proces koji se razlikuje od tradicionalnog razvoja, te mora adresirati kako poslovne tako i izazove specifične za razvoj. Ovo istraživanje fokusira izazove specifične za proces razvoja. Glavni cilj ovoga istraživanja je identificirati ponovno iskoristive artefakte koji nastaju u metodički vođenom procesu razvoja mobilnih aplikacija koje se razvijaju za dvije ili više ciljanih platformi, kako bi se kreirala osnova za efikasniji i interoperabilniji proces više-platformskog razvoja mobilnih aplikacija. Nakon identificiranja svih artefakata koji nastaju u procesu razvoja za jednu platformu, proveli smo analizu ponovne iskoristivosti za razvoj za drugu platformu. Utvrdili smo da 66% artefakata može biti potpuno ili djelomično ponovno iskorišteno.*

**Ključne riječi.** razvoj softvera, razvoj mobilnih aplikacija, metodike razvoja, ponovna iskoristivost.

## 1 Uvod

Razvoj aplikacija za mobilne uređaje i klasični razvoj programskih proizvoda se razlikuju u nekoliko važnih aspekata koji čine razvoj za mobilne uređaje izazovnim i zahtjevnim zadatkom. Postoje različite klasifikacije spomenutih različitosti, a na primjer Hosbond (2005) navodi dva osnovna skupa izazova koje treba uzeti u obzir pri mobilnom razvoju. To su izazovi vezani uz poslovne aspekte i specifični izazovi vezani uz razvojne aspekte. U ovom istraživanju naglasak je na specifičnim razvojnim izazovima, te će se posebna pozornost usmjeriti na korištenje metodika razvoja, budući da prema nekim autorima, kao što su Rahimian i Ramsin (2008), Spataru (2010) ili La i Kim (2009), upravo te izazove treba prve adresirati i njima upravljati.

Klasične ili agilne razvojne metodike trebaju biti prilagođene razvoju mobilnih aplikacija, budući da postojeće i najčešće korištene metodike razvoja ne uzimaju u obzir zahtjeve koji se odnose isključivo na mobilne platforme (La & Kim, 2009). Postoje pokušaji nekolicine autora da kreiraju nove metodike kojima bi se pokrili spomenuti nedostaci u primjeni u domeni mobilnih aplikacija.

Osim problema specifičnih za metodike razvoja, problem rascjepkanosti mobilnih platformi i verzija zahtjeva od timova koji se bave mobilnim razvojem fokusiranje na samo pojedine platforme i njihove verzije (Shah et al., 2019), ali kako se razvojem mobilnih aplikacija prvenstveno želi uključiti veliki broj korisnika takav pristup nije najbolji te razvojni timovi posežu za različitim rješenjima koje predlažu znanstvena i stručna zajednica. Tu najprije možemo spomenuti pristup koji omogućuje razvojnim timovima korištenje posredničkog programskog jezika ili posredničkog alata za transformaciju kôda za više odredišnih platformi. Neki od utjecajnijih projekata koji koriste ovaj pristup su React Native (Facebook Inc., 2020) i Flutter (Google Inc., 2020). Iako spomenuti pristupi imaju određene prednosti, oni također imaju i značajne nedostatke, kao što su njihova ovisnost o kvaliteti samog posredničkog alata za transformaciju, korištenju specifičnih programskih sučelja u često samo specifičnoj domeni, manjak kontrole nad generiranim programskim kodom i drugi. Drugo moguće rješenje iznad navedenog problema je korištenje aplikacija za prilagodbu (eng. adapters) kao prirodnih aplikacija za svaku ciljanu platformu. Prema Agarwalu i suradnicima (2009) ovo je jedna od dvije najvažnije tehnike upravljanja problemom rascjepkanosti. Budući da standardizacija aplikativnih programskih sučelja u mobilnom svijetu još nije moguća, korištenje programskih tehnika pomoću kojih su pozivi prema mobilnim sučeljima apstrahirani i omotani u specifične module koji se potom prilagođavaju svakoj pojedinoj platformi ostaje kao druga moguća opcija. Predstavnici ovog modela su MobiVine (Agarwal et al., 2009), Adobe PhoneGap (PhoneGap, 2020) ili Adobe AIR® (Adobe Inc., 2020). Međutim, skoro svi nedostaci koji se odnose na rješenja pomoću alata za transformaciju su također prisutni u ovom rješenju. Na kraju, treći pristup pretpostavlja korištenje mobilnih tehnologija i razvoj

više-platformskih aplikacija za web, ali ovo rješenje nije u našem fokusu jer se bitno razlikuje od temeljnih pretpostavki od kojih polazi naše istraživanje, te također ima i svoje brojne nedostatke.

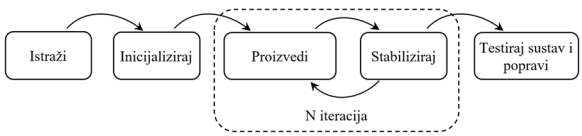
Stoga, u ovom istraživanju se fokusiramo na pronalazak rješenja koje će poboljšati metodičku interoperabilnost unutar tima ili između timova koji rade na istoj aplikaciji ali za različita (ali urođena) razvojna okruženja. U kontekstu šireg istraživanja, u ovom radu želimo identificirati koji artefakti (obvezni ulazi ili izlazi iz metodički definiranih koraka u procesu razvoja) nastaju tijekom procesa razvoja, te postoje li i u kojoj mjeri sličnosti između artefakata kako bi ih se moglo ponovno iskoristiti tijekom procesa razvoja za drugu ili svaku sljedeću ciljanu platformu. Time bi smo kreirali osnovu za efikasniji i interoperabilan proces razvoja mobilnih aplikacija za više odredišnih platformi.

Rad je strukturiran na sljedeći način: U drugom poglavlju predstavljamo kontekst metodički vođenog procesa razvoja u kojem se provodi analiza artefakata dok u trećem poglavlju prezentiramo identificirane artefakte i njihovu klasifikaciju po tipovima. Četvrto poglavlje prikazuje rezultate analize ponovne iskoristivosti, dok u posljednjem poglavlju zaključujemo temu te usmjeravamo pogled prema budućim istraživanjima.

# 2 Priprema analize

U našem prethodnom istraživanju (Stapic et al., 2016), proveli smo proces sistematskog pregleda literature kako bi smo identificirali novo-kreirane metodike razvoja koje su dizajnirane za razvoj mobilnih aplikacija, te smo otkrili 14 takvih metodika. Međutim, postoje izvješća da je samo jedna od spomenutih metodika zapravo korištena na projektima u praksi – Mobile-D.

Kako je opisano u (Abrahamsson et al., 2004; Supan et al., 2013; VTT Technical Research Centre of Finland, 2006), proces Mobile-D uključuje pet faza koje se izvršavaju iterativno ali djelomično inkrementalno (vidi sliku 1). Tijekom prve faze – Istraži – se priprema osnova budućeg razvoja. Faza Inicijalizacije bi trebala opisati i pripremiti sve aplikacijske komponente kao i predvidjeti sve moguće kritične segmente projekta.




**Slika 1.** Proces razvoja po metodici Mobile-D (VTT Technical Research Centre of Finland, 2006)

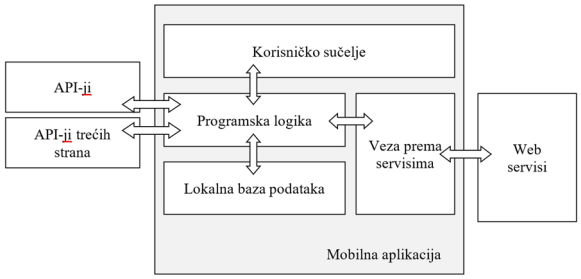
Faze Proizvedi i Stabiliziraj se izvršavaju inkrementalno kako bi se razvilo sve funkcionalnosti mobilnog proizvoda. Svaka iteracija u fazi Proizvedi započinje Danom planiranja, a prva aktivnost je retrospektivna radionica koja temeljem spoznaja iz prethodne iteracije ima za cilj poboljšati proces razvoja i uskladiti ga sa trenutnim potrebama razvojnog tima. Slijede zadaci izrade analize zahtjeva, plana iteracije, testova prihvatljivosti, a svi se izvode tijekom Dana planiranja. Radni dan se temelji na provedbi procesa razvoja temeljenog na testiranju, programiranju u paru, kontinuiranoj integraciji i refaktoriranju. Na kraju ovoga Dana slijedi zadatak obavještavanja naručitelja o implementiranim funkcionalnostima. Potom, Dan objave uključuje aktivnosti integracije i testiranja. Faza Stabiliziraj ima cilj završiti implementaciju uz integraciju svih podsustava ako je to potrebno. Kako i ova faza može sadržavati dodatno programiranje i razvoj aktivnosti su slične aktivnostima u fazi Proizvedi. Jedina dodatna aktivnost odnosi se na završavanje dokumentacije. Svaka iteracija bi trebala završiti sa novom gotovom funkcionalnošću isporučenom korisniku.

Posljednja, faza testiranja sustava i popravaka služi kako bi se utvrdilo zadovoljava li razvijeni proizvod korisničke zahtjeve. Također, projektnom timu daje povratnu informaciju o funkcionalnosti cijelog sustava te informacije o nedostacima kako bi se u posljednjoj iteraciji procesa ti nedostaci uklonili. Spomenuta posljednja iteracija nije obvezna, ali ako su ispravci potrebni, iteracija se sastoji od istih aktivnosti kao i prethodno spomenute implementacijske iteracije (Abrahamsson et al., 2004; Supan et al., 2013; VTT Technical Research Centre of Finland, 2006).

Mobile-D preporuča korištenje razvoja vođenog testiranjem (eng. TDD), koji je povezan sa svim fazama u Mobile-D. Osnove, kao i napredni koncepti razvoja vođenog testiranjem, mogu biti pronađeni u (Hammond & Umphress, 2012). Svrha ovog pristupa je dati razvojnim inženjerima pouzdanje da je programski kôd koji razvijaju ispravan, ali i da usmjerava dizajn programskog kôda prema strukturi koju je jednostavno testirati. Dodatno, prakse refaktoriranja su također temeljene na TDD-u kako bi se osiguralo da promjene koje nastaju na kodu ne narušavaju niti jednu od postojećih ili novih funkcionalnosti (Abrahamsson et al., 2005).

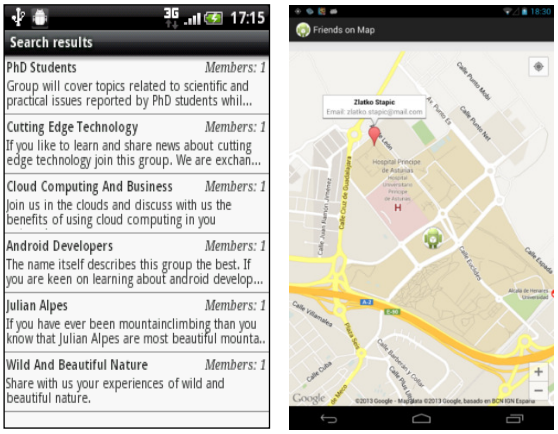
Kako bi se sistematski promotrio ovaj složeni razvojni proces, te kako bi identificirali artefakti u njemu kreirani, za Android platformu je razvijena prototipna mobilne aplikacija, imenom KnowLedge. Namjena aplikacije je omogućit korisnicima učenje i dijeljenje znanja na interaktivan način i pomoću društvenih mreža. Između ostalog, zahtijevane su funkcionalnosti kao što su pretraživanje kategorija kako bi se pronašlo postojeće znanje na određenu temu, ili pak postavljanje pitanja za novim pojašnjenjima ili uputama, ili jednostavno dijeljenja znanja u određenu grupu i tako dalje.

Arhitektura cijelog sustava uključuje servisno orijentiranu arhitekturu koja povezuje mobilnu aplikaciju, udaljenu bazu te korištenje geolokacijskog sustava. Dodatno, kako je prikazano na slici 2, arhitektura mobilne aplikacije je građena s namjenom da bude više-slojna s tri odvojena ali povezana sloja. Unutarnja kohezija (vidi (Miller, 2008)) predloženog modula bi trebala biti visoka, ali u isto vrijeme i vanjska ovisnost bi trebala biti što manja.




**Slika 2.** Arhitektura mobilne aplikacije KnowLedge

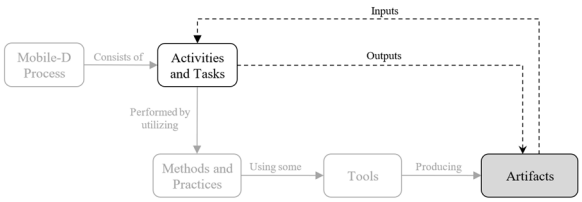
Cjelokupni proces Mobile-D s pripadajućom jasnom tehničkom specifikacijom je izvrsno dokumentiram i jednostavan za korištenje. Nekoliko slika zaslona kreirane aplikacije su prikazani na slici ispod (slika 3). Aplikacija je razvijena na engleskom jeziku.




**Slika 3.** Primjeri slika ekrana mobilne aplikacije KnowLedge

# 3 Identificiranje artefakata

Kako postoji više različitih definicija artefakta (npr. od Hilpinena (2011) ili od Parkera (2011)), mi smo preuzeli definiciju od Conradija (2004) koji kaže da je artefakt " *bilo koji dio softvera (to jest model/opis/kôd) kreiran i korišten tijekom razvoja i održavanja računalnog programa* ". Kako je cilj ovog istraživanja bio analizirati strukturalne i semantičke aspekte niza artefakata, proveli smo analizu samo promatrajući semantičke koncepte, dok drugi pristupi, kao promatranje proceduralnih koncepata ili pragmatičnih koncepata nisu uključeni. Stoga, samo smo promatrali artefakte i njihove veze prema aktivnostima i zadacima, kako je prikazano na slici 4.




**Slika 4.** Usmjerenje na izvor i semantiku artefakta

Analiza artefakata je provedena u dva koraka. Prvo, analizirali smo Mobile-D procesnu biblioteku (Abrahamsson et al., 2005) i identificirali dokumente i druge isporuke koje su neovisne o platformi, a definirani su na visokoj razini apstrakcije. Drugo, budući da pristup u identificiranju i grupiranju artefakata isključivo temeljem faze izvornog nastanka nije dobar, te budući da smo tijekom implementacije prikupili dodatne informacije o artefaktima, sistematizirali smo i opisali sve identificirane artefakte koristeći predložak kao u tablici 1.

Stoga, konceptualno, kreirali smo temelje identificiranja ne samo kreiranih dokumenata već i onih artefakata koje bi bilo teško identificirati da je projekt realiziran izvan laboratorija.

Tablica 1 prikazuje isječak iz popisa identificiranih artefakata, uključujući i inicijalnu klasifikaciju, opis i vezu prema fazama Mobile-D procesa. Koristili smo klasičnu CRU notaciju kako bi smo označili artefakte koji su bili kreirani (C), korišteni/čitani (R) i ažurirani (U).

**Tablica 1.** Isječak iz popisa identificiranih artefakata u procesu razvoja za Android

| **Naziv artefakta** | **Tip** | **Opis** | **Ulazi i izlazi po fazama** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | **I** | | **II** | | **III** | | **IV** | | **V** | |
| | | | **Ulaz** | **Izlaz** | **Ulaz** | **Izlaz** | **Ulaz** | **Izlaz** | **Ulaz** | **Izlaz** | **Ulaz** | **Izlaz** |
| Biblioteka Mobile-D procesa | Dokument | Procesna biblioteka koja detaljno opisuje Mobile-D metodiku. Korištena je kao vodič za implementaciju svake faze. (Abrahamsson et al., 2005) | R | | R | | R | | R | | R | |

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Prijedlog proizvoda | Dokument | Kreiran prije procesa razvoja. Opisuje inicijalnu ideju i osnovne funkcionalnosti proizvoda. | R | | | | | | | | | | |
| Projektni plan | Dokument | Sadrži sve informacije o projektu uključujući podatke o korisnicima, domenu projekta, planirane aktivnosti i njihovo trajanje, planove dokumentacije i slično. U skladu je s agilnom praksom, te je ažuriran tijekom iteracija. | | C | R | U | R | U | | | | | |
| … | … | … | | | | | | | | | | | |

Proces identificiranja rezultirao je s ukupno 60 različitih artefakata nastalih pri razvoju za Android, a koji su potom analizirani kako bi se utvrdilo koji od ovih artefakata su potencijalno ponovno iskoristivi pri razvoju za druge platforme kao što je iOS. Artefakti su grupirani u 12 kategorija prikazanih u tablici ispod. Cjelokupan popis je dostupan u prilogu 1.

**Tablica 2.** Tipovi artefakata u razvoju za Android

| **Tip artefakta** | **Opis** |
|---|---|
| Dokument | Predstavlja korištenu dokumentaciju ili kreirane artefakti koji su objavljeni kao dokumenti tijekom ili na kraju procesa razvoja. |
| Ugrađeni dokument | Predstavlja dokument koji može biti promatran kao samostalni artefakt, ali je uobičajeno uključen u neki drugi dokument. |
| Predložak | Predstavlja predloške koji se koriste pri kreiranju određenih artefakata. |
| Model | Predstavlja modele koji su kreirani pri procesu razvoja. Modele možemo promatrati kao samostalne artefakte, ali ih se uobičajeno prezentira kao dio nekog dokumenta. |
| Dio modela | Predstavlja najmanji unutarnji dio artefakta modela koji se može promatrati kao samostalan ali je korišten u izradi složenijih modela. |
| Kôd | Predstavlja bilo koji artefakt kreiran tijekom implementacije koji je napisan u bilo kojem programskom ili opisnom jeziku. |
| Primjer treće strane | Predstavlja dijelove programskog koda kreirane od trećih strana koji se koriste kao primjeri implementirane funkcionalnosti ili kako bi riješili neki programerski problem. |
| Softverski alat | Predstavlja softverski alat koji se koristi u bilo kojoj fazi procesa. |
| Licenca | Predstavlja jedinstveni ključ namijenjen jednoj osobi koji je dohvaćen ili korišten tijekom procesa razvoja. |
| Standard | Predstavlja dokument koji sadrži formalan i međunarodno priznat opis nekog koncepta ili elementa. |
| Resurs za objavu | Predstavlja resurse kreirane tijekom procesa razvoja, a koji su korišteni u svrhu objave proizvoda u dućan. |
| Proizvod | Predstavlja konačan proizvod kao najvažniji rezultat projekta. |

Budući da je prototipna aplikacija dizajnirana da uključi najvažnije i najčešće izazove pri razvoju mobilnih aplikacija (vidi sliku 4), ne očekujemo da će broj artefakata značajno rasti ako bi broj funkcionalnosti aplikacije bio veći. Prototipna aplikacija KnowLedge, zajedno sa pripadajućim testovima i pozadinskih servisima ima 12.000 linija kôda (LOC).

Pojedini dokumenti sadrže dijelove (ugrađene dokumente) koje bi trebalo promatrati samostalno, te smo ih stoga klasificirali u poseban (novi) tip. Slično, dio modela se može promatrati kao samostalni artefakt koji se koristi za izgradnju složenijih modela.

# 4 Analiza ponovne iskoristivosti

Mijač (2015) definira ponovnu iskoristivost kao mogućnost bilo kojeg dijela proizvoda da u određenoj mjeri bude ponovno korišten, pri čemu je ponovno korištenje definirano kao korištenje postojećeg softvera ili ugrađenog znanja pri izgradnji novih proizvoda. U kontekstu analize koja slijedi, mi smo prihvatili i djelomično prilagodili spomenute definicije kako bi ih primijenili na artefakte.

Analizom ponovne iskoristivosti za više-platformski razvoj utvrđeno je da je 50 artefakata (71.43% od svih artefakata identificiranih za Android) zajedničko u oba razvojna procesa. Dodatno, mnogi od spomenutih zajedničkih artefakata su neovisni o platformi kao rezultati provedbe metodički vođenog procesa razvoja. Ukupno 20 od 50 identificiranih zajedničkih artefakata (40%) bi trebalo kreirati ili dohvatiti samo jedanput, budući da su oni identični u oba analizirana procesa razvoja. S druge strane, tu je i 13 artefakata (26%) koji se mogu samo djelomično iskoristiti pri provedbi razvojnog procesa za drugu ili bilo koju sljedeću ciljanu platformu. Konačno, prepoznali smo 17 artefakata (34% svih zajedničkih artefakata) s neznatno malom mogućnošću ponovne iskoristivosti. Njih smo klasificirali kao one koje treba razviti iz početka za svaku ciljanu platformu. Isječak rezultata više-platformske analize ponovne iskoristivosti može se vidjeti u tablici 3. Svi ostali artefakti, iako bi mogli imati dijelove semantike ili sintakse kao ponovno iskoristive (npr. slijed, iteracije, algoritme…), klasificirani su kao artefakti ovisni o platformi. Cjelokupan popis zajedničkih artefakata i pripadajuća klasifikacija su dostupni u 2. dodatku ovome radu.

**Tablica 3.** Isječak iz popisa artefakata zajedničkih za platforme Android i iOS

| Naziv artefakta | Isti | Djelomično iskoristivi | Različiti |
|---|---|---|---|
| Biblioteka Mobile-D procesa | X | | |
| Prijedlog proizvoda | X | | |
| Dokument inicijalnih zahtjeva | X | | |
| Projektni plan | | X | |
| Lista provjere projektnog plana | | X | |
| Predložak liste provjere projektnog plana | X | | |
| Plan mjerenja | | X | |
| Opis arhitekture sustava | | | X |
| … | | | |

Ukupno, 33 artefakta (66% zajedničkih artefakata), su djelomično ili potpuno ponovno iskoristivi što nas je ohrabrilo i dalo nam motivaciju za sljedeće faze istraživanja prema metodičkoj interoperabilnosti i ponovnoj iskoristivosti artefakata.

# 5 Zaključak

U ovom istraživanju prikazani su rezultati analize ponovne iskoristivosti procesno orijentiranih (metodičkih) i razvojnih artefakata koji nastaju pri razvoju mobilnih aplikacija za dvije ili više ciljanih platformi. Najprije je analizirana biblioteka procesa razvoja po metodici Mobile-D, te su identificirani dokumenti i drugi izlazni artefakti na najvišoj razini apstrakcije. Nadalje, prikupili smo dodatne podatke o artefaktima i identificirali 60 različitih artefakata koji nastaju pri procesu razvoja za Android i koji su grupirani u 12 različitih kategorija. U analizi ponovne iskoristivosti pri više-platformskom razvoju, utvrđeno je da je 71.43% svih artefakata zajedničko i prisutno pri razvoju za dvije ili više platformi. 66% spomenutih zajedničkih artefakata su potpuno ili djelomično ponovno iskoristivi, što potvrđuje da ponovna iskoristivost artefakata predstavlja čvrstu osnovu za unaprjeđenje procesa više-platformskog razvoja mobilnih aplikacija.

Kroz buduća istraživanja željeli bi smo semantički opisati metodički vođen proces razvoja kao i artefakte koji u njemu nastaju, s posebnim naglaskom na ponovno iskoristive artefakte, kako bi kreirali novi metodički okvir za razvoj više-platformskih prirodnih mobilnih aplikacija.

# Zahvala

Autor želi zahvaliti recenzentima na cijenjenim komentarima i prijedlozima, čija implementacija je značajno poboljšala ovaj rad.

# Reference

Abrahamsson, P., Hanhineva, A., Hulkko, H., Ihme, T., Jäälinoja, J., Korkala, M., Koskela, J., Kyllönen, P., & Salo, O. (2004). Mobile-D: An agile approach for mobile application development. Companion to the 19th Annual ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages, and Applications - OOPSLA '04, 174. https://doi.org/10.1145/1028664.1028736

Abrahamsson, P., Hanhineva, A., Hulkko, H., Jäälinoja, J., Komulainen, K., Korkala, M., Koskela, J., Kyllönen, P., & Eporwei, O. T. (2005). Agile Development of Embedded Systems: Mobile-D (Agile Deliverable D.2.3; p. 203). ITEA. http://www.agile-itea.org/public/deliverables/ITEA-AGILE-D2.3_v1.0.pdf

Adobe Corporation. (2011). Adobe Announces Agreement to Acquire Nitobi, Creator of PhoneGap. Adobe.Com - Press Releases. http://www.adobe.com/aboutadobe/pressroom/pressreleases/201110/AdobeAcquiresNitobi.html

Agarwal, V., Goyal, S., Mittal, S., & Mukherjea, S. (2009). MobiVine: A middleware layer to handle fragmentation of platform interfaces for mobile applications. Proceedings of the 10th ACM/IFIP/USENIX International Conference on Middleware, 24:1–24:10. http://dl.acm.org/citation.cfm?id=1656980.1657013

Amanquah, N., & Eporwei, O. T. (2009). Rapid application development for mobile terminals. 2nd International Conference on Adaptive Science & Technology (ICAST), 410–417. https://doi.org/10.1109/ICASTECH.2009.5409691

Conradi, R. (2004). Software engineering mini glossary. http://www.idi.ntnu.no/grupper/su/publ/ese/se-defs.html

Hammond, S., & Umphress, D. (2012). Test driven development. 158. https://doi.org/10.1145/2184512.2184550

Hilpinen, R. (2011). Artifact. Stanford Encyclopedia of Philosophy. http://plato.stanford.edu/entries/artifact/

Hosbond, J. H. (2005). Mobile Systems Development: Challenges, Implications and Issues. In J. Krogstie, K. Kautz, & D. Allen (Eds.), Mobile Information Systems II (Vol. 191, pp. 279–286). Springer Boston. http://dx.doi.org/10.1007/0-387-31166-1_20

La, H. J., & Kim, S. D. (2009). A service-based approach to developing Android Mobile Internet Device (MID) applications. 2009 IEEE International Conference on Service-Oriented Computing and Applications (SOCA), 00(MID), 1–7. https://doi.org/10.1109/SOCA.2009.5410278

Manjunatha, A., Ranabahu, A., Sheth, A., & Thirunarayan, K. (2010). Power of Clouds in Your Pocket: An Efficient Approach for Cloud Mobile Hybrid Application Development. 2010 IEEE Second International Conference on Cloud Computing Technology and Science, 496–503. https://doi.org/10.1109/CloudCom.2010.78

Mijač, M., & Stapić, Z. (2015). Reusability Metrics of Software Components: Survey. https://doi.org/10.13140/RG.2.1.3611.4642

Miller, J. (2008). Cohesion And Coupling. MSDN Magazine - The Microsoft Journal for Developers, 23(10). http://msdn.microsoft.com/en-us/magazine/cc947917.aspx

Parker, P. M. (2011). Definition of artifact. Webster's Online Dictionary. http://www.websters-online-dictionary.org/definitions/artifact

PhoneGap. (2011). Take the pain out of compiling mobile apps for multiple platforms. PhoneGap Build. https://build.phonegap.com/

Rahimian, V., & Ramsin, R. (2008). Designing an agile methodology for mobile software development: A hybrid method engineering approach. Proceedings of Second International Conference on Research Challenges in Information Science, RCIS (2008), 337–342. https://doi.org/10.1109/RCIS.2008.4632123

Rhomobile, Inc. (2011). Smartphone Enterprise Application Integration, White paper. http://tiny.cc/rhomobile

Shah, K., Sinha, H., & Mishra, P. (2019). Analysis of Cross-Platform Mobile App Development Tools. 2019 IEEE 5th International Conference for Convergence in Technology (I2CT), 1–7. https://doi.org/10.1109/I2CT45611.2019.9033872

Spataru, A. C. (2010). Agile Development Methods for Mobile Applications [PhD Thesis, University of Edinburgh, The University of Edinburgh]. http://www.inf.ed.ac.uk/publications/thesis/online/IM100767.pdf

Stapic, Z., Mijac, M., & Strahonja, V. (2016). Methodologies for development of mobile applications. 2016 39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), 688–692. https://doi.org/10.1109/MIPRO.2016.7522228

Supan, D., Teković, K., Škalec, J., & Stapić, Z. (2013). Using Mobile-D methodology in development of mobile applications: Challenges and issues. Razvoj Poslovnih i Informatičkih Sustava CASE 25, 91–98.

VTT Technical Research Centre of Finland. (2006). Mobile-D Online Presentation (Web Application). AGILE Software Technologies Research Programme. http://agile.vtt.fi/mobiled.html

# Dodaci

1. Identificirani artefakti pri provedbi procesa razvoja za Android, dostupno na: http://tiny.cc/identified_artifacts
2. Zajednički artefakti pri provedbi procesa razvoja za različite platforme i njihova klasifikacija, dostupno na: http://tiny.cc/common_artifacts