2021 ‹ ŠTEVILKA 3 ‹ LETNIK XXIX ‹ ISSN 1318-1882

03 UPORABNA INFORMATIKA

U P O R A B N A
I N F O R M A T I K A
2021 ŠTEVILKA 3 JUL/AVG/SEP LETNIK XXIX ISSN 1318-1882

Znanstveni prispevki

Kratki znanstveni prispevki

Informacije

U P O R A B N A
I N F O R M A T I K A

2021 ŠTEVILKA 3 JUL/AVG/SEP LETNIK XXIX ISSN 1318-1882

Ustanovitelj in izdajatelj
Slovensko društvo INFORMATIKA
Litostrojska cesta 54, 1000 Ljubljana

Predstavnik
Niko Schlamberger

Odgovorni urednik
Saša Divjak

Tehnični urednik
Slavko Žitnik

Lektoriranje angleških izvlečkov
Marvelingua (angl.)

Oblikovanje
KOFEIN DIZAJN, d. o. o.

Prelom in tisk
Boex DTP, d. o. o., Ljubljana

Naklada
200 izvodov

Naslov uredništva
Slovensko društvo INFORMATIKA
Uredništvo revije Uporabna informatika
Litostrojska cesta 54, 1000 Ljubljana
www.uporabna-informatika.si

Revija izhaja četrtletno. Cena posamezne številke je 20,00 EUR. Letna naročnina za podjetja 85,00 EUR, za vsak nadaljnji izvod 60,00 EUR, za posameznike 35,00 EUR, za študente in seniorje 15,00 EUR. V ceno je vključen DDV.

Revija Uporabna informatika je od številke 4/VII vključena v mednarodno bazo INSPEC.

Revija Uporabna informatika je pod zaporedno številko 666 vpisana v razvid medijev, ki ga vodi Ministrstvo za kulturo RS.

Revija Uporabna informatika je vključena v Digitalno knjižnico Slovenije (dLib.si).

Vabilo avtorjem

V reviji Uporabna informatika objavljamo kakovostne izvirne članke domačih in tujih avtorjev z najširšega področja informatike v poslovanju podjetij, javni upravi in zasebnem življenju na znanstveni, strokovni in informativni ravni; še posebno spodbujamo objavo interdisciplinarnih člankov. Zato vabimo avtorje, da prispevke, ki ustrezajo omenjenim usmeritvam, pošljejo uredništvu revije po elektronski pošti na naslov ui@drustvo-informatika.si.

Avtorje prosimo, da pri pripravi prispevka upoštevajo navodila, objavljena v nadaljevanju ter na naslovu http://www.uporabna-informatika.si.

Za kakovost prispevkov skrbi mednarodni uredniški odbor. Članki so anonimno recenzirani, o objavi pa na podlagi recenzij samostojno odloča uredniški odbor. Recenzenti lahko zahtevajo, da avtorji besedilo spremenijo v skladu s priporočili in da popravljeni članek ponovno prejmejo v pregled. Uredništvo pa lahko še pred recenzijo zavrne objavo prispevka, če njegova vsebina ne ustreza vsebinski usmeritvi revije ali če članek ne ustreza kriterijem za objavo v reviji.

Pred objavo članka mora avtor podpisati izjavo o avtorstvu, s katero potrjuje originalnost članka in dovoljuje prenos materialnih avtorskih pravic. Nenaročenih prispevkov ne vračamo in ne honoriramo. Avtorji prejmejo enoletno naročnino na revijo Uporabna informatika, ki vključuje avtorski izvod revije in še nadaljnje tri zaporedne številke.

S svojim prispevkom v reviji Uporabna informatika boste prispevali k širjenju znanja na področju informatike. Želimo si čim več prispevkov z raznoliko in zanimivo tematiko in se jih že vnaprej veselimo.

Uredništvo revije

Navodila avtorjem člankov

Članke objavljamo praviloma v slovenščini, članke tujih avtorjev pa v angleščini. Besedilo naj bo jezikovno skrbno pripravljeno. Priporočamo zmernost pri uporabi tujk in – kjer je mogoče – njihovo zamenjavo s slovenskimi izrazi. V pomoč pri iskanju slovenskih ustreznic priporočamo uporabo spletnega terminološkega slovarja Slovenskega društva Informatika Islovar (www.islovar.org).

Znanstveni članek naj obsega največ 40.000 znakov, strokovni članki do 30.000 znakov, obvestila in poročila pa do 8.000 znakov.

Članek naj bo praviloma predložen v urejevalniku besedil Word (*.doc ali *.docx) v enojnem razmaku, brez posebnih znakov ali poudarjenih črk. Za ločilom na koncu stavka napravite samo en prazen prostor, pri odstavkih ne uporabljajte zamika.

Naslovu članka naj sledi za vsakega avtorja polno ime, ustanova, v kateri je zaposlen, naslov in elektronski naslov. Sledi naj povzetek v slovenščini v obsegu 8 do 10 vrstic in seznam od 5 do 8 ključnih besed, ki najbolje opredeljujejo vsebinski okvir članka. Pred povzetkom v angleščini naj bo še angleški prevod naslova, prav tako pa naj bodo dodane ključne besede v angleščini. Obratno velja v primeru predložitve članka v angleščini.

Razdelki naj bodo naslovljeni in oštevilčeni z arabskimi številkami.

Slike in tabele vključite v besedilo. Opremite jih z naslovom in oštevilčite z arabskimi številkami. Vsako sliko in tabelo razložite tudi v besedilu članka. Če v članku uporabljate slike ali tabele drugih avtorjev, navedite vir pod sliko oz. tabelo. Revijo tiskamo v črno-beli tehniki, zato barvne slike ali fotografije kot original niso primerne. Slik zaslonov ne objavljamo, razen če so nujno potrebne za razumevanje besedila. Slike, grafikoni, organizacijske sheme ipd. naj imajo belo podlago. Enačbe oštevilčite v oklepajih desno od enačbe.

V besedilu se sklicujte na navedeno literaturo skladno s pravili sistema APA navajanja bibliografskih referenc, najpogosteje torej v obliki (Novak & Kovač, 2008, str. 235). Na koncu članka navedite samo v članku uporabljeno literaturo in vire v enotnem seznamu po abecednem redu avtorjev, prav tako v skladu s pravili APA. Več o sistemu APA, katerega uporabo omogoča tudi urejevalnik besedil Word 2007, najdete na strani http://owl.english.purdue.edu/owl/resource/560/01/.

Članku dodajte kratek življenjepis vsakega avtorja v obsegu do 8 vrstic, v katerem poudarite predvsem strokovne dosežke.

Pomen uporabe arhitekturnih načrtovalskih vzorcev pri razvoju mobilnih aplikacij

Luka Pavlič, Luka Četina
Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, Koroška cesta 46, 2000 Maribor
luka.pavlic@um.si, luka.cetina1@um.si

Izvleček

Izbira ustrezne arhitekture je pomemben in nujen korak razvoja mobilnih aplikacij za operacijski sistem Android. V članku predstavljamo sistematično primerjavo najbolj priljubljenih arhitekturnih načrtovalskih vzorcev in njihovo vlogo pri načrtovanju mobilnih aplikacij. Pri raziskovanju smo se omejili na osem arhitekturnih vzorcev iz kataloga Jetpack. Ugotoviti smo želeli, v kakšni meri ti vzorci vplivajo na potek in končni izid razvoja mobilnih aplikacij. Z uporabo vsakega izmed obravnavanih vzorcev smo razvili mobilno aplikacijo za operacijski sistem Android. Zanimalo nas je, kako se aplikacije, razvite z uporabo arhitekturnih vzorcev, primerjajo z njihovimi alternativami. Razvili smo osem mobilnih aplikacij, ki izkoriščajo prednosti obravnavanih načrtovalskih vzorcev. V namen primerjave smo razvili osem dodatnih mobilnih aplikacij, pri katerih nismo uporabili ustreznih arhitekturnih vzorcev, temveč smo aplikacijo razvili brez njih. Tako razvite aplikacije smo primerjali po več kriterijih, med katerimi so bili poglavitni zahtevnost razvoja, čas razvoja ter vrednosti programskih metrik. Na podlagi tako pridobljenih podatkov smo argumentirali smiselnost uporabe vzorcev v podanih kontekstih. V članku tako pokažemo, da je razvoj mobilnih aplikacij z uporabo arhitekturnih vzorcev kataloga Jetpack ne samo manj zahteven, ampak je kljub večjemu številu komponent, obseg izvorne kode manjši. Ugotovili smo, da sta, presenetljivo, čas razvoja in notranja kakovost aplikacij kljub uporabi načrtovalskih vzorcev primerljiva z alternativnimi pristopi.

Ključne besede: Android, mobilne aplikacije, arhitekturni vzorci, načrtovanje mobilnih aplikacij, Android Jetpack

Abstract

Choosing an appropriate architecture is a crucial and necessary step in mobile application development. In this paper, we present a systematic comparison of the most popular architectural patterns and discuss their importance for mobile app development. In our research, we focused on eight architectural patterns from the Jetpack library. We wanted to investigate their impact on the course and outcomes of mobile app development. This is why we have developed an Android application using each of the eight patterns. In order to assess their role, we developed eight more mobile applications. They differ from the previous ones only in that we did not use the Jetpack architectural patterns, but instead developed the applications ad hoc. We compared the applications using several criteria, including time, effort and code quality metrics. Based on this, we argued the rationale for using the patterns in the given contexts. In this paper, we demonstrate that the mobile app development using Jetpack architectural patterns is not only less demanding, but also requires fewer lines of code despite the higher component number. We show how, despite the use of architectural patterns, product development time and internal quality of applications was comparable to those developed with alternative approaches.

Keywords: Android, architectural patterns, mobile application design, Android Jetpack

1 UVOD

Za preživetje v visoko tekmovalnem trgu aplikacij za operacijski sistem Android je ključno, da razvijalci v kratkem času dostavijo kakovostne mobilne aplikacije. Razvoj aplikacij za operacijski sistem Android v povprečju traja 20-30 % dlje in je za tretjino dražji kot razvoj aplikacij za iOS. To je med drugim posledica tega, da je mobilne aplikacije potrebno testirati na

več napravah (Ardas Group Inc., 2017). Ustrezna arhitektura je zaradi tega pri razvoju mobilnih aplikacij za operacijski sistem Android še toliko bolj pomembna. Le-ta namreč omogoča lažje vzdrževanje in testiranje, kar zmanjša skupen čas, potreben za razvoj in vzdrževanje. Kot vidimo na sliki 1, bo uporaba ustrezne arhitekture sicer potek projekta na začetku upočasnila, vendar bodo prednosti postale vidne že v nekaj tednih. Takrat bomo lahko nove funkcionalnosti dodajali bistveno hitreje, kot če ustrezne arhitekture ne bi imeli (Fowler, Software Architecture Guide, 2019). Izbira dobre zasnove je pristop, ki lahko razvijalcem že na kratek rok pomaga pri doseganju višje kakovosti mobilnih aplikacij.

Kaj sploh je dobra arhitektura mobilne aplikacije za operacijski sistem Android, je vprašanje, ki so ga skušali nasloviti že mnogi avtorji (npr. Aymen, et al., 2019, Prabowo, et al., 2018, Verdecchia, et al., 2019). Kot ugotavljajo avtorji, mnenja pogosto kroji navdušenje za trenutno popularne tehnologije in ne objektivni dokazi. Ustrezna arhitektura je odvisna od vrste in namena mobilne aplikacije. Zato ni enotnega odgovora na vprašanje »Kaj je dobra arhitektura ?«. Kljub temu pa obstajajo splošni vzorci, dobre prakse, ki se pojavijo v večini arhitektur. Potrebo po dobri arhitekturi in lažji implementaciji teh vzorcev so prepoznali tudi pri podjetju Google in izdali katalog z imenom Android Jetpack (Android Developers, 2020). Katalog se deli na 4 kategorije: podporne knjižnice (ang. *foundation)*, arhitekturne komponente, vedenjske komponente (ang. *behaviour*) in komponente za gradnjo uporabniškega vmesnika (ang. *user interface*). V tej raziskavi smo se osredotočili zgolj na arhitekturni del, ki vsebuje osem arhitekturnih vzorcev.

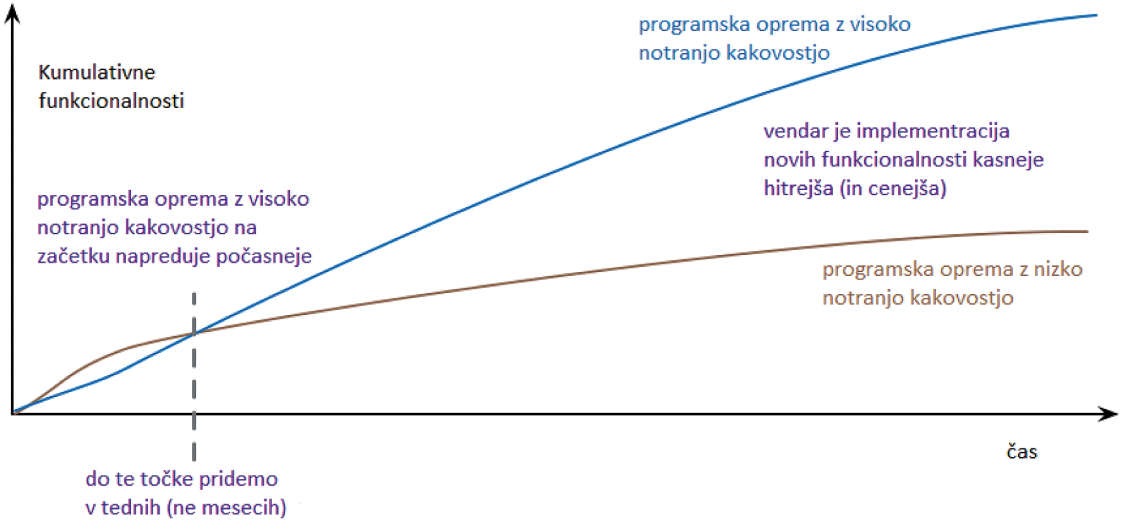
Raziskovalni vprašanji, na kateri smo odgovarjali, sta:

1. Katere so prednosti uporabe arhitekturnih vzorcev Jetpack pri razvoju mobilnih aplikacij?
   a. krajši čas razvoja,
   b. manjša zahtevnost razvoja,
   c. velikost izvorne kode izdelka,
   d. izboljšana notranja kakovost.
2. Ali, kdaj in v kolikšni meri je arhitekturne vzorce kataloga Jetpack smiselno uporabiti?

V nadaljevanju članka bodo v 2. poglavju predstavljena sorodna dela in arhitekturni vzorci podrobneje. 3. poglavje povzema metode raziskovanja. Podrobneje bomo predstavili raziskovalna vprašanja, postopek raziskovanja in način izvedbe meritev. 4. poglavje vsebuje predstavitev rezultatov, ki jih bomo v 5. poglavju podrobneje analizirali. V zadnjem poglavju na kratko povzemamo bistvo in rezultate članka.

2 SORODNA DELA

Konec 70. let se je ob nastanku grafičnih uporabniških vmesnikov pojavil arhitekturni načrtovalski vzorec MVC (ang. *Model-View-Controller*). Le-ta programsko logiko razdeli na model (ang. *model*), po-



Slika 1: **Čas implementacije funkcionalnosti pri programski opremi s kakovostno in nekakovostno zasnovo (Fowler, Software Architecture Guide, 2019)**

gled (ang. *view*) in krmilnik (ang. *controller*). Njegova glavna naloga je ločitev predstavitve podatkov od poslovne logike (Reenskaug, 2003). Iz te ideje se je razvil vzorec MVP (ang. *Model-View-Presenter*), ki je idejo ločitve logike od predstavitve podatkov popeljal še dlje in uvedel bolj pasivni pogled, ki ne vsebuje predstavitvene logike (Potel, 1996). Vzorec MVVM (ang. *Model-View-ViewModel*) je še ena variacija vzorca MVC, ki se ukvarja s problematiko ločitve poslovne logike in predstavitve podatkov. Osnova zanj je ideja Martina Fowlerja (Fowler, Presentation model, 2004), s pomočjo katere so inženirji pri Microsoftu ustvarili konkretno arhitekturo (Gossman, 2005). Razvijalci mobilnih aplikacij se pri načrtovanju arhitekture pogosto odločajo tudi za t.i. pristop *Clean Architecture*, ki ni le vzorec, ampak celostna načrtovalska filozofija. Njen glavni cilj je ločiti komponente tako, da odvisnosti prihajajo le od zunanjih proti notranjim slojem (Martin, 2012).

Avtorji (Verdecchia, et al., 2019) so v svojem delu raziskali načine, na katere razvijalci mobilnih aplikacij za operacijski sistem Android načrtujejo arhitekturo aplikacij. Zanimalo jih je predvsem, katerih vzorcev in dobrih praks se razvijalci najbolj pogosto poslužujejo in kakšen vpliv imajo le-te na kakovost. V sklopu dela so opravili vodene razgovore z razvijalci ter preučili obstoječo literaturo. Pri rezultatih so predstavili najbolj uporabljene arhitekturne vzorce in knjižnice za snovanje arhitekture mobilnih aplikacij. Sestavili so tudi seznam zahtev kakovosti, ki se razvijalcem zdijo najpomembnejše pri razvoju mobilnih aplikacij. Ugotovili so, da razvijalci najpogosteje uporabljajo arhitekturni vzorec MVP, najpogosteje uporabljeni katalogi pa so RxJava, Dagger in Jetpack. Rezultati so pokazali tudi, da je izmed množice atributov kakovosti razvijalcem najbolj pomembna ravno vzdrževalnost, sledita ji možnost testiranja in učinkovitost delovanja mobilne aplikacije.

Avtor (Lou, 2016) je v svojem delu preverjal, ali sta vzorca MVVM in MVP res boljša od tradicionalnega pristopa. Primerjavo vzorcev je pripravil na osnovi treh kriterijev: možnost testiranja, možnost spreminjanja in učinkovitost delovanja. Ugotovil je, da sta vzorca MVVM in MVP od MVC boljša po vseh kriterijih, med seboj pa sta si preveč podobna, da bi lahko enega označil za boljšega. Kot poglavitno razliko med vzorcema je izpostavil to, da nudi MVP boljšo možnost spreminjanja, MVVM pa boljšo možnost testiranja.

Avtorji (Aymen, et al., 2019) so se v svojem delu osredotočili na MVC arhitekturne vzorce mobilnih aplikacij za operacijski sistem Android. Želeli so ugotoviti, katere arhitekturne vzorce je priporočljivo uporabljati, ter kakšni so trendi na področju načrtovanja arhitekture mobilnih aplikacij za operacijski sistem Android. Zanimalo jih je tudi, kako pogosto mobilne aplikacije uporabljajo arhitekturne vzorce na osnovi vzorca MVC in katere vrste mobilnih aplikacij se takšnih vzorcev poslužujejo najbolj pogosto. V svojem delu so šli korak dlje od prejšnjega avtorja (Lou, 2016), saj so želeli avtomatizirati prepoznavo arhitekture, ki jo mobilna aplikacija uporablja. S pristopom RI-MAZ, ki z uporabo hevristik identificira dominanten arhitekturni vzorec mobilne aplikacije, so preučili 5480 aplikacij, dostopnih na tržnici Google Play. Ugotovili so, da je najbolj dominanten vzorec MVC, redkeje je uporabljen vzorec MVP, vzorec MVVM pa je v vzorcu aplikacij skoraj neuporabljen. Rezultati so pokazali tudi, da veliko število aplikacij (predvsem manjših) ne sledi nobenemu arhitekturnem vzorcu.

Avtorji (Prabowo, et al., 2018) so se v svojem delu prav tako osredotočili na arhitekturne vzorce, ki so osnovani na vzorcu MVC. Zanimalo jih je, kako se mobilne aplikacije, ki uporabljajo arhitekturne vzorce, primerjajo z aplikacijami, ki sledijo anti-vzorcem (slabim praksam) in posledično jasne arhitekture nimajo. Pri primerjavi mobilnih aplikacij so se osredotočili na modularnost in vzdrževalnost. Analizirali in primerjali so dve mobilni aplikaciji, od katerih ena uporablja arhitekturne vzorce, druga pa jasne arhitekture ni imela. Ugotovili so, da uporaba vzorca MVP znatno poveča modularnost mobilne aplikacije, vzdrževalnost pa je bila pri obeh mobilnih aplikacijah primerljiva.

Izbira na vzorcih temelječe arhitekture je torej pomemben korak načrtovanja mobilnih aplikacij, saj bo ta izbira vplivala na potek celotnega projekta. Uporaba ustrezne arhitekture sicer potek razvoja na začetku upočasni. Prednosti bodo postale vidne že v nekaj tednih, ko bomo lahko nove funkcionalnosti dodajali bistveno hitreje, kot če ustrezne arhitekture ne bi imeli (Fowler, Software Architecture Guide, 2019).

2.1 Arhitektura mobilnih aplikacij na primeru platforme Android

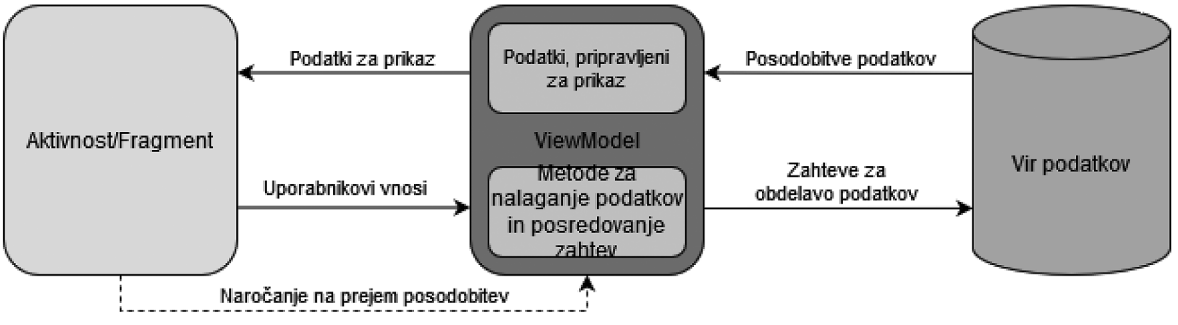
Do leta 2018 je večina mobilnih aplikacij za platformo Android uporabljala t.i. podporne knjižnice (ang. support libraries), ki so naslavljale izziv združljivosti

med napravami in različicami sistema Android. Pomembne so predvsem zaradi zagotavljanja t.i. združljivosti za naprej (ang. *forward compatibility*), kar pomeni, da bodo mobilne aplikacije, razvite za nove različice sistema Android, delovale tudi na starejših. Podporne knjižnice so se od nastanka zelo spreminjale in postajale nepregledne, zato so se pri Googlu odločili, da začnejo znova. Na podlagi prepoznanih dobrih praks so objavili katalog Jetpack, s katerim so želeli podati smernice, priporočena orodja in delno implementacijo v obliki knjižnice. Le-ta razvijalcem pomaga pri grajenju kakovostnih mobilnih aplikacij za operacijski sistem Android (Moore, 2018). Katalog se deli na 4 kategorije: podporne knjižnice (ang. *foundation)*, arhitekturne komponente, vedenjske komponente (ang. *behaviour*) in komponente za grajenje uporabniškega vmesnika (ang. *user interface*). Kljub kratkemu obstoju je katalog Jetpack hitro postal popularen, saj so ga avtorji (Verdecchia, et al., 2019) v svoji raziskavi uvrstili na 3. mesto (takoj za RxJava in Dagger) po uporabljanosti pri snovanju arhitekture mobilnih aplikacij za operacijski sistem Android. Ker se v tem članku ukvarjamo z ustrezno arhitekturo mobilnih aplikacij, se bomo osredotočili le na arhitekturni del kataloga. Ta vsebuje 8 vzorcev, ki razvijalcem pomagajo pri snovanju ustrezne arhitekture ter upravljanju in prikazovanju podatkov. To so:

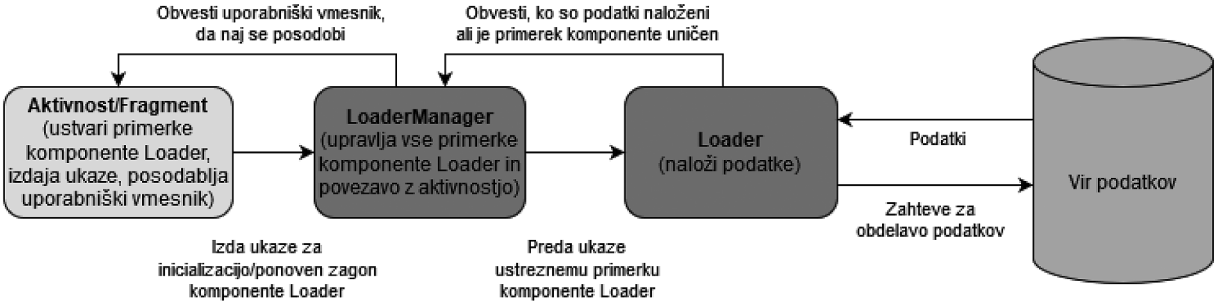
- vzorec 1: DataBinding,
- vzorec 2: Lifecycles,
- vzorec 3: LiveData,
- vzorec 4: Navigation,
- vzorec 5: Paging,
- vzorec 6: Room,
- vzorec 7: ViewModel in
- vzorec 8: WorkManager.

Za tipičnega predstavnika arhitekturnih vzorcev kataloga Jetpack lahko označimo vzorec 7 (ViewModel), ki naslavlja problematiko ločitve logike za obdelavo in prikaz podatkov (slika 2). Vzorec kataloga Jetpack je tesno povezan z arhitekturnim načrtovalskim vzorcem MVVM, saj omogoča implementacijo dela ViewModel omenjenega vzorca.

Za razumevanje implementacije vzorca ViewModel je ključno poznavanje življenjskega cikla krmilnikov uporabniškega vmesnika mobilnih aplikacij za operacijski sistem Android (aktivnosti in fragmenti). Življenjske cikle teh komponent upravlja operacijski sistem, ki lahko zaradi uporabnikovega vnosa, pomanjkanja sistemskih virov ali drugih okoliščin kadarkoli uniči ali posodobi krmilnik. S tem izbriše vse podatke, ki jih je le-ta vseboval. Za uporabnika ob neustrezni obravnavi aplikacije to pomeni nevšečnosti. Ko bo ob rotaciji zaslona aktivnost ponovno naložila začetno stanje, bodo vsi podatki, ki niso trajno shranjeni, izgubljeni. To poslabša uporabniško izkušnjo in vodi do večje porabe virov. Dodaten izziv predstavlja proženje dolgo trajajočih asinhronih klicev. Razvijalec naj bi te klice upravljal in poskrbel, da se počistijo, ko krmilnik ni več aktiven. To upravljanje zahteva ponovno veliko sistemskih virov, klici pa se pogosto ponavljajo, saj se prožijo vsakič, ko je krmilnik ponovno naložen. Vzorec ViewModel našteto problematiko reši tako, da logiko za pridobivanje podatkov loči od logike za upravljanje uporabniškega vmesnika. Slednjo umakne v razred tipa ViewModel. Objekti tega tipa se zato med spremembami konfiguracije samodejno ohranijo. Tako so podatki, ki jih vsebujejo, takoj na voljo naslednji aktivnosti ali fragmentu. S tem preprečimo izgubo podatkov, hkrati pa porabimo manj sistemskih virov, saj podatkov ni potrebno ponovno nalagati..



Slika 2: **Nalaganje podatkov z uporabo arhitekturnega vzorca ViewModel (Android Developers, 2020)**



Slika 3: **Nalaganje podatkov s pomočjo arhitekturnega vzorca Loader (Android Developers, 2020)**

Za nalaganje in posodabljanje podatkov se je tradicionalno uporabljal tudi pristop, prikazan na sliki 3. Ta definira razreda Loader, ki skrbi za pridobivanje podatkov in posredovanje podatkov ter LoadManager, ki upravlja vse primerke razreda Loader. Ko primerek razreda Loader podatke naloži, o tem obvesti razred objekt LoadManager, ta pa aktivnosti oz. fragmentu sporoči, da naj posodobi prikaze na uporabniškem vmesniku. Uporaba tega pristopa zahteva vključitev asinhronega obnašanja izvorne kode za upravljanje primerkov razreda Loader v aktivnost oz. fragment. To posledično pomeni večjo sklopljenost razredov in med drugim oteži izvajanje testov. Vzorec ViewModel je tak pristop v veliki meri zamenjal, saj omogoča enake funkcionalnosti, vendar to stori na bolj preprost in razvijalcu prijazen način. Njegova prednost je tudi v tem, da logiko za upravljanje podatkov bolje loči od uporabniškega vmesnika.

Arhitekturni vzorec ViewModel implementiramo tako, da ustvarimo razred, ki implementira vmesnik ViewModel (poleg konceptualnega opisa vzorcev je del kataloga JetPack tudi njihova implementacija). Razred vsebuje objekte, ki hranijo stanje. Te objekte nato posredujemo krmilniku uporabniškega vmesnika in jih ob interakciji uporabnika tudi posodobimo. ViewModel pridobiva podatke iz poljubnega vira in jih obdela do te mere, da jih bo krmilnik prikazal brez dodatne obdelave. Vsebuje metode za pridobivanje in vračanje podatkov ter metode za vsa dejanja, ki jih uporabnik lahko proži. ViewModel se pogosta uporablja tudi v kombinaciji z vzorcem LiveData, ki omogoča, da krmilnik uporabniškega vmesnika ob spremembah v razredu ViewModel samodejno prejme posodobljene podatke.

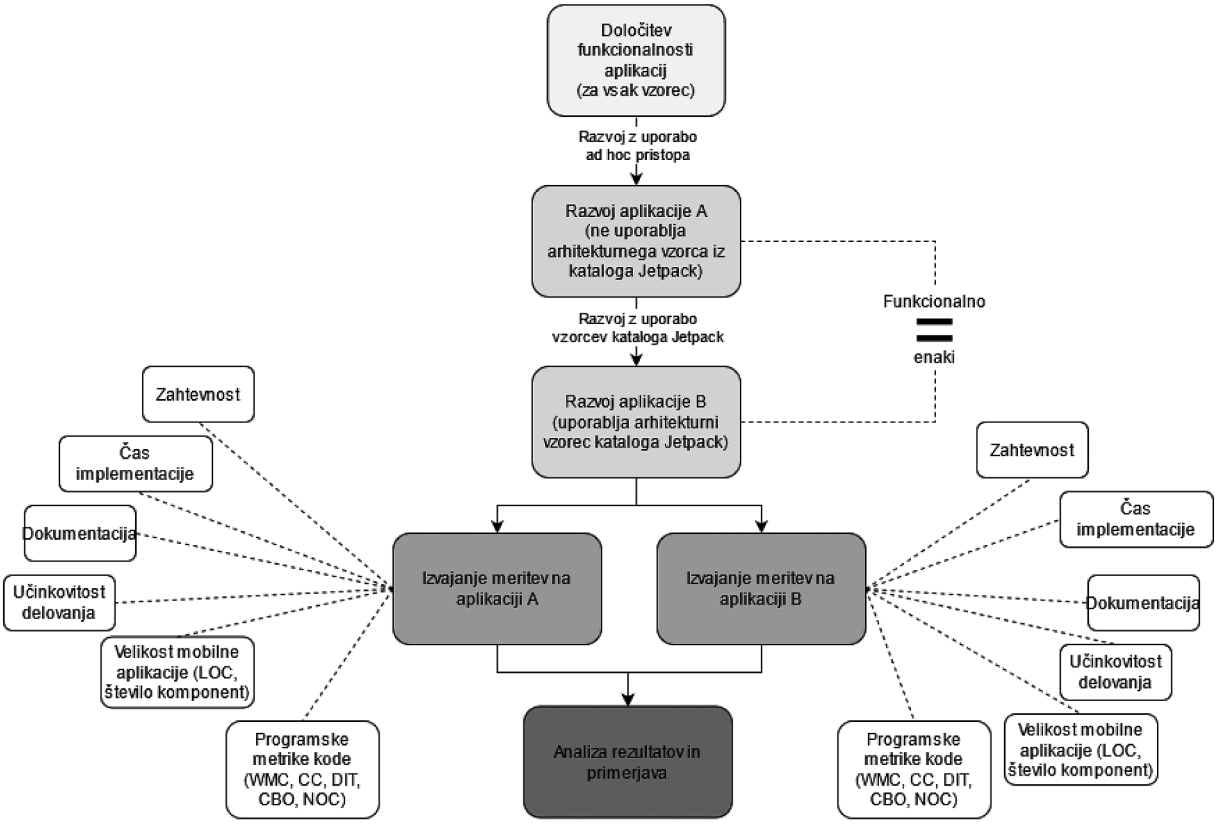
Primer razreda v programskem jeziku Kotlin, ki implementira vzorec ViewModel in skrbi za shrambo, pridobivanje ter posredovanje seznama uporabnikov:

```
class UporabnikiViewModel : ViewModel() {
    private val uporabniki: MutableLiveData<List<Uporabnik>> by lazy {
        MutableLiveData().also {
            naloziUporabnike ()
        }
    }
    fun vrniUporabnike(): LiveData<List<Uporabnik>> {
        return uporabniki
    }
    private fun naloziUporabnike() {
       //asinhrono pridobivanje seznama uporabnikov
    }
}
```

Vzorec ViewModel kataloga Jetpack omogoča relativno preprosto implementacijo ločitve podatkov od prikaza in ohranjanja stanja. Uporabimo pa ga lahko celo za prenos podatkov med različnimi aktivnostmi oz. fragmenti. Poleg lažjega prikaza podatkov razvijalce vzpodbuja tudi k vpeljavi arhitekturnega vzorca MVVM v mobilno aplikacijo.

3 METODE RAZISKOVANJA

Arhitekturne komponente kataloga Jetpack smo preizkusili tako, da smo za vsak vzorec ustvarili dve mobilni aplikaciji, ki vsebujeta enake funkcionalnosti. Arhitekturne vzorce smo uporabili le pri eni.



Slika 4: **Potek raziskovanja**

Raziskovalna metoda je prikazana na sliki 4:

1. Določitev funkcionalnosti
   Vsak arhitekturni vzorec smo preučili in določili funkcionalnosti, s katerimi bi lahko izčrpno preizkusili delovanje izbranega vzorca.
2. Na podlagi funkcionalnosti smo razvili mobilno aplikacijo A. Pri razvoju nismo uporabili arhitekturnih vzorcev, ampak je ta potekal ad hoc. V praksi je to pomenilo, da smo uporabljali že obstoječe knjižnice, funkcionalnosti, ki jih nudi že operacijski sistem Android sam. V nekaterih primerih smo komponente, potrebne za implementacijo željenih funkcionalnosti, ustvarili sami.
3. Razvili smo mobilno aplikacijo B, ki je funkcionalno enaka aplikaciji A, le da smo pri razvoju uporabili arhitekturne vzorce kataloga Jetpack. Da bi zagotovili, da je edina sprememba med aplikacijama A in B uporaba ustreznega vzorca, smo mobilno aplikacijo B razvili na osnovi implementacije A z uvedbo ustreznih sprememb.
4. Na osnovi izvorne kode obeh mobilnih aplikacijah smo zbrali podatke. Izvedli smo anketo (zahtevnost implementacije, čas implementacije, kako dobra je podpora oz. dokumentacija) in izvedli meritve. Z njimi smo ugotavljali učinkovitost delovanja, velikost rešitve (s pomočjo metrike LOC, števila komponent, metrike WMC), njeno kompleksnost (metrika CC) ter ostale strukturne značilnosti (metrike DIT, CBO in NOC).
5. Analizirali smo rezultate in pripravili primerjavo nastalih mobilnih aplikacij.

Izvajane meritve so bile sledeče:

- Zahtevnost (subjektivna ocena razvijalca: zelo nizka (1) – zelo visoka (7)),
- Čas implementacije (subjektivna ocena razvijalca,

temelječa na dejanskem merjenju časa: zelo kratek (1) – zelo dolg (6)),
- Podpora/dokumentacija (subjektivna ocena razvijalca: zelo slaba (1) – odlična (6)),
- Učinkovitost delovanja (merjenje porabe virov in odzivnosti mobilne aplikacije, agregirano v razrede: zelo slaba (1) – odlična (6)),
- Velikost mobilne aplikacije (LOC, število potrebnih komponent):
  - Število komponent – število med seboj različnih delov programske, kode, ki so nujni za implementacijo izbranega vzorca. V to štejemo razrede, statične objekte jezika Kotlin, notranje razrede, poslušalce ipd.,
- Metrike notranje kakovosti izvorne kode (CC, WMC, NOC, DIT, CBO).

Pri *zahtevnosti* smo upoštevali število potrebnih komponent, čas, težavnost in količino novega znanja, ki ga za uporabo komponente razvijalec potrebuje. Ocenili smo jo z opisno oceno na lestvici, ki se giblje od vrednosti »zelo nizka« do vrednosti »zelo visoka«.

Pri *času* smo merili čas, ki je bil potreben za implementacijo. Tudi tu smo se zaradi boljše preglednosti odločili za predstavitev podatkov v obliki opisne ocene na lestvici, ki se giblje od vrednosti »zelo kratek« do vrednosti »zelo dolg«.

Pri *podpori/dokumentaciji* smo se posvetili količini relevantne dokumentacije in njeni kakovosti. Ocenili smo jo z opisno oceno na lestvici, ki se giblje od vrednosti zelo slaba do vrednosti odlična.

Pri *učinkovitosti* delovanja smo merili porabo virov in gladkost izrisovanja uporabniškega vmesnika. Ocenili smo jo z opisno oceno na lestvici, ki se giblje od vrednosti »zelo slaba« do vrednosti »odlična«.

Metrike kakovostni izvorne kode smo izvajali s pomočjo vtičnikov CodeMR (CodeMR, 2020) in Metrics Reloaded (Leijdekkers, n.d.) ter orodja SonarCloud (SonarCloud, 2020).

4 REZULTATI

Meritve vseh implementacij smo združili in jih predstavili v tabeli 1 ter grafikonu (slika 5).

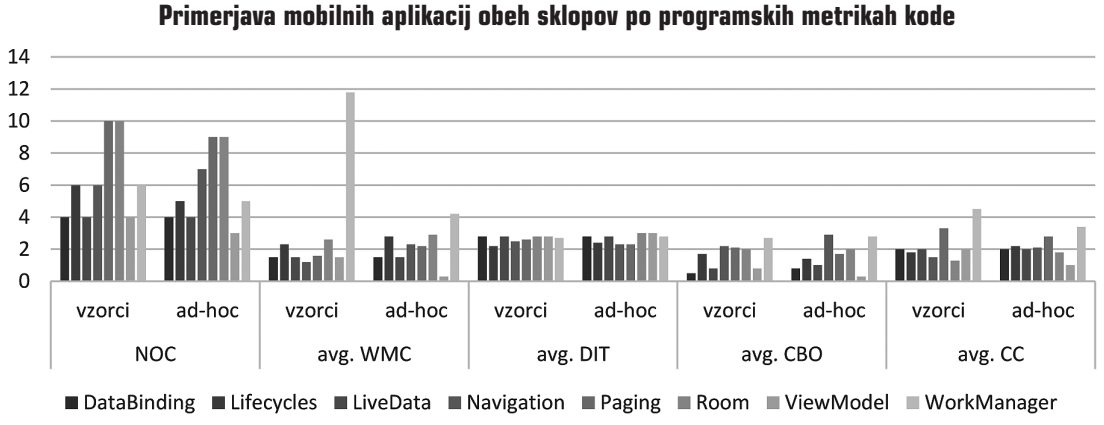
Mobilnim aplikacijam, ki so uporabljale vzorce iz kataloga Jetpack smo podali zelo podobne ocene pri kriterijih dokumentacija, čas in učinkovitost delovanja. Vsem smo pri dokumentaciji in učinkovitosti delovanja podelili najvišjo oceno, pri času implementacije pa najvišje ocene nismo podelili le mobilnima aplikacijama z vzorcema Paging in Room. Večje odstopanje smo opazili pri mobilni aplikaciji, ki uporablja vzorec WorkManager, saj vsebuje približno 500 (za faktor 3,5) vrstic kode več kot njena alternativa. Pri mobilnih aplikacijah, ki ne uporabljajo arhitekturnih vzorcev kataloga Jetpack so rezultati bolj raznoliki, vendar nobena posamezna implementacija ne izstopa.

Tudi pri programskih metrikah kakovosti izvorne kode in bilo večjih razlik. Izstopa le vrednost metrike WMC za implementacijo, ki uporablja vzorec Work Manager (slika 5).

Tabela 1: **Meritve značilnosti implementacij z in brez uporabe arhitekturnih vzorcev kataloga Jetpack**

vzorec	**dokum.**		**čas**		**zahtevnost**		**št. komp.**		**učinkovitost delovanja**		**LOC**	
Uporablja vzorce?	da	ne	da	ne	da	ne	da	ne	da	ne	da	ne
DataBinding	6	6	2	2	2	2	3	3	6	2	30	32
Lifecycles	6	5	2	1	2	1	2	1	6	6	102	88
LiveData	6	4	2	4	1	4	2	2	6	5	32	34
Navigation	6	4	2	3	1	3	1	2	6	6	76	109
Paging	6	2	5	2	5	4	4	1	6	2	232	234
Room	6	4	4	5	4	5	5	4	6	6	152	153
ViewModel	6	6	2	1	2	2	2	1	6	6	32	21
WorkManager	6	3	2	4	4	5	4	4	6	5	723	209

Legenda. dokum. = dokumentacija; čas = čas implementacije; zahtevnost = zahtevnost implementacije; št. komp. = število potrebnih komponent; LOC = število vrstic kode;



Slika 5: **Primerjava mobilnih aplikacij obeh sklopov po programskih metrikah kode**

V tabeli 2 so zbrani rezultati vseh implementacij. Podane so skupne ocene za implementacije z uporabo vzorcev kataloga Jetpack in implementacije brez. Skupno oceno za vsako vrsto implementacij smo pridobili tako, da smo izračunali povprečje vrednosti implementacij posameznega sklopa. Med analizo rezultatov smo ugotovili, da število vrstic kode pri vzorcu WorkManager izstopa in močno vpliva na skupen rezultat. Zaradi tega smo se odločili, da podamo tudi rezultate, ki ne vključujejo vzorca WorkManager.

5 RAZPRAVA

Z rezultati smo pokazali, da so se mobilne aplikacije, razvite z uporabo arhitekturnih vzorcev kataloga Jetpack in tiste razvite ad hoc, med seboj najbolj razlikovale po zahtevnosti implementacije, kakovosti dokumentacije in obsegu izvorne kode. Pri učinkovitosti delovanja, času implementacije in programskih metrikah kode so bile zaznane razlike zelo majhne. Najbolj je izstopala implementacija aplikacije, ki uporablja vzorec WorkManager, saj je imela za faktor 3,5 več vrstic izvorne kode kot implementacija aplikacije, razvite ad hoc. To odstopanje je lahko posledica tega, da so pri podjetju Google ukinili vse ostale knjižnice, ki omogočajo izvajanje nalog v ozadju in jih združili v vzorcu WorkManager. Zato je bilo ad hoc težavno ustvariti aplikacijo, ki je funkcionalno popolnoma enaka tisti z vzorcem WorkManager. Povprečen čas implementacije mobilne aplikacije z uporabo arhitekturnih vzorcev kataloga Jetpack je primerljiv, v nekaterih primerih pa malo daljši od časa implementacije aplikacije, razvite ad hoc. To je v skladu z avtorjem (Fowler, Software Architecture Guide, 2019), ki pravi da uporaba kakovostne arhitekture razvoj na začetku upočasni. Pri podjetju Google obljubljajo, da uporaba arhitekturnih vzorcev kataloga Jetpack zmanjša zahtevnost, pohitri razvoj in izboljša kakovost. Na podlagi naših rezultatov trditev o krajšem času razvoja in višji kakovosti mobilnih aplikacij nismo mogli z gotovostjo potrditi ali ovreči. Dopuščamo možnost, da se večje razlike pojavijo šele pri bolj kompleksnih implementacijah in na dolgi rok. V nadaljevanju podajamo odgovore na začetku zastavljenih raziskovalnih vprašanj.

1. **Katere so prednosti uporabe arhitekturnih vzorcev Jetpack pri razvoju mobilnih aplikacij?**
   a. **Krajši čas razvoja?**
      Pri Googlu obljubljajo, da bo uporaba kataloga Jetpack zmanjšala zahtevnost in pohitrila razvoj, vendar rezultati našega testiranja kažejo, da temu ni vedno tako. Čas implementacije je drugačen pri vsakem vzorcu. Povprečna časa implementacij z in brez uporabe vzorcev kataloga Jetpack sta podobna. Povprečen čas implementacij z uporabo kataloga je celo daljši. Kljub temu smo ugotovili, da vzorci kataloga Jetpack razvijalcu omogočajo lažje dodajanje novih funkcionalnosti in zmanjšajo čas za vzdrževanje in nadaljnji razvoj mobilnih aplikacij. **Uporaba vzorcev kataloga Jetpack ni vedno hitrejša, vendar pa lahko na dolgi rok zmanjša trud in čas za vzdrževanje in razvoj mobilne aplikacije. To je še posebej pomembno pri velikih projektih.**
   b. **Manjša zahtevnost razvoja?**
      Pri pregledu rezultatov smo ugotovili, da je vzorcem iz kataloga Jetpack skupna dobra

Tabela 2: **Primerjava implementacij, ki uporabljajo katalog Jetpack s tistimi, ki ga ne**

	Z uporabo kataloga Jetpack	**Brez uporabe kataloga Jetpack**
Zahtevnost	Nizka – srednje nizka	Srednje nizka – srednja
Čas	Srednje kratek. Za nekaj odstotkov daljši na začetku.	Srednje kratek. Čas je na začetku krajši vendar je vzdrževanje bolj dolgotrajno.
Podpora/ Dokumentacija	Odlična. Pri Googlu nudijo obsežno dokumentacijo in vodiče za vse vzorce.	Dobra. Dokumentacija je bila pogosto pomanjkljiva ali zastarela, podpora za knjižnice pa opuščena.
Povprečno število potrebnih komponent	2,9. Vzorci kataloga Jetpack v povprečju potrebujejo dobre pol komponente več. Implementacija le enega vzorca (Navigation) je potrebovala manj delov, kot implementacija brez njegove uporabe.	2,3. Implementacije brez uporabe kataloga Jetpack so dosledno (razen Navigation) potrebovale manj komponent.
Učinkovitost delovanja	Odlična. Vse implementacije vzorcev imajo majhno porabo virov, mnoge pa vsebujejo dodatne funkcionalnosti, ki aktivno pomagajo preprečiti napake.	Zelo dobra. Zaznali smo le majhna odstopanja od implementacij z uporabo vzorcev kataloga Jetpack, velika razlika pri vzorcu Paging.
Število vrstic kode (LOC)	Rezultati vseh vzorcev: Povprečno: 172 Mediana: 89 Skupno: 1379 Rezultati brez vzorca WorkManager: Povprečno: 94 Mediana: 76 Skupno: 656	Rezultati vseh vzorcev: Povprečno: 110 Mediana: 99 Skupno: 880 Rezultati brez vzorca WorkManager: Povprečno: 96 Mediana: 88 Skupno: 671
Programske metrike kode	Rezultati vseh vzorcev: Povp. WMC: 3 Povp. število razredov: 6,3 Povp. DIT: 2,7 Povp. CBO: 1,6 Povp. CC: 2,3 Rezultati brez vzorca WorkManager: Povp. WMC: 1,7 Povp. število razredov: 6,3 Povp. DIT: 2,6 Povp. CBO: 1,4 Povp. CC: 2	Rezultati vseh vzorcev: Povp. WMC: 2,2 Povp. število razredov: 5,8 Povp. DIT: 2,7 Povp. CBO: 1,6 Povp. CC: 2,2 Rezultati brez vzorca WorkManager: Povp. WMC: 1,9 Število razredov: 35 Povp. DIT: 2,7 Povp. CBO: 1,4 Povp. CC: 2

podpora in predvsem preprostost uporabe. Katalog, ki so ga ustvarili pri Google-u, nudi komponente, ki jih lahko razvijalci hitro in preprosto vključijo v svoj izdelek. Da bi dosegli večjo robustnost in preprostost uporabe, večina vzorcev zanemari določene napredne funkcionalnosti, ki jih nudijo tretje knjižnice. To je še posebej opazno, ko primerjamo vzorec LiveData in knjižnico RxJava. Slednja nudi več funkcionalnosti, vendar je zaradi preprostosti uporabe za večino projektov vzorec LiveData vseeno bolj privlačna izbira. Dobra primera tega sta tudi vzorca Navigation in Room. Le-ta nista uvedla novih funkcionalnosti, ampak sta obstoječe le ovila in razvijalcu poenostavila njihovo uporabo. Ugotovili smo, da so se pri Googlu držali obljub glede nižje zahtevnosti implementacije vzorcev kataloga Jetpack, saj je bila ta v povprečju za petino manjša od implementacij brez njihove uporabe. **Preprostost uporabe je tako eden izmed glavnih adutov kataloga Jetpack. S tem se knjižnica približa tudi začetnikom, ki z razvojem mobilnih aplikacij za operacijski sistem Android še nimajo veliko izkušenj.**

c. Velikost produkta?

Pri analizi števila vrstic kode smo ugotovili, da so bile od njihovih alternativ krajše implemen-

tacije kar petih vzorcev: DataBinding, LiveData, Navigation, Paging, Room. Štiri od teh so bile od alternativ krajše le za 2 vrstici kode ali manj, implementacija vzorca Navigation pa je bila od alternative krajša kar za 40 %. Še posebej je izstopala implementacija vzorca Workmanager, ki je bila od alternative daljša za faktor 3,5. Menimo, da je to zaradi pomanjkanja dobrih alternativ za vzorec WorkManager, zaradi česar je bilo brez njene uporabe zelo težavno ustvariti popolnoma enakovredno implementacijo. **Kljub temu odstopanju smo ugotovili, da v splošnem implementacija z uporabo kataloga Jetpack zahteva manj vrstic kode, v primerjavi z implementacijo brez njegove uporabe.** Čeprav je razlika zelo majhna, lahko rečemo, da so se pri podjetju Google obljube o manj t.i. obrtniške kode držali. Kljub manjšemu številu vrstic kode pa so implementacije, ki so uporabljale vzorce kataloga Jetpack, v povprečju potrebovale 26 % več komponent in skoraj 6 % več razredov kot njihove alternative.

d. Izboljšana notranja kakovost?

Aplikacije, razvite z uporabo vzorcev kataloga Jetpack, v povprečju potrebujejo več komponent in razredov kot aplikacije razvite ad hoc. Vendar za njihovo implementacijo porabijo manjše število vrstic kode in približno enako časa. Poleg tega je njihova implementacija tudi manj zahtevna, kar nakazuje na dobro strukturiranost vzorcev. Mobilne aplikacije, ki so uporabljale vzorce kataloga Jetpack, so imele po naših meritvah tudi večjo učinkovitost delovanja. Kljub temu smo po analizi izmerjenih vrednosti metrik kakovosti programske kode ugotovili, da z izjemo metrik WMC in NOC med aplikacijami, ki uporabljajo vzorce iz kataloga Jetpack in tistimi ki jih ne, ni večjih razlik. Aplikacije z vzorci kataloga Jetpack v povprečju vsebujejo več razredov, ciklomatična kompleksnost, globina delovanja in sklopljenost pa so primerljive z vrednosti alternativnih implementacij. Povprečna vrednost metrike WMC za aplikacije, ki uporabljajo arhitekturne vzorce kataloga Jetpack je za 36 % večja od povprečne vrednosti mobilnih aplikacij, ki teh vzorcev niso uporabljale. Ko smo izračunali še povprečne vrednosti brez aplikacij vzorca WorkManager, se je trend obrnil in so bile vrednosti mobilnih aplikacij, ki uporabljajo vzorce kataloga Jetpack, za 11 % manjše od vrednosti njihovih alternativ. Nižji sta bili tudi vrednosti metrik CC in DIT, vendar sta se vrednosti spremenili za manj kot 5 %, zato jim ne moremo pripisati velikega pomena. **Vzorci kataloga Jetpack se osredotočajo na učinkovitost delovanja ter nizko kompleksnost, vendar po naših rezultatih z gotovostjo ne moremo trditi, da uporaba arhitekturnih vzorcev vodi do višje notranje kakovosti mobilnih aplikacij.**

2. Ali, kdaj in v kolikšni meri je arhitekturne vzorce kataloga Jetpack smiselno uporabiti?

Kot so pokazali rezultati, arhitekturne vzorce kataloga Jetpack odlikujeta predvsem nizka zahtevnost implementacije in dobra dokumentacija. To vzorce približa manjšim projektom in manj izkušenim razvijalcem. **Zaradi modularnosti in nezahtevne vključitve v projekt je vzorce smiselno uporabiti tudi pri obstoječih projektih, ki so bili razviti z drugimi pristopi.** Vzorci so primerni tudi za uporabo v večjih projektih, vendar pod pogojem, da razvijalcem zagotavljajo vse funkcionalnosti, ki jih ti potrebujejo. Arhitekturni vzorci kataloga Jetpack za dosego nizke zahtevnosti pogosto žrtvujejo dodatne funkcionalnost in prilagodljivost. Če potrebujejo razvijalci več nadzora in prilagodljivosti pri implementaciji funkcionalnosti, potem arhitekturni vzorci kataloga Jetpack pogosto niso prava izbira. Dobra primera tega sta vzorca LiveData in Navigation. Vzorec navigation omogoča hitro vključitev v projekt in v veliki meri olajša pomikanje med fragmenti. Če želi razvijalec sam upravljati dejanje ob pritisku gumba »nazaj«, mora uporabiti drug pristop, saj vzorec Navigation tega ne omogoča. Podobno smo opazili pri vzorcu LiveData. Ta je zelo preprost za uporabo, vendar se je v kompleksnejših primerih iz vidika ponujenih funkcionalnosti bolje izkazala knjižnica RxJava.

Vzorci kataloga Jetpack so zasnovani tako, da jih je preprosto vključiti v nov ali že obstoječ projekt. Vzorci niso vezani na specifično različico operacijskega sistema Android in so združljivi za nazaj. To pomeni, da bodo funkcionalnosti, implementirane s pomočjo vzorcev kataloga Jetpack, delovale tudi na starejših različicah sistema Android.

Za uporabo enega vzorca ni potrebno vključiti celotnega kataloga vzorcev, ampak se lahko uporabljajo tudi povsem individualno. Kljub temu se nekateri vzorci najbolj izkažejo takrat, ko so uporabljeni skupaj. Najboljši primer tega so vzorci Room, ViewModel in LiveData. Skupaj poskrbijo za hranjenje in prenos podatkov od denimo podatkovne baze vse do uporabniškega vmesnika. Bili so zasnovani za skupno uporabo, zato jih je povezati lažje, kot če bi katerega izmed njih nadomestili z zunanjim vzorcem. Nekateri arhitekturni vzorci kataloga Jetpack so celo integrirani v Android Studio (Room, Navigation), kar še dodatno zmanjša zahtevnost implementacije in vzorce približa razvijalcem.

6 ZAKLJUČEK

V članku smo demonstrirali pomen in vpliv izbire ustrezne arhitekture na potek razvoja mobilnih aplikacij za operacijski sistem Android. Pri raziskovanju smo se osredotočili na osem arhitekturnih vzorcev kataloga Jetpack. Pokazali smo, da je razvoj mobilnih aplikacij z njihovo uporabo ne samo manj zahteven, temveč je kljub večjemu številu komponent število obseg izvorne kode manjši, čas razvoja in kakovost izdelka pa sta primerljiva z alternativnimi pristopi. Ugotovili smo, da so glavne prednosti arhitekturnih vzorcev kataloga Jetpack nizka zahtevnost implementacije, dobra dokumentacija in dobra strukturiranost. Izkazalo se je, da je vzorce smiselno uporabiti tako pri preprostih kot tudi bolj kompleksnih projektih. Preprostost (in posledično manjša prilagodljivost) arhitekturnih vzorcev kataloga Jetpack lahko predstavlja tudi dodatne izzive. Če razvijalci že vnaprej vedo, da potrebujejo veliko nadzora nad delovanjem vseh delov mobilne aplikacije, potem arhitekturni vzorci kataloga Jetpack morda niso prava izbira.

Naš članek izpostavlja pomen arhitekturnih vzorcev pri razvoju mobilnih aplikacij za operacijski sistem Android ter prednosti in pomanjkljivosti uporabe arhitekturnih vzorcev kataloga Jetpack. Razvijalcem pomaga tudi, da se na podlagi zahtev projekta odločijo, če je uporaba vzorcev kataloga Jetpack smiselna ali ne.

ZAHVALA

Ta raziskava je nastala ob podpori raziskovalnega programa št. P2-0057, katerega je sofinancirala Javna agencija za raziskovalno dejavnost Republike Slovenije iz državnega proračuna.

LITERATURA

[1] Android Developers. (2020). Android Jetpack. Pridobljeno iz Android Developers: https://developer.android.com/jetpack

[2] Android Developers. (30. 10 2020). ViewModel. Pridobljeno iz Android Developers: https://developer.android.com/topic/libraries/architecture/viewmodel

[3] Ardas Group Inc. (15. Junij 2017). How long will it take to create your mobile application. Pridobljeno iz Ardas: https://ardas-it.com/how-long-will-it-take-to-create-your-mobile-application

[4] Aymen, D., Ghizlane, E., Naouel, M., Sègla, K. (2019). An exploratory study of MVC-based architectural patterns in Android apps. SAC '19: Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, 1711–1720.

[5] CodeMR. (2020). CodeMR. Pridobljeno iz CodeMR: https://www.codemr.co.uk/

[6] Fowler, M. (19. 7 2004). Presentation model. Pridobljeno iz martinfowler.com: https://martinfowler.com/eaaDev/PresentationModel.html

[7] Fowler, M. (18. 7 2006). GUI Architecture. Pridobljeno 26. 5 2020 iz https://martinfowler.com/eaaDev/uiArchs.html

[8] Fowler, M. (1. 8 2019). Software Architecture Guide. (Martin Fowler) Pridobljeno 28. 5 2020 iz https://martinfowler.com/architecture/

[9] Gossman, J. (8. 10 2005). Introduction to Model/View/ViewModel pattern for building. (Microsoft) Pridobljeno 27. 5 2020 iz https://docs.microsoft.com/en-gb/archive/blogs/johngossman/introduction-to-modelviewviewmodel-pattern-for-building-wpf-apps

[10] Leijdekkers, B. (brez datuma). Metrics Reloaded. (Github) Pridobljeno 2. 8 2020 iz https://github.com/BasLeijdekkers/MetricsReloaded

[11] Lou, T. (2016). A Comparison of Android Native App Architecture – MVC, MVP and MVVM. Eindhoven.

[12] Martin, R. C. (13. 8 2012). The Clean Architecture. Pridobljeno iz Clean Coder Blog: https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html

[13] Moore, K. (4. 7 2018). Introduction to Android Jetpack. (Ray Wenderlich) Pridobljeno 29. 5 2020 iz https://www.raywenderlich.com/5376-introduction-to-android-jetpack#toc-anchor-001

[14] Potel, M. (1996). MVP: Model-View-Presenter. Taglient Inc, 5-9.

[15] Prabowo, G., Suryotrisongko, H., Tjahyanto, A.. (2018). A Tale of Two Development Approach: Empirical Study on The Maintainability and Modularity of Android Mobile Application with Anti-Pattern and Model-View-Presenter Design Pattern. 2018 International Conference on Electrical Engineering and Informatics (ICELTICs).

[16] Reenskaug, T. (2003). MVC XEROX PARC 1978-79. Pridobljeno iz Trygve M. H. Reenskaug: http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html

[17] SonarCloud. (2020). SonarCloud. Pridobljeno iz SonarCloud: https://sonarcloud.io/

[18] Tatarka, E. (2014). holdr. Pridobljeno iz https://github.com/evant/holdr

[19] Verdecchia, R., Malavolta, I., Lago, P. (2019). Guidelines for Architecting Android Apps: A Mixed-Method Empirical Study. 2019 IEEE International Conference on Software Architecture (ICSA). doi:10.1109/ICSA.2019.00023

[20] Wharton, J. (brez datuma). Butterknife. (Jake Wharton) Pridobljeno iz http://jakewharton.github.io/butterknife/

▪

Luka Pavlič je docent na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Doktoriral je leta 2009 iz tematike ponovne uporabe s pomočjo vzorcev. Njegovo raziskovalno delo obsega tehnične in organizacijske aspekte razvoja informacijskih rešitev, njihovo kakovost ter IT arhitekture. Je avtor oz. soavtor večjega števila izvirnih znanstvenih člankov, ki so objavljeni v najuglednejših revijah.

▪

Luka Četina je tehniški sodelavec na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Trenutno je podiplomski študent na študijskem programu Informatika in tehnologije komuniciranja.

Analiza gibanja oči med branjem pri bolnikih z različnimi stopnjami kognitivnega upad

Vida Groznik[1,2,3], Aleksander Sadikov[1,2]
[1]NEUS Diagnostics, d.o.o., Ziherlova ulica 40E, 1000 Ljubljana
[2]Univerza v Ljubljani, Fakulteta za računalništvo in informatiko, Večna pot 113, 1000 Ljubljana
[3]Univerza na Primorskem, Fakulteta za matematiko, naravoslovje in informacijske tehnologije, Glagoljaška 8, 6000 Koper
vida.groznik@fri.uni-lj.si, aleksander.sadikov@fri.uni-lj.si

Izvleček

S staranjem prebivalstva se močno povečuje število bolnikov z demenco, kar predstavlja vedno večji družbeni problem. Poleg samega bolnika z demenco, breme bolezni občutijo tudi njegovi bližnji. Blag kognitivni upad (MCI) je stanje kognitivnega funkcioniranja med tistim, ki ga opazimo pri običajnem staranju in tistim, ki ga zaznamo pri osebah z demenco. Raziskave kažejo, da okvirno 10 % do 15 % bolnikov z MCI v roku enega leta napreduje v Alzheimerjevo demenco. Obenem obstajajo indici, da je v zgodnji fazi kognitivnega upada razvoj bolezni velikokrat še možno zavreti; posledično je zgodnje odkrivanje kognitivnega upada precejšnjega pomena. Gibanje oči med branjem se je v zadnjih letih izkazalo kot eden izmed obetavnih bioloških označevalcev za zaznavanje kognitivnega upada. V pričujoči raziskavi smo analizirali gibanje oči med branjem pri 115 preiskovancih, ki so bili razdeljeni v štiri skupine glede na stopnjo kognitivnega upada; od povsem zdravih do preiskovancev z demenco. Njihov pogled smo posneli s sistemom za spremljanje očesnih gibov, posnetke pa smo opisali z lastnim naborom atributov. Analiza porazdelitev vrednosti atributov je pokazala, da se te značilno razlikujejo v odvisnosti od stopnje kognitivnega upada.

Ključne besede: blag kognitivni upad, demenca, branje, sledenje očesnih gibov

Abstract

With the aging of the population, the number of people suffering from dementia is on the rise. This represents an ever-growing problem for the society as the close relatives and caretakers of patients with dementia share the burden of the disease. Mild cognitive impairment (MCI) is a state of cognitive decline between normal aging and dementia. Recent studies estimate that between 10 % and 15 % of patients with MCI progress to dementia within a year. There are indications that the progression of the disease can often be slowed down during the early stage of the cognitive decline. Therefore, the early detection of the cognitive decline is highly desirable. Eye movement during reading has been shown to be a promising biomarker of MCI. In this study, we analysed the eye movement of 115 participants during reading. The participants were grouped into four levels of cognitive decline; from healthy control to patients with dementia. Their eye movements were recorded with an eye-tracker and these recordings were described with our proposed set of descriptive attributes. The distributions of the attribute values were shown to be significantly different between various levels of cognitive decline.

Keywords: Mild cognitive impairment, dementia, reading, eye-tracking

1 UVOD IN MOTIVACIJA

Večkot 50 milijonov ljudi po celem svetu ima eno izmed oblik demence, kar bolezen uvršča na prvo mesto med vsemi nevrodegenerativnimi boleznimi. Zaradi staranja prebivalstva se predvideva, da bo konec tega desetletja število obolelih z demenco naraslo na 82 milijonov, leta 2050 pa bo to diagnozo imelo kar 152 milijonov ljudi [Alzheimer's Disease International, 2019]. Blag kognitivni upad (MCI) je stanje kognitivnega funkcioniranja med tistim, ki ga opazimo pri običajnem staranju in tistim, ki ga zaznamo pri osebah z demenco.

Za razliko od demence, pri osebah z MCI dejavnosti vsakdanjega življenja in kakovost življenja niso bistveno prizadeti. [Hugo and Ganguli, 2014]

Po ocenah pri nekje med 10 in 15% bolnikov z

MCI bolezen v roku enega leta napreduje v Alzheimerjevo demenco (AD) [Farias et al., 2009]. Raziskave kažejo, da je moč z uporabo različnih pristopov zavreti ali celo preprečiti nastanek demence oz. MCI [Ayati et al., 2020]. Med te pristope sodijo različne terapije za nekatere druge bolezni, kot so različna nesteroidna protivnetna zdravila [Szekely et al., 2008] ter statini [Swiger et al., 2013], različne nefarmacevtske metode, kot na primer spremembe življenjskega sloga [Lourida et al., 2019], uživanje mediteranske prehrane [Lourida et al., 2013], zmanjšano uživanje al- kohola [Letenneur, 2004], fizična telesna aktivnost [Cass, 2017], ipd. Vse te ugotovitve so še dodatna motivacija za raziskovanje različnih bioloških označevalcev za zgodnje odkrivanje kognitivnega upada.

V zadnjih letih se je izkazalo, da je gibanje oči med branjem lahko eden izmed možnih bioloških označevalcev za detekcijo AD [Lueck et al., 2000] in tudi MCI [Fraser et al., 2017]. Razlog za to bi lahko bilo dejstvo, da je branje eno izmed bolj zahtevnih miselnih procesov, ki zahteva uporabo številnih kognitivnih sposobnosti kot so pozornost, kratkoročni in dolgoročni spomin, vidno in slušno procesiranje ter senzorika. Poleg tega so raziskave pokazale, da imajo bolniki z AD težave z branjem zaradi primarnih okulomotornih nepravil- nosti kot je nestabilnost fiksacije [Lueck et al., 2000, Wilcockson et al., 2019]. Pri teh bolnikih se namreč postopoma razvije okvara nadzora inhibicije in popravljanja napak pri premikanju oči, predvsem sposobnost prostovoljnega usmerjanja pogleda stran od stimulusa pri testu antisakad (AST). Poleg tega pogostost napak pri izvajanju nalog AST korelira s stopnjo AD. [Wilcockson et al., 2019] Pomembna ugotovitev je tudi, da se težave z gibanjem oči lahko pojavijo že v zelo zgodnjih fazah razvoja bolezni, ko različnih kognitivnih primanjkljajev s standardnimi nevropsihološkimi testi še ni mogoče zaznati [Crawford et al., 2005].

Glavni namen pričujoče raziskave je ugotoviti, kako se izraža gibanje oči med branjem pri različnih stopnjah kognitivnega upada in posledično, če med skupinami obstajajo pomembne razlike. To bi v nadaljevanju lahko vodilo do avtomatiziranega testa za odkrivanje zgodnjega kognitivnega upada in stopnje le-tega na podlagi branja.

V nadaljevanju članka najprej povzamemo klinično raziskavo, postopek rekrutacije preiskovancev in metode, ki so bile uporabljene med izvajanjem raziskave. V tretjem poglavju orišemo strukturo preiskovancev v tej analizi in atribute, ki smo jih predlagali za potrebe analize očesnih gibov med branjem. Rezultati raziskave so predstavljeni v poglavju štiri, čemur sledi diskusija rezultatov. Zaključki in nadaljnje delo so podani v sklepnem poglavju.

2 KLINIČNA RAZISKAVA

Podatki uporabljeni v pričujoči raziskavi so bili zbrani v okviru klinične raziskave, ki je bila odobrena s strani Komisije Republike Slovenije za medicinsko etiko (št. 0120-400/2015-5 in 0120-400/2015/9). Raziskava je potekala v skladu z dobro klinično prakso in nacionalnimi predpisi s čimer smo zagotovili zaščito pravic, varnost in dobro počutje preiskovancev v skladu z etičnimi načeli, ki izhajajo iz Helsinške deklaracije Svetovnega zdravniškega združenja [World Medical Association, 2013].

2.1 Rekrutacija preiskovancev

Vsi preiskovanci so k raziskavi pristopili prostovoljno. Informacijo o možnosti sodelovanja so prejeli na različne načine: s strani osebnih zdravnikov ali nevrologov po tem, ko so izrazili skrb, da imajo kognitivne težave (subjektivna kognitivna motnja), od raziskovalcev vključenih v raziskavo ali od predhodno vključenih preiskovancev.

Ob sami rekrutaciji smo upoštevali vključitvene in izključitvene kriterije. Osebe, vključene v raziskavo, so tako morale biti starejše od 40 let pri čemer niso smele imeti nekorigiranih vidnih napak, konkomitantnih nevroloških bolezni (izjema sta seveda blag kognitivni upad in demenca) ter niso smele zlorabljati mamil ali alkohola.

2.2 Potek raziskave

Pred vključitvijo v raziskavo je bil vsak preiskovanec pisno in ustno seznanjen z vsebino in namenom raziskave, predvidenim trajanjem raziskave in z njegovo pravico, da lahko kadarkoli odstopi od raziskave. Vsak preiskovanec je nato podpisal privolitev za sodelovanje v raziskavi.

Raziskava je potekala v treh sklopih:

1. Nevrološki pregled

Nevrološki pregled je zajemal ocenjevanje kognitivnega stanja preiskovancev in po potrebi še njihovega motoričnega in nemotoričnega stanja.

S pomočjo vprašalnika smo preiskovance povprašali o demografskih podatkih in sicer o: letnici

rojstva, ročnosti, izobrazbi, poklicu, datumu (morebitne) diagnoze, dolžini trajanja (morebitne) bolezni, trenutni terapiji, ostalih boleznih, morebitnih poškodbah glave in o družinski anamnezi.

Okviren čas trajanja: 15–20 minut

2. Nevropsihološko testiranje

Nevropsihološko testiranje je zajemalo oceno višjih kognitivnih sposobnosti, kot so izvršilne funkcije in spomin. Pri testiranju so bili uporabljeni naslednji standardizirani teti: Kognitivni preizkus Addenbrooke-KPA-R [Mioshi et al., 2006], Kratka baterija frontalnih testov - FAB (*angl. Frontal Assessment Battery*) [Slachevsky et al., 2004], CTMT (*angl. Comprehensive Trail Making Test*) [Bowie and Harvey, 2006] in Geriatrična lestvica depresivnosti - 15 vprašanj (*angl. Geria- tric Depression Scale - 15 questions*) [Conradsson et al., 2013].

Okviren čas trajanja: 45–60 minut

3. Testiranje očesnih gibov

Testiranje očesnih gibov je bilo opravljeno z namensko programsko opremo NEUS slovenskega proizvajalca NEUS Diagnostics, d.o.o., ki je bil sponzor raziskave. Programska oprema je bila nameščena na prenosni računalnik, ki je imel priklopljen dodatni zaslon (z diagonalo velikosti 61 cm) in napravo za sledenje očesnim gibom (*angl. eye-tracker*). Uporabili smo model naprave 4C švedskega proizvajalca Tobii s frekvenco zajema 90Hz. Preiskovanec je sedel približno 70cm od zaslona in ni imel neposrednega stika z opremo. Interakcija preiskovanca s programom je potekala zgolj z njegovim pogledom na zaslon. Celoten postopek testiranja je izvedel in nadzoroval tehnik, ki je upravljal z računalnikom in programsko opremo. Navodila za posamezno nalogo so bila izpisana na računalniškem zaslonu in hkrati predvajana v zvočni obliki.

Preiskovancem smo vizualne dražljaje prikazali na računalniškem zaslonu z uporabo namenske programske opreme NEUS, ki jih je tudi vodila skozi vse naloge baterije testov. Programska oprema je podatke o gibanju oči, ki jih je beležila s pomočjo naprave za sledenje pogleda, shranila v podatkovno bazo. Vsi podatki so bili anonimizirani, dostop do podatkovne baze pa so imeli zgolj raziskovalci, ki so sodelovali v raziskavi.

Pred začetkom samega testiranja, je bilo potrebno izvesti kalibracijo sistema ločeno za vsakega preiskovanca. Po uspešni začetni kalibraciji so bili preiskovanci pozvani, da preberejo besedilo, ki je bilo prikazano na zaslonu. Besedilo je obsegalo 13 vrstic in je bilo razdeljeno v štiri odstavke. Ko je preiskovanec prebral celotno besedilo, je moral to eksplicitno potrditi s pogledom v spodnji desni kot zaslona, kjer je bila izrisana kljukica.

Okviren čas trajanja: 25–30 minut

Na podlagi nevrološkega pregleda in nevropsihološkega testiranja so preiskovanci prejeli eno izmed naslednjih diagnoz: *zdrav* (brez kognitivnega upada), *mejno* (zaznanih nekaj znakov kognitivnega upada a ne dovolj, da bi preiskovanca lahko diagnosticirali z blagim kognitivnim upadom), *MCI* (blag kognitivni upad) ali *demenca*.

3 PODATKI

3.1 Preiskovanci

Podatki, ki smo jih uporabili za to raziskavo, so vključevali 115 zaporedno vključenih preiskovancev starih med 43 in 94 let z mediano 68 let. Med njimi je bilo 85 preiskovank in 30 preiskovancev.

Zdravih preiskovancev je bilo 53, diagnozo *mejno* je imelo 32 preiskovancev, 19 preiskovancev je imelo MCI in 11 demenco. Podrobnejša porazdelitev preiskovancev po starosti in spolu glede na diagnozo/skupino je prikazana v Tabeli 1.

Tabela 1: **Porazdelitev preiskovancev glede na spol in starost za posamezno skupino.**

	zdrav	**mejno**	**MCI**	**demenca**
N	53	32	19	11
Starostna porazdelitev				
Najnižja starost	48	60	43	72
Najvišja starost	83	87	91	94
Srednja starost	63	68.5	72	83
Porazdelitev po spolu				
Ženska	40	24	12	9
Moški	13	8	7	2

3.2 Atributi za opis očesnih premikov med branjem

Gibanje oči med branjem ni gladko, temveč tvori izmenično zaporedje sakad (angl. saccade) in fiksacij (angl. fixation). Sakade so hitri očesni premiki med dvema točkama, fiksacije pa krajše zaustavitve pogleda na neki točki. Upoštevaje frekvenco zajema podatkov, lastnosti zaslona in velikost besedila v našem konkretnem primeru, smo fiksacije definirali kot zaporedje

posnetkov trajajoče najmanj 48ms, pri čemer največja razdalja poljubnih dveh posnetkov znotraj zaporedja ne presega 100 pikslov. Sakade smo izračunali kot premike med dvema zaporednima fiksacijama.

Predobdelava posnetkov branja je vključevala izločitev neveljavnih časovnih točk v posnetku; to so časovne točke, ko naprava za sledenje pogleda le-tega ni zaznala. Razlogi za to so lahko različni, najpogosteje gre za mežikanje z očmi, lahko pa do tega pride tudi zaradi pogleda izven površine zaslona, nerodne pozicije glave ipd. Poleg tega smo odstranili začetnih 5 % in končnih 10 % posnetka. Na teh mestih zna biti obnašanje preiskovanca nepovezano s samim branjem.

Branje oziroma gibanje pogleda med branjem smo opisali z naborom atributov, ki smo jih v ta namen definirali. Pri tem smo zasledovali cilj, da poskušamo zajeti kar se da veliko pomembnih značilnosti branja s čim manjšim naborom atributov za kasnejšo uporabo v strojnem učenju. Pri zasnovi atributov smo se opirali na izkušnje iz lastne pilotne raziskave ter na domensko znanje, ki smo ga pridobili od nevrologov in psihologov. Atribute smo definirali na podlagi predhodno zaznanih sakad in fiksacij. Podrobneje so opisani v nadaljevanju.

Dolžina sakad v smeri naprej (*fdist*) in v smeri nazaj (*bdist*) je definirana kot srednja dolžina sakad v ustrezni smeri (naprej oziroma nazaj). Smer sakade je razvidna iz zaslonskih koordinat fiksacije pred in po izvedeni sakadi. Oba atributa sta povezana s hitrostjo branja, vsebujeta pa tudi druge zanimive informacije. Dolžina sakad nazaj je na primer neposredno povezana s skoki v novo vrstico besedila ter tudi morebitnim ponovnim branjem določenih besed.

Razpršenost dolžin sakad nazaj (*bdist.stdev*) se izračuna kot standardni odklon dolžin vseh zaznanih sakad v smeri nazaj. Kot že omenjeno, ta atribut je tesno povezan s skokom v novo vrstico besedila. Večinoma gre zato tukaj za precej dolge sakade, dolge približno toliko kolikor je dolga vrstica teksta. Visoka vrednost tega atributa, torej velika razpršenost dolžin sakad, bi lahko sporočala, da uporabnik poleg skokov v novo vrstico dela tudi precej krajših sakad nazaj, morda zato, ker ponovno prebira določene besede ali ker se je v besedilu izgubil.

Razpršenosti dolžin sakad naprej nismo uporabili kot atribut, ker ne izhaja iz domenskega znanja, kot razpršenost dolžin sakad nazaj. Slednji je zanimiv zaradi morebitne mešanice dolgih skokov v novo vrstico in kratkih pomikov pogleda nazaj, ki so odraz ponovnega branja ene ali več predhodnih besed. Pri branju naprej daljši skoki v smeri branja niso prisotni in bi bil atribut posledično odveč.

Hitrost sakad v smeri naprej (*fspeed*) in sakad v smeri nazaj (*bspeed*) je definirana kot srednja hitrost sakad v ustrezni smeri. Kljub temu, da sta dolžina in hitrost sakad do neke mere korelirani med seboj, smo že v pilotni raziskavi opazili, da vsaka doprinese dodatne informacije o gibanju pogleda med branjem.

Trajanje fiksacije (*fdur*) izračunamo kot srednjo vrednost trajanja vseh zaznanih fiksacij, razpršenost trajanja fiksacij (*fdur.std*) pa kot standardni odklon trajanj fiksacij. Oba atributa lahko povežemo z nekonsistentim in zmedenim obnašanjem med branjem.

Razmerje med sakadami naprej in nazaj (*fsV bs*) izračunamo kot število sakad naprej deljeno s številom sakad nazaj. Število fiksacij na časovno enoto (*fxt*) pa izračunamo kot število vseh zaznanih fiksacij deljeno s trajanjem posnetka. Ta atribut predstavlja bolj robustno mero za hitrost branja.

4 REZULTATI

V raziskavi nas je predvsem zanimalo, če (1) preiskovanci z različno stopnjo kognitivnega upada izkazujejo različne značilnosti gibanja pogleda med branjem in (2) če naši atributi uspejo te razlike zaznati. S tem namenom smo primerjali porazdelitve vrednosti posameznih atributov med vsemi štirimi skupinami preiskovancev.

Porazdelitve vrednosti atributov večinoma ne sledijo normalni porazdelitvi, zato smo za njihovo primerjavo uporabili neparametrični Kruskal-Wallisov H test. Rezultati so zbrani v Tabeli 2. Poleg statistične značilnosti (p-vrednosti) so v tabeli navedene še nekatere druge lastnosti porazdelitev vrednosti.

Statistične značilnosti nakazujejo, da so med skupinami preiskovancev (velike) razlike v porazdelitvah predlaganih atributov. Z izjemo atributa *bdist.std*, ki je dokaj mejno statistično značilen, so vsi preostali močno statistično značilni. Vendar pa je Kruskal-Wallisov H test omnibus test, torej sporoča, če je katera od skupin značilno drugačna od katerekoli druge skupine v porazdelitvi atributa. Namesto naknadnih dodatnih primerjav med posameznimi skupinami smo porazdelitve vrednosti predstavili raje grafično na Sliki 1. Na sliki so prikazani histogrami za vse atribute po vseh skupinah. Poleg tega so gostote porazdelitev vrednosti atributov po skupinah

ocenjene z jedri (angl. kernel density estimation) in prikazane s krivuljami na histogramih.

Iz grafov na Sliki 1 lahko razberemo, da vsi atributi (z izjemo *bdist.std*) izkazujejo različne porazdelitve med skupinami, predvsem med skupinama zdravih preiskovancev in preiskovancev z demenco. Nadalje je iz grafov razvidno tudi, da skupine izkazujejo urejenost po stopnji kognitivnega upada, od zdravih preko mejnih in preiskovancev z blagim kognitivnim upadom pa do preiskovancev z demenco.

Tabela 2: **Porazdelitev vrednosti atributov sledenja očesnim gibom po skupinah preiskovancev (zdrav, mejno, MCI, demenca).**

Atribut	p–vrednost	Skupina	Mediana	Stand. dev.	Minimum	Maksimum
		zdrav	–45.91	15.86	–52.23	–8.75
bdist	< 0.001	mejno	–37.56	15.85	–65.59	–8.11
[% zaslona]		MCI	–16.95	16.10	–47.77	–7.48
		demenca	–13.16	4.45	–20.52	–7.81
		zdrav	18.15	3.01	7.79	22.45
bdist.std	0.036	mejno	18.62	2.01	14.12	22.26
[% zaslona]		MCI	19.89	3.86	8.51	25.95
		demenca	16.49	3.80	8.34	21.10
		zdrav	–4.30	1.70	–8.37	–0.37
bspeed	< 0.001	mejno	–4.36	1.31	–7.44	–1.35
[piksel/ms]		MCI	–3.63	1.36	–6.10	–0.94
		demenca	–2.74	0.89	–4.45	–0.92
		zdrav	7.74	1.01	6.43	11.48
fdist	0.003	mejno	7.40	1.10	5.90	10.46
[% zaslona]		MCI	7.55	1.13	6.08	9.86
		demenca	6.85	0.70	5.72	7.80
		zdrav	216.51	41.87	127.69	366.41
fdur	0.001	mejno	244.29	58.06	111.03	421.92
[ms]		MCI	249.82	72.71	111.03	377.51
		demenca	294.31	92.76	144.34	410.82
		zdrav	111.48	33.91	42.51	220.09
fdur.std	< 0.001	mejno	152.53	48.64	86.38	270.36
[ms]		MCI	148.25	75.53	92.66	400.54
		demenca	226.20	37.39	156.10	286.56
		zdrav	4.43	1.26	1.93	6.30
fsVbs	0.003	mejno	4.03	1.12	2.10	6.57
[brez enote]		MCI	3.55	0.87	2.20	5.21
		demenca	2.88	0.74	1.94	4.57
		zdrav	3.17	0.61	0.23	3.99
fspeed	< 0.001	mejno	3.07	0.74	0.68	4.00
[piksel/ms]		MCI	2.85	0.79	0.80	3.80
		demenca	2.65	0.66	0.64	3.03
		zdrav	3.28	0.67	0.71	4.80
fxt	< 0.001	mejno	2.76	0.49	1.86	3.82
[#fiksacij/s]		MCI	2.57	0.40	2.01	3.63
		demenca	2.04	0.34	1.59	2.82

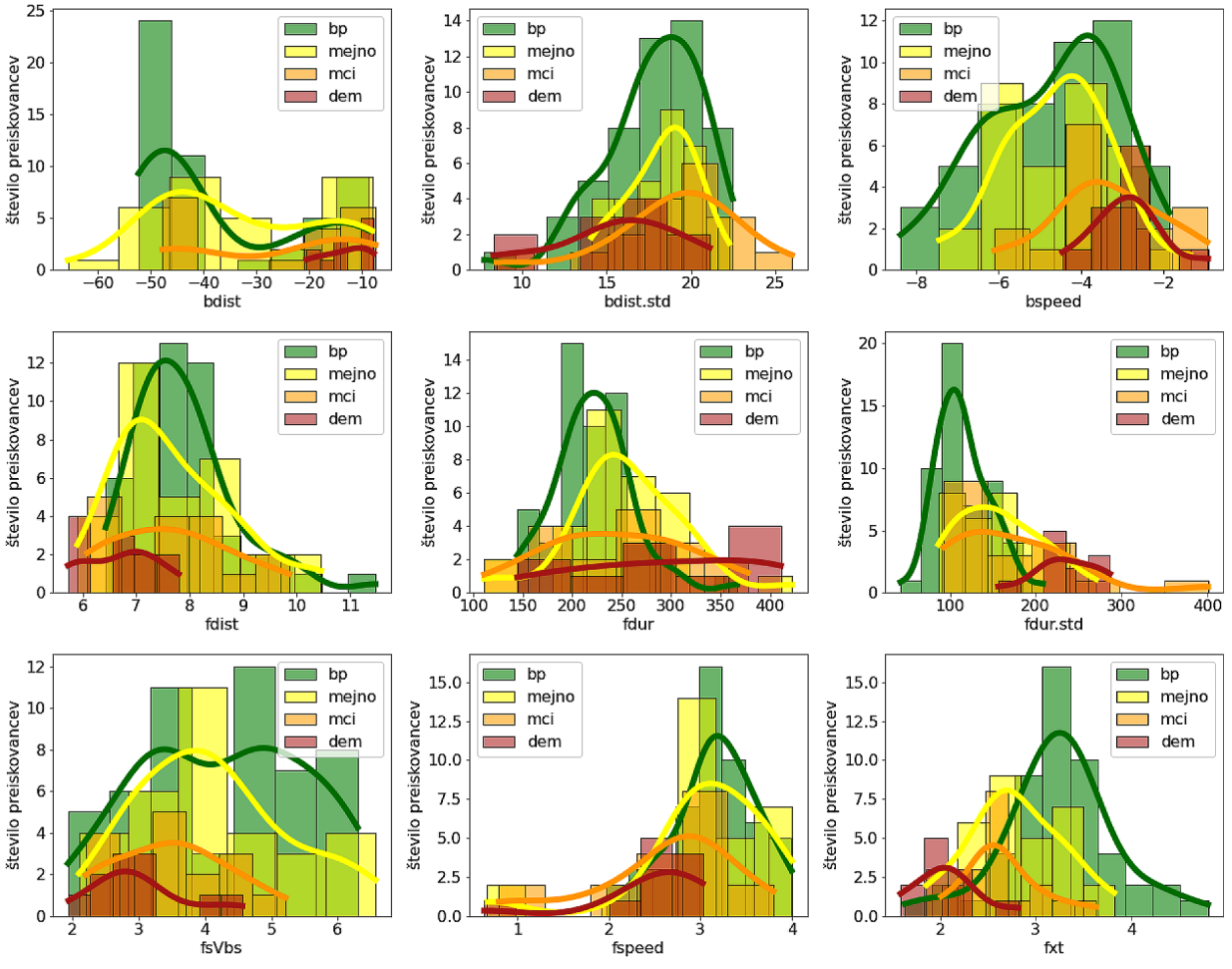
5 RAZPRAVA

V pričujočo študijo je bilo vključenih 115 preiskovancev. Vsi preiskovanci so opravili precej izčrpen nabor nevropsiholoških preiskav pri specialistu nevrologu in psihologu, na podlagi česar so bili razvrščeni v ustrezno skupino glede na stopnjo (morebitnega) kognitivnega upada. Na ta način je študija dobro pokrita tako z vidika velikosti vzorca kot tudi z vidika kvalitete podatkov.

Študija je pokazala, da obstajajo precejšnje razlike v gibanju pogleda med branjem pri preiskovancih z različno stopnjo kognitivnega upada. Prav tako je tudi razvidno, da so te razlike sposobni sistematično zaznati predlagani atributi za opis značilnosti gibanja pogleda med branjem. Tu velja posebej poudariti urejenost skupin kot jo prikazujejo porazdelitve vrednosti posameznih atributov. Le-ta ustreza nevrološkim in psihološkim pričakovanjem, da je kognitivni upad pravzaprav kontinuum, ki se počasi stopnjuje; od mejnega preko blagega kognitivnega upada pa vse do demence. To se tudi ujema z izsledki raziskav, ki predvidevajo, da relativno velik delež bolnikov z MCI v nekaj letih razvije različne oblike demence (npr. [Alexopoulos et al., 2006, Farias et al., 2009, Glynn et al., 2021]).

Iz porazdelitev vrednosti atributov na Sliki 1 je možno razbrati, da imajo preiskovanci z večjo stopnjo kognitivnega upada vse nižje število fiksacij na časovno enoto, njihove fiksacije so tipično vse daljše in predvsem vse bolj razpršenih trajanj (mešanica kratkih in dolgih fiksacij), hitrost njihovih sakad je vse počasnejša in dolžina sakad vse krajša (predvsem sakad v smeri nazaj). Tudi te ugotovitve se skladajo s pričakovanji.

Branje je vseprisotna človeška aktivnost. Kot tako je zelo naravno in primerno za poljubnega preisko-



Slika 1: **Porazdelitve vrednosti atributov po posameznih skupinah. Na grafih so prikazani histogrami in krivulje, ki predstavljajo oceno gostote porazdelitve z jedri.**

vanca, še posebej glede na to, da gre za kratko besedilo s precej veliko pisavo. Naprava za spremljanje pogleda je prav tako nemoteča, marsikareri preiskovanec tanke podolgovate ploščice pod zaslonom niti zares ne opazi. S tega vidika so dobljeni rezultati verjetno zelo realistični (gre res za značilnosti pogleda med branjem) in niso posledica npr. boljše sposobnosti prilagajanja preiskovanca na situacijo (angl. ecological validity).

6 ZAKLJUČKI IN NADALJNJE DELO

Na podlagi pričujoče raziskave lahko zaključimo, da obstajajo precejšnje razlike pri branju med osebami z različnimi stopnjami kognitivnega upada. Nadalje lahko potrdimo, da je te razlike možno zaznati preko predlaganih atributov s pomočjo naprav za spremljanje očesnih gibov. Prav tako se je pokazalo, da je kognitivni upad kontinuum. Faza "mejno", ki klinično še ne izkazuje zadostnih znakov za diagnozo blagega kognitivnega upada, se vendarle razlikuje od zdravega stanja. To je koristen podatek, ker nakazuje, da lahko kognitivni upad v določenih primerih zaznamo v še bolj zgodnji fazi.

Pomen odkritih razlik oziroma dejstva, da je s pomočjo predlaganega nabora atributov in spremljanja pogleda med branjem moč zaznati (blag) kognitivni upad, je v možnosti avtomatizacije tega postopka. Nadaljnje delo bo usmerjeno predvsem v uporabo predlaganih atributov v postopku izdelave napovednega modela prisotnosti znakov kognitivnega upada s pomočjo strojnega učenja ter njegove evalvacije. Poleg tega načrtujemo izdelavo baterije testov, ki bi vključevala branje, ter nekatere druge nevropsihološke teste, prilagojene za delovanje s sistemi za spremljanje očesnih gibov, kot je na primer naloga s Corsijevimi kockami [Groznik, 2020].

ZAHVALA

Raziskava je potekala v okviru projekta NEUS, ki je bil financiran s strani Evropskega inštituta za inovacije in tehnologijo - skupnost za inovacije znanja "Zdravje" (European Institute of Innovation and Technology (EIT) Health KIC). To telo Evropske Unije prejema podporo v okviru raziskovalnega in inovacijskega programa Horizon 2020.

LITERATURA

[1] [Alexopoulos et al., 2006] Alexopoulos, P., Grimmer, T., Perneczky, R., Domes, G., and Kurz, A. (2006). Progression to dementia in clinical subtypes of mild cognitive impairment. *Dementia and geriatric cognitive disorders*, 22(1):27–34.

[2] [Alzheimer's Disease International, 2019] Alzheimer's Disease International, L. (2019). *World Alzheimer Report 2019: Attitudes to dementia*. Alzheimer's Disease International (ADI).

[3] [Ayati et al., 2020] Ayati, Z., Chang, D., and Lake, J. (2020). Advances in treatment of mild cognitive impairment (mci) and dementia: A review of promising non-pharmaceutical modalities. *Frontiers in Clinical Drug Research-Dementia: Volume 1*, 1:78.

[4] [Bowie and Harvey, 2006] Bowie, C. R. and Harvey, P. D. (2006). Administration and interpretation of the Trail Making Test. *Nature protocols*, 1(5):2277.

[5] [Cass, 2017] Cass, S. P. (2017). Alzheimer's disease and exercise: a literature review. *Current sports medicine reports*, 16(1):19–22.

[6] [Conradsson et al., 2013] Conradsson, M., Rosendahl, E., Littbrand, H., Gustafson, Y., Olofsson, B., and Lövheim, H. (2013). Usefulness of the Geriatric Depression Scale 15-item version among very old people with and without cognitive impairment. *Aging & mental health*, 17(5):638–645.

[7] [Crawford et al., 2005] Crawford, T. J., Higham, S., Renvoize, T., Patel, J., Dale, M., Suriya, A., and Tetley, S. (2005). Inhibitory control of saccadic eye movements and cognitive impairment in Alzheimer's disease. *Biological Psychiatry*, 57(9):1052 – 1060.

[8] [Farias et al., 2009] Farias, S. T., Mungas, D., Reed, B. R., Harvey, D., and DeCarli, C. (2009). Progression of mild cognitive impairment to dementia in clinic- vs community-based cohorts. *Archives of neurology*, 66(9):1151–1157.

[1] [Fraser et al., 2017] Fraser, K. C., Fors, K. L., Kokkinakis, D., and Nordlund, A. (2017). An analysis of eye-movements during reading for the detection of mild cognitive impairment. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1016–1026.

[9] [Glynn et al., 2021] Glynn, K., O'Callaghan, M., Hannigan, O., Bruce, I., Gibb, M., Coen, R., Green, E., A Lawlor, B., and Robinson, D. (2021). Clinical utility of mild cognitive impairment subtypes and number of impaired cognitive domains at predicting progression to dementia: A 20-year retrospective study. *International Journal of Geriatric Psychiatry*, 36(1):31–37.

[10] [Groznik, 2020] Groznik, V. (2020). Digitalizacija in prilagoditev psihološkega testa za uporabo s sistemom za spremljanje očesnih gibov. *Uporabna informatika*, 28(4).

[11] [Hugo and Ganguli, 2014] Hugo, J. and Ganguli, M. (2014). Dementia and cognitive impairment: epidemio- logy, diagnosis, and treatment. *Clinics in geriatric medicine*, 30(3):421—442.

[12] [Letenneur, 2004] Letenneur, L. (2004). Risk of dementia and alcohol and wine consumption: a review of recent results. *Biological research*, 37(2):189–193.

[13] [Lourida et al., 2019] Lourida, I., Hannon, E., Littlejohns, T. J., Langa, K. M., Hyppönen, E., Kuz´ma, E., and Llewellyn, D. J. (2019). Association of lifestyle and genetic risk with incidence of dementia. *Jama*, 322(5):430–437.

[14] [Lourida et al., 2013] Lourida, I., Soni, M., Thompson-Coon, J., Purandare, N., Lang, I. A., Ukoumunne, O. C., and Llewellyn, D. J. (2013). Mediterranean diet, cognitive function, and dementia: a systematic review. *Epidemiology*, pages 479–489.

[15] [Lueck et al., 2000] Lueck, K. L., Mendez, M. F., and Perryman, K. M. (2000). Eye movement abnormalities during reading in patients with Alzheimer disease. *Neuropsychiatry, Neuropsychology, & Behavioral Neurology*.

[16] [Mioshi et al., 2006] Mioshi, E., Dawson, K., Mitchell, J., Arnold, R., and Hodges, J. R. (2006). The Adden- brooke's Cognitive Examination Revised (ACE-R): a brief cognitive test battery for dementia screening. *International Journal of Geriatric Psychiatry: A journal of the psychiatry of late life and allied sciences*, 21(11):1078–1085.

[17] [Slachevsky et al., 2004] Slachevsky, A., Villalpando, J. M., Sarazin, M., Hahn-Barma, V., Pillon, B., and Dubois, B. (2004). Frontal assessment battery and differential diagnosis of frontotemporal dementia and Alzheimer disease. *Archives of neurology*, 61(7):1104–1107.

[18] [Swiger et al., 2013] Swiger, K. J., Manalac, R. J., Blumenthal, R. S., Blaha, M. J., and Martin, S. S. (2013). Statins and cognition: a systematic review and meta-analysis of short-and long-term cognitive effects. In *Mayo clinic proceedings*, volume 88, pages 1213–1221. Elsevier.

[19] [Szekely et al., 2008] Szekely, C. A., Breitner, J. C., Fitzpatrick, A. L., Rea, T. D., Psaty, B. M., Kuller, L. H., and Zandi, P. P. (2008). Nsaid use and dementia risk in the cardiovascular health study*: Role of apoe and nsaid type. *Neurology*, 70(1):17–24.

[20] [Wilcockson et al., 2019] Wilcockson, T. D., Mardanbegi, D., Xia, B., Taylor, S., Sawyer, P., Gellersen, H. W., Leroi, I., Killick, R., and Crawford, T. J. (2019). Abnormalities of saccadic eye movements in dementia due to Alzheimer's disease and mild cognitive impairment. *Aging (Albany NY)*, 11(15):5389.

[21] [World Medical Association, 2013] World Medical Association (2013). World Medical Association Declara- tion of Helsinki: Ethical Principles for Medical Research Involving Human Subjects. *JAMA*, 310(20):2191– 2194.

▪

Vida Groznik je soustanoviteljica in direktorica podjetja NEUS Diagnostics, d.o.o. v okviru katerega razvijajo sisteme za pomoč pri oceni zdravstvenega stanja uporabnikov s pomočjo metod umetne inteligence. Poleg tega je docentka na Fakulteti za matematiko, naravoslovje in informacijske tehnologije na Univerzi na Primorskem ter raziskovalka v Laboratoriju za umetno inteligenco na Fakulteti za računalništvo in informatiko Univerze v Ljubljani. Doktorirala je leta 2018 z delom, ki je tesno povezovalo umetno inteligenco in nevrologijo. Je (so)avtorica več raziskovalnih člankov in poglavij v knjigah s področja umetne inteligence v medicini. Pridobila in delala je na različnih projektih financiranih s programov EU, Ministrstva za izobraževanje, znanost in šport ter Slovenskega podjetniškega sklada.

▪

Aleksander Sadikov je docent in predstojnik Laboratorija za umetno inteligenco na Fakulteti za računalništvo in informatiko Univerze v Ljubljani. Vodil je oz. vodi več evropskih in domačih projektov, ki so povezani z uporabo umetne inteligence v medicini. Na tem področju ima tudi precejšnje število objav v znanstvenih revijah in konferencah. Poleg tega je soustanovitelj podjetja NEUS Diagnostics, d.o.o., ki razvija sisteme za oceno zdravstvenega stanja uporabnikov s pomočjo metod umetne inteligence.

Kategorizacija uporabnikov na podlagi njihovega z informacijsko varnostjo povezanega znanja, stališčin vedenja: pilotna študija

Damjan Fujs[1], Simon Vrhovec[2], Damjan Vavpotič[1]
[1]Univerza v Ljubljani, Fakulteta za računalništvo in informatiko, Večna pot 113, 1000 Ljubljana 2Univerza V Mariboru, Fakulteta za varnostne vede, Kotnikova 8, 1000 Ljubljana poštni naslov ustanove damjan.fujs@fri.uni-lj.si, simon.vrhovec@um.si, damjan.vavpotic@fri.uni-lj.si

Izvleček

V tem prispevku predstavljamo pristop za kategorizacijo uporabnikov, ki temelji na preverjenem vprašalniku o človeških vidikih informacijske varnosti (HAIS-Q). Na tej podlagi smo izvedli razvrščanje uporabnikov (N = 165). Analiza je pokazala tri skupine uporabnikov (uporabniki z nizkim, zmernim in visokim tveganjem). Izračunali smo indeks silhuete (0,44) za potrditev kakovosti razvrščanja, ki kaže na ustrezno kakovost razvrščanja. Naš pristop (kombinacija HAIS-Q in razvrščanja v skupine) omogoča prilagojeno usposabljanje 1) uporabnikov, ki dosežejo najnižje vrednosti pri HAIS-Q in 2) uporabnikov, ki dosegajo splošno nizko tveganje, vendar se na nekaterih področjih odrežejo nekoliko slabše. V primerjavi s sorodnimi pristopi je naša prednost enostavnost uporabe in ublažitev pristranskosti, saj je pristop namenjen analiziranju skupin in ne posameznikov. Poleg tega naš pristop omogoča razvrščanje na podlagi prioritizacije spremenljivk, medtem ko obstoječe raziskave obravnavajo vse spremenljivke enako pomembne.

Ključne besede: kibernetska varnost, informacijska varnost, inženirstvo zahtev, segmentacija uporabnikov.

Abstract

In this paper, we present an approach for user categorization based on the established Human Aspects of Information Security Questionnaire (HAIS-Q). Clustering based on HAIS-Q data (N = 165) was performed. In doing so, three groups of users were identified (low, moderate and high-risk users). The silhouette measure of cohesion and separation (0.44) was conducted to validate the quality of clustering. A fair cluster quality was indicated. Our approach (a combination of HAIS-Q and clustering) allows for the tailored training of 1) users who achieve the lowest values in HAIS-Q and 2) users who achieve overall high results but perform slightly worse in certain focus areas. In comparison to similar approaches, our advantage is the ease of use and mitigation of social desirability bias since the approach is designed to analyze user groups only. Additionally, our approach allows for focus area (variable) prioritization while existing studies consider all variables to be of equal value.

Keywords: Cyber security, information security, requirements engineering, user segmentation

1 UVOD

Pomanjkanje ustrezne informacijske varnosti lahko vodi v izgubo dobička in slab ugled organizacije (Roy Sarkar, 2010). Da bi dobili vpogled v dejansko stanje informacijske varnosti moramo implementirati ustrezne merilne mehanizme (Prislan et al., 2020), kot so kvantitativne in kvalitativne metrike (Fujs et al., 2020, 2019). Informacijska varnost se je tradicionalno zagotavljala s pomočjo tehničnih varnostnih mehanizmov (kot so npr. požarni zidovi), kljub temu, pa ne smemo zanemariti pomembnosti človeških vidikov zagotavljanja varnosti (Wiley et al., 2020). Da bi uporabniki z informacijskimi sistemi ravnali varno, morajo biti ustrezno usposobljeni, pri čemer pa univerzalni pristopi niso optimalni, saj ne upoštevajo varnostnih karakteristik posameznega uporabnika (Fujs et al., 2020).

Namen pričujočega prispevka je odgovoriti na zastavljena raziskovalna vprašanja: 1) kako oceniti človeške vidike informacijske varnosti? 2) kako kategorizirati uporabnike na tej podlagi? 3) kako nam ti rezultati lahko pomagajo pri prilagojenem usposabljanju?

2 ČLOVEŠKI VIDIKI INFORMACIJSKE VARNOSTI

Varnostne funkcije programske opreme so bile tradicionalno uvedene na koncu procesa razvoja programske opreme, brez upoštevanja uporabnikovih - z varnostjo povezanih karakteristik (Wiley et al., 2020). Cˇ e se je pojavila ranljivost, so jih razvijalci naslovili z varnostnimi popravki (Niazi et al., 2020). Pravilno nastavljene tehnične rešitve so učinkovite, vendar ne zadostne, da bi zagotovile celovito informacijsko varnost, saj obstajajo razlike med uporabniki v njihovem znanju, stališčih in vedenju glede informacijske varnosti (Neigel et al., 2020). V ta namen je bil razvit vprašalnik, ki meri ozaveščenost o informacijski varnosti (v nadaljevanju HAIS-Q; angl. Human Aspects of Information Security Questionnaire). HAIS-Q obravnava uporabnikovo z varnostjo-povezano znanje, stališča do varnosti in varno vedenje (Parsons et al., 2017), ki se osredotoča na sledečih sedem področij: *USO* (uporaba socialnih omrežij), *UIN* (uporaba interneta), *POI* (poročanje o incidentih), *RZI* (ravnanje z informacijami), *UMN* (uporaba mobilnih naprav), *UEP* (uporaba e-pošte) in *UGE* (upravljanje gesel). Raziskave so pokazale, da je HAIS-Q dober napovedovalec človeških vidikov informacijske varnosti, saj uporabniki, ki so dosegali v povprečju višji indeks ozaveščenosti o informacijski varnosti, so bili uspešnejši v poskusih o zvabljanju (angl. *pshishing*) (Parsons et al., 2017). Podobno je bilo ugotovljeno v raziskavi o razlikah glede kibernetske higiene med moškimi in ženskami, kjer so s pomočjo regresijskih modelov ugotovili dobro napovedno moč posameznih dejavnikov oz. področij iz HAIS-Q (Neigel et al., 2020). HAIS-Q je bil uporabljen v še eni od študij glede razlik med demografskimi spremenljivkami (natančneje spol), kjer so ugotovili, da obstajajo statistično pomembne razlike med spoloma glede ozaveščenosti o informacijski varnosti (Wiley et al., 2020). Poleg ozaveščenosti o informacijski varnosti, imata pomembno vlogo tudi splošna organizacijska in varnostna kultura, kar pomeni, da če dvignemo splošno organizacijsko kulturo, dvignemo tudi varnostno kulturo (Wiley et al., 2020).

3 KATEGORIZACIJA UPORABNIKOV

Koncept kategorizacije (v literaturi poimenovan tudi kot *segmentacija*) je uporaben z vidika identifikacije različnih tipov uporabnikov. Beseda segmentacija se dandanes uporablja na področju računalniškega vida, vendar je bila prvotno uporabljena (in se še uporablja) na področju marketinga za identifikacijo različnih skupin uporabnikov Tolley (1975). V okviru našega pristopa bomo uporabili kategorizacijo uporabnikov, saj bomo na ta način identificirali različne uporabnike (npr. ranljive), na podlagi katerih bomo lahko predlagali prilagojena usposabljanja. Usposabljanja na področju informacijske varnosti so pomembna, saj težijo k njenemu izboljšanju (Neigel et al., 2020). V literaturi so poudarjeni številni pristopi za kategorizacijo uporabnikov na podlagi varnostnih karakteristik. Xiao (2021) kategorizira uporabnike pametnih mobilnih telefonov na podlagi spola ter tipa operacijskega sistema. Poleg tega, lahko kategorizacijo na podlagi spola zasledimo tudi v ostalih študijah, glej npr. (Wiley et al., 2020; Neigel et al., 2020; Parsons et al., 2017). Cain et al. (2018) raziskujejo vpliv starosti na poznavanje kibernetske higiene, pri čemer ugotavljajo, da obstoječi neprilagojeni pristopi za usposabljanje uporabnikov ne izboljšujejo znanja in vedenja uporabnikov. Anichiti et al. (2021) kategorizirajo uporabnike na podlagi njihovega znanja o kibernetski varnosti hotelov, pri čemer se za razliko od ostalih prej omenjenih avtorjev, osredotočajo na pripadnost generaciji - rojeni v različnih obdobjih (npr. generacija x, y in z). Poleg tega lahko v literaturi zasledimo kategorizacijo uporabnikov na podlagi vlog, ki jih imajo v organizacijah (Sarkar et al., 2020).

Optimalne skupine uporabnikov je nemogoče identificirati *a priori* brez podatkov, zato je pomembno, da izhajamo iz zbranih podatkov. V našem prispevku se osredotočamo na kategorizacijo uporabnikov na podlagi njihovih tveganj glede informacijske varnosti. Z vidika prilagojenih usposabljanj je treba imeti v mislih, da je manjše skupine lažje obvladovati. Namen gručenja na nivoju skupin je dvoplasten. Personalizirana usposabljanja so v realnem oziroma industrijskem okolju finančno breme, kar pomeni, da si velika večina tovrstnih usposabljanj ne more privoščiti. Na ta način iščemo razmerje med »realnimi« in »idealnimi« razmerami na področju informacijske varnosti. Zaradi tega je bolje, da izobražujemo uporabnike na nivoju skupin.

4 METODA

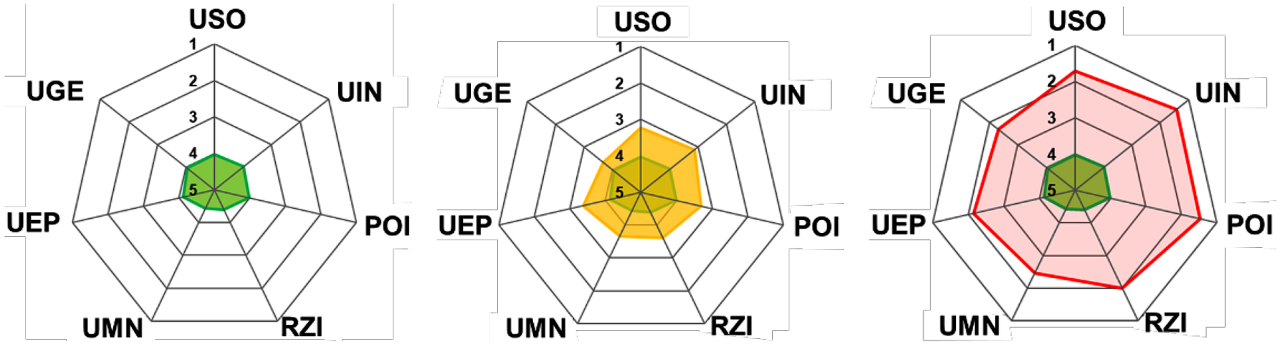
Za potrebe našega pristopa smo analizirali rezultate ankete 165 uporabnikov informacijskih sistemov ene izmed slovenskih univerz. V sklopu informacijskovarnostnih študij so študenti zelo primerna populacija za preučevanje informacijske varnosti (Aurigemma et al., 2019). Študenti so pogosti uporabniki informacijske tehnologije (mobilne naprave, aplikacije, socialna omrežja itd.) (Aurigemma et al., 2019) in so s tem tudi varnostno poudarjeni. Hkrati Hewitt in White (2020) med študenti opažata pomanjkanje zavedanja o informacijski varnosti, kar pa je značilno tudi za celotno populacijo uporabnikov informacijskih storitev (Falessi et al., 2018; Hameed and Arachchilage, 2021).V okviru našega pristopa smo prilagodili in prevedli HAIS-Q. HAIS-Q zajema 63 spremenljivk oziroma vprašanj, ki na Likertovi lestvici od 1 (se sploh ne strinjam) do 5 (se popolnoma strinjam) merijo ozaveščenost o informacijski varnosti. Raziskava je bila izvedena februarja 2021. Anketiranci so bili obveščeni, da je raziskava anonimna in prostovoljna. Vabilo k raziskavi je bilo poslano na 964 študentskih e-naslovov, pri čemer smo dosegli 17 % odziv. Zbrani podatki so bili kvalitativno pregledani. Glede na to, da ni bilo odstopanj, smo vse spremenljivke ohranili za nadaljnje analize. Podrobnejša demografska slika anketirancev je prikazana v Tabeli 1. 63 spremenljivk je bilo združenih v sedem faktorjev. Teh sedem faktorjev je bilo uporabljenih za potrebe razvrščanja v skupine (angl. *clustering*). Poleg tega smo izvedli tudi razvrstitev spremenljivk znotraj gruče glede na pomembnost, kar predstavlja dodano vrednost, saj obstoječi pristopi obravnavajo vse spremenljivke kot enako pomembne (Parsons et al., 2017). Za potrebe razvrščanja v skupine na podlagi HAIS-Q smo izbrali algoritem *TwoStep*, ki omogoča samodejno (ali vnaprej podano) identifikacijo primernega števila skupin (Tan, 2019). Dodatna prednost algoritma je iterativnost, saj lahko razvrščamo skupine, dokler identificiramo najbolj optimalno rešitev za preučevani primer. Za preverbo kakovosti razvrščanja smo izmerili indeks silhuete (angl. Silhouette index), ki lahko zavzame vrednost od -1 do 1, pri čemer 1 pomeni optimalno vrednost, saj so razdalje znotraj gruč kratke in razdalje med gručami velike (Celestino et al., 2018).

Tabela 1: **Demografske značilnosti vzorca na podlagi 165 anketirancev, pri čemer povprečna starost znaša 24 let.**

	Število #	Odstotek %
Vrsta študija		
Redni	136	83,0
Izredni	29	17,0
Stopnja in letnik študija		
1. letnik dodiplomskega študija	78	48,0
2. letnik dodiplomskega študija	18	11,0
3. letnik dodiplomskega študija	28	17,0
1. letnik podiplomskega študija	19	11,0
2. letnik podiplomskega študija	19	11,0
Doktorski študij	3	2,0
Spol		
Moški	48	29,0
Ženski	117	71,0
Življenjsko okolje		
Mestno	91	55,0
Podeželsko	74	45,0

5 REZULTATI IN RAZPRAVA

Slika 1 prikazuje polarni grafikon z identificiranimi skupinami na podlagi razvrščanja v skupine (TwoStep). Indeks silhuete znaša 0,44, kar pomeni, da je kakovost razvrščanja v skupine ustrezna. Iz Slike 1 lahko razberemo, da uporabniki z nizkim tveganjem dosegajo najboljše rezultate pri posameznih področjih HAIS-Q. Te skupine uporabnikov uporabimo za primerjavo med ostalima dvema skupinama. Upoštevajte, da je lestvica na polarnih grafikonih obrnjena, saj želimo prikazati odstopanje od normalnosti (torej od nizkega tveganja). Namen je identificirati stanje informacijske varnosti na sedmih področij od MSO do UGE ter nato vsakemu uporabniku, katerega znanje je na določenem področju pomanjkljivo, ponuditi namensko usposabljanje, da vrzeli s področja zapolnijo. Z vidika prilagojenega usposabljanja so najbolj zanimivi uporabniki z zmernim in/ali visokim tveganjem, saj najbolj odstopajo od normalnega oziroma zaželenega. Pristop je zasnovan na način, da se uporabnike izobražuje na tistih področjih, ki jih najslabše pokrivajo – oziroma tam kjer je odstopanje največje. Od podatkov oziroma gručenja pa je odvisno, v katero skupino bo posameznik uvrščen. Prednosti gručenja pred klasičnim skupinskim treningom sta predvsem finančna optimizacija in učinkovitejše usposabljanje saj vsakega uporabnika pošljemo na največ eno usposabljanje, pri čemer tovrstni način



a) Uporabniki z nizkim tveganjem (*N* = 137)

b) Uporabniki z zmernim tveganjem (*N* =24)

c) Uporabniki z visokim tveganjem (*N* =4)

Slika 1: **Primerjava skupin uporabnikov pri izbranih področjih človeških vidikov informacijske varnosti. Uporabniki z nizkim tveganjem (a) služijo kot merilo za primerjavo med ostalima dvema skupinama uporabnikov (b, c).**

zagotavlja tudi manj prekrivanj. Usposabljanja ki temeljijo na gručenju sicer niso tako prilagojena kot pri individualni obravnavi, vendar gre za nek kompromis med učinkovitostjo in stroški. Cilj izobraževanja mora biti doseganje nizkega tveganja. Da bi podrobneje razumeli dinamiko izbranih področij HAIS-Q znotraj gruče, predstavljamo Tabelo 2, v kateri izpostavimo pomembnost posameznega področja znotraj gruče.

Namen Tabele 2 je izpostaviti povprečne vrednosti izbranih področij HAIS-Q znotraj posamezne gruče. V primeru razvrstitev spremenljivk znotraj gruče glede na pomembnost gre za standardno meritev na skali od 0 do 1, ki se izračuna na podlagi statistične značilnosti posamezne spremenljivke znotraj gruče (Tkaczynski, 2016). Za lažji prikaz smo izbrali pet stopenj (naraščajoče z 0.2), pri čemer vrednosti manjše od 0.6 ni. Bolj, kot se vrednosti na barvni lestvici približujejo 1, bolj je spremenljivka statistično pomembna znotraj posamezne gruče. Na ta način omogočimo identifikacijo usposabljanja, ne samo za uporabnike z visokim tveganjem, ampak tudi za tiste, ki imajo splošno nizko tveganje, vendar na določenih področjih dosegajo odstopanje (npr. UIN in POI pri uporabniki z nizkim tveganjem).

V primerjavi s sorodnimi pristopi (Parsons et al., 2017), naš pristop omogoča 1) kategorizacijo uporabnikov na podlagi HAIS-Q spremenljivk in ne samo sodeč po demografskih karakteristikah, kot je npr. spol, izobrazba itd. S tem naslavljamo enega izmed pomembnih faktorjev pristranskosti pri evalvaciji informacijske varnosti (Wiley et al., 2020; Roy Sarkar, 2010), saj je znano, da je mogoče pristranskost znižati na način, da uporabnike vključimo v skupine (Kwak et al., 2019) ali pa da ne zbiramo vseh demografskih podatkov (Larson, 2019), če to res ni nujno. 2) prednost uporabe algoritma TwoStep je, da lahko vnaprej podamo število skupin in na podlagi parametrizacije najdemo optimalno število skupin ter pripadajočih enot/uporabnikov. S tem naslovimo optimizacijo stroškov usposabljanja, saj je manjše skupine lažje obvladovati in da s pomočjo prilagaja-

Tabela 2: **Razvrstitev spremenljivk znotraj gruče glede na pomembnost. Vrednosti v oklepajih predstavljajo povprečno vrednost posameznega področja informacijske varnosti.**

Uporabniki z nizkim tveganjem	Uporabniki z zmernim tveganjem	Uporabniki z visokim tveganjem	Legenda:
(N=137) 83.0 %	(N=24) 14.5 %	(N=4) 2.4 %	(barvna lestvica pomembnosti)
UIN (3.98)	USO (3.24)	UIN (1.44)	1.0
USO (4.03)	UMN (3.66)	USO (1.72)	0.8
POI (4.02)	UIN (3.12)	POI (1.47)	0.6
RZI (4.40)	UEP (3.37)	UMN (2.47)	0.4
UEP (4.13)	POI (3.31)	RZI (2.00)	0.2
UMN (4.43) UGE (4.05)	RZI (3.61) UGE (3.68)	UGE (2.31) UEP (2.14)	0.0

nja usposabljamo samo tiste skupine, ki jima manjka neka kompetenca.

V nadaljnjem delu bi nadgradili naš pristop s prilagajanjem varnostnih zahtev na podlagi kategorizacije, kar pomeni, da bi bilo smiselno razviti priporočilni sistem na podlagi katerega bi priporočali 1) tehnične varnostne funkcionalnosti ali 2) zahteve v smislu prilagojenih usposabljanj. Poleg tega pa bi nadgradili metodologijo za oceno rezultatov našega pristopa.

Kot vsaka raziskava, ima tudi pričujoči prispevek določene omejitve, ki jih je pri interpretaciji treba upoštevati. Prvič, vzorec študentov ni bil naključen, saj so anketiranci prihajali iz ene izmed slovenskih univerz. To pomeni, da ugotovitev ne moremo posploševati na vso populacijo študentov. Drugič, raziskavo bi bilo smiselno izvesti tudi med ostalo „ne-študentsko“populacijo (npr. zaposleni v podjetju). S tem bi dobili bolj celovit in raznovrsten vpogled v človeške vidike informacijske varnosti. Tretjič, bilo bi smiselno izvesti raziskavo na večjem vzorcu. Ne glede na to, pa velja omeniti, da je večina sorodnih raziskav narejena na vzorcih podobne velikosti (Tkaczynski, 2016; Parsons et al., 2014; Farooq et al., 2021; Hewitt and White, 2020; Cheolho et al., 2012)).

LITERATURA

[1] Anichiti, A., Dragolea, L. L., Hârs, an, G. D. T., Haller, A. P., and Butnaru, G. I. (2021). Aspects regarding safety and security in hotels: Romanian experience. *Information*, 12(1):1–22.

[2] Aurigemma, S., Mattson, T., and Leonard, L. (2019). Evaluating the Core and Full Protection Motivation Theory Nomologies for the Voluntary Adoption of Password Manager Applications. *AIS Transactions on Replication Research*, 5:1–21.

[3] Cain, A. A., Edwards, M. E., and Still, J. D. (2018). An exploratory study of cyber hygiene behaviors and knowledge. *Journal of Information Security and Applications*, 42:36–45.

[4] Celestino, A. E. M., Cruz, D. A. M., Sánchez, E. M. O., Reyes, F. G., and Soto, D. V. (2018). Groundwater quality assessment: An improved approach to K-means clustering, principal component analysis and spatial analysis: A case study. *Water (Switzerland)*, 10(4):1–21.

[5] Cheolho, Y., Hwang, J.-W., and Rosemary, K. (2012). Exploring Factors That Influence Students' Behaviors in Information Security. *Journal of Information Systems Education,* 23(4):407–415.

[6] Falessi, D., Juristo, N., Wohlin, C., Turhan, B., Münch, J., Jedlitschka, A., and Oivo, M. (2018). Empirical software engineering experts on the use of students and professionals in experiments. *Empirical Software Engineering*, 23(1):452–489.

[7] Farooq, A., Dubinina, A., Virtanen, S., and Isoaho, J. (2021). Understanding dynamics of initial trust and its antecedents in password managers adoption intention among young adults. *Procedia Computer Science*, 184:266–274.

[8] Fujs, D., Mihelič, A., and Vrhovec, S. L. R. (2019). The power of interpretation: Qualitative methods in cybersecurity research. In *Proceedings of the 14th International Conference on Availability, Reliability and Security*, pages 1–10, New York, NY, USA. ACM.

[9] Fujs, D., Vrhovec, S., and Vavpotič, D. (2020). Bibliometric mapping of research on user training for secure use of information systems. *Journal of Universal Computer Science,* 26(7):764–782.

[10] Hameed, M. A. and Arachchilage, N. A. G. (2021). The role of self-efficacy on the adoption of information systems security innovations: a meta-analysis assessment. *Personal and Ubiquitous Computing*.

[11] Hewitt, B. and White, G. L. (2020). Optimistic Bias and Exposure Affect Security Incidents on Home Computer. *Journal of Computer Information Systems*, 00(00):1–11.

[12] Kwak, D. H., Holtkamp, P., and Kim, S. S. (2019). Measuring and controlling social desirability bias: Applicati- ons in information systems research. *Journal of the Association for Information Systems*, 20(4):317–345.

[13] Larson, R. B. (2019). Controlling social desirability bias. *International Journal of Market Research*, 61(5):534– 547.

[14] Neigel, A. R., Claypoole, V. L., Waldfogle, G. E., Acharya, S., and Hancock, G. M. (2020). Holistic cyber hygiene education: Accounting for the human factors. *Computers and Security,* 92:101731.

[15] Niazi, M., Saeed, A. M., Alshayeb, M., Mahmood, S., and Zafar, S. (2020). A maturity model for secure requirements engineering. *Computers and Security*, 95:101852.

[16] Parsons, K., Calic, D., Pattinson, M., Butavicius, M., McCormac, A., and Zwaans, T. (2017). The Human Aspects of Information Security Questionnaire (HAIS-Q): Two further validation studies. *Computers and Security*, 66:40–51.

[17] Parsons, K., McCormac, A., Butavicius, M., Pattinson, M., and Jerram, C. (2014). Determining employee awareness using the Human Aspects of Information Security Questionnaire (HAIS-Q). *Computers and Security*, 42(May):165–176.

[18] Prislan, K., Mihelič, A., and Bernik, I. (2020). A real-world information security performance assessment using a multidimensional socio-technical approach. *PLoS ONE*, 15(9 September):1–28.

[19] Roy Sarkar, K. (2010). Assessing insider threats to information security using technical, behavioural and organisational measures. *Information Security Technical Report*, 15(3):112–133.

[20] Sarkar, S., Vance, A., Ramesh, B., Demestihas, M., and Wu, D. T. (2020). The influence of professional subculture on information security policy violations: A field study in a healthcare context. *Information Systems Research*, 31(4):1240–1259.

[21] Tan, S. C. (2019). Improving association rule mining using clustering-based discretization of numerical data. In *2018 International Conference on Intelligent and Innovative Computing Applications, ICONIC 2018*, pages 18–22, Mon Tresor, Mauritius. IEEE.

[22] Tkaczynski, A. (2016). Segmentation Using Two-Step Cluster Analysis. In Dietrich, T., Rundle-Thiele, S., and Kubacki, K., editors, *Segmentation in Social Marketing: Process, Methods and Application*, pages 109–125.

[23] Tolley, S. B. (1975). Identifying Users through a Segmentation Study. *Journal of Marketing*, 39(2):69–71.

[24] Wiley, A., McCormac, A., and Calic, D. (2020). More than the individual: Examining the relationship between culture and Information Security Awareness. *Computers and Security*, 88.

[25] Xiao, Q. (2021). Understanding the asymmetric perceptions of smartphone security from security feature perspective: A comparative study. *Telematics and Informatics*, 58 (May 2020):101535.

▪

Damjan Fujs je doktorski študent in asistent na Fakulteti za računalništvo in informatiko Univerze v Ljubljani. Njegova raziskovalna področja zajemajo varnost informacijskih sistemov (IS), metodologije razvoja IS, prilagojeno usposabljanje in izobraževanje za varno rabo IS ter kibernetsko in informacijsko varnost. Trenutno je član programskega odbora dveh konferenc: 1) Dnevi slovenske informatike (DSI 2021) in 2) The Sixth International Conference on Cyber-Technologies and Cyber-Systems (CYBER 2021).

▪

Simon Vrhovec je izredni profesor na Univerzi v Mariboru. Leta 2015 je doktoriral na Fakulteti za računalništvo in informatiko Univerze v Ljubljani. V letih 2018 in 2019 je sopred- sedoval mednarodni konferenci Central European Cybersecurity Conference(CECC). Od leta 2019 je član usmerjevalnega odbora European Interdisciplinary Cybersecurity Conference (EICC) ter član uredniškega odbora revije Journal of Cyber Security and Mobility. Njegova glavna raziskovalna področja so človeški dejavniki v kibernetski varnosti, razvoj varne programske opreme, agilne metode, odpor do sprememb in zdravstvena informatika

▪

Damjan Vavpotič je izredni profesor na Fakulteti za računalništvo in informatiko Univerze v Ljubljani. Njegovo raziskovalno delo vključuje področja metodologije razvoja programske opreme in njihovega sprejemanja vključno z razvojem varnih informacijskih sistemov, sprejemanje metod e-učenja ter napredne metode analize podatkov v zdravstvu in turizmu. Je član programskih odborov mednarodnih konferenc s področja računalništva in informatike. Objavil je več kot 50 člankov v revijah in na konferencah. Za leto 2019 je prejel nagrado TheaSinclair Award for Journal Article Excellence pri podjetju Sage Publishing.

PRILOGA (VPRAŠALNIK)

Tabela 3: Vprašalnik o človeških vidikih informacijske varnosti (HAIS-Q) v slovenskem jeziku, ki je povzet po Parsons et al. (2017). Simbol (*) nakazuje, da je merska lestvica obrnjena. Trditve so bile merjene na podlagi petstopenjske Likertove lestvice (od 1 - se sploh ne strinjam do 5 - se popolnoma strinjam). Pomen kratice indikatorja: _z (znanje), _s (stališče) in _v (vedenje).

Konstrukt	Indikator	Trditev
UGE	UGE1_z*	Sprejemljivo je, da uporabljam enaka gesla za moja socialna omrežja in za moje univerzitetne račune.
	UGE1_s*	Varno je uporabljati enaka gesla za socialna omrežja in za univerzitetne račune.
	UGE1_v	Uporabljam različna gesla za moja socialna omrežja in univerzitetne račune.
	UGE2_z*	Svoja univerzitetna gesla smem deliti s kolegi.
	UGE2_s	Ni dobro deliti mojih univerzitetnih gesel s kolegi, tudi če me prosijo za to.
	UGE2_v*	Svoja univerzitetna gesla delim s kolegi.
	UGE3_z	Univerzitetna gesla bi morala biti sestavljena iz črk, številk in simbolov.
	UGE3_s	Varno je imeti univerzitetno geslo, ki je sestavljeno samo iz črk.
	UGE3_v	Za univerzitetna gesla uporabljam kombinacijo črk, številk in simbolov.
UEP	UEP1_z*	Klikniti smem na katerokoli povezavo v e-pošti od ljudi, ki jih poznam.
	UEP1_s*	Vedno je varno klikniti na povezave v e-pošti od ljudi, ki jih poznam.
	UEP1_v	Na povezave v e-pošti ne klikam le zato, ker prihaja od nekoga, ki ga poznam.
	UEP2_z	Ne smem klikniti na povezavo v e-pošti, ki prihaja od neznanega pošiljatelja.
	UEP2_s*	Nič slabega se ne more zgoditi, če kliknem na povezavo v e-pošti, ki prihaja od neznanega pošiljatelja.
	UEP2_v*	Če e-pošta od neznanega pošiljatelja izgleda zanimivo, kliknem na vključeno povezavo.
	UEP3_z*	Priponke v e-pošti neznanih pošiljateljev smem odpreti.
	UEP3_s	Tvegano je odpreti priponko v e-pošti neznanega pošiljatelja.
	UEP3_v	Ne odpiram priponk v e-pošti, če mi pošiljatelj ni znan.
UIN	UIN1_z*	Na svoj delovni računalnik smem prenesti katerokoli datoteko, če mi to pomaga pri mojih nalogah.
	UIN1_s	Prenašanje datotek na moj delovni računalnik je lahko tvegano.
	UIN1_v*	Na svoj delovni računalnik prenašam katerekoli datoteke, ki mi pomagajo pri mojih nalogah.
	UIN2_z	Med delom naj nebi dostopal do določenih spletnih strani.
	UIN2_s	Le zato, ker lahko med delom dostopam do spletne strani, to še ne pomeni, da je varna.
	UIN2_v*	Ko med delom dostopam do interneta, obiščem katerokoli spletno stran želim.
	UIN3_z*	Svoje podatke lahko vnesem na katerokoli spletno stran, če mi to pomaga pri mojih nalogah.
	UIN3_s*	Če mi to pomaga pri mojih nalogah, ni pomembno, katere podatke vnesem na spletno stran.
	UIN3_v	Preden začnem vnašati podatke ocenim varnost spletnih strani.
USO	USO1_z	Nastavitve zasebnosti na mojih računih na socialnih omrežjih moram redno pregledovati.
	USO1_s	Nastavitve zasebnosti na mojih računih na socialnih omrežjih je dobro redno pregledovati.
	USO1_v*	Ne pregledujem redno nastavitev zasebnosti na svojih računih na socialnih omrežjih.
	USO2_z	Ne morejo me izključiti iz fakultete zaradi mojih objav na socialnih omrežjih.
	USO2_s*	Ni pomembno, če na socialnih omrežjih objavim nekaj, česar sicer ne bi javno izjavil.
	USO2_v	Na socialnih omrežjih ne objavim ničesar brez razmisleka o morebitnih negativnih posledicah.
	USO3_z*	Na socialnih omrežjih smem o svojem delu objaviti karkoli želim.
	USO3_s	Na socialnih omrežjih je tvegano objaviti določene informacije o mojem delu.
	USO3_v*	Na socialnih omrežjih objavljam karkoli želim o svojem delu.
UMN	UMN1_z	Pri delu na javnih krajih je treba imeti svoj prenosnik ves čas pri sebi.
	UMN1_s*	Pri delu v kavarni bi bilo varno pustiti svoj prenosnik za minuto brez nadzora.
	UMN1_v*	Pri delu na javnem kraju pustim svoj prenosnik brez nadzora.
	UMN2_z*	Občutljive z delom povezane datoteke smem pošiljati preko javnega omrežja Wi-Fi.
	UMN2_s	Občutljive z delom povezane datoteke je tvegano pošiljati čez javno omrežje Wi-Fi.
	UMN2_v*	Občutljive z delom povezane datoteke pošiljam čez javno omrežje Wi-Fi.
	UMN3_z	Pri delu z občutjivimi dokumenti moram zagotoviti, da neznanci ne morejo videti na zaslon mojega prenosnika.
	UMN3_s	Tvegano je dostopati do občutljivih z delom povezanih datotek na prenosniku, če lahko neznanci vidijo na moj zaslon.
	UMN3_v	Preverim, da neznanci ne morejo videti na zaslon mojega prenosnika, če delam z občutljivimi dokumenti.
RZI	RZI1_z*	Občutljive izpise na papirju lahko zavržemo na enak način kot neobčutljive.
	RZI1_s*	Občutljive izpise na papirju je varno zavreči z odlaganjem v koš za smeti.
	RZI1_v	Ko je treba zavreči občutljive izpise na papirju, poskrbim, da so razrezani ali uničeni.
	RZI2_z	Če na javnem kraju najdem USB ključek, ga ne smem priključiti v svoj delovni računalnik.
	RZI2_s*	Če na javnem kraju najdem USB ključek, se ne more zgoditi nič slabega, če ga priključim v svoj delovni računalnik.
	RZI2_v	V svoj delovni računalnik ne bi priključil na javnem mestu najdenega USB ključka.
	RZI3_z*	Občutljive izpise na papirju smem čez noč pustiti na svoji mizi.
	RZI3_s	Tvegano je pustiti občutljive izpise na papirju čez noč na moji mizi.
	RZI3_v*	Občutljive izpise na papirju puščam na moji mizi, ko me ni tam.
POI	POI1_z	Če opazim, da se nekdo v prostorih fakultete obnaša sumljivo, bi moral to prijaviti.
	POI1_s*	Če se ne menim za nekoga, ki se obnaša sumljivo v prostorih fakultete, se ne more zgoditi nič slabega.
	POI1_v	Če bi videl, da se nekdo obnaša sumljivo v prostorih fakultete, bi nekaj ukrenil glede tega.
	POI2_z	Ne smem zanemariti pomanjkljivega odnosa do varnosti s strani mojih kolegov.
	POI2_s*	Nič slabega se ne more zgoditi, če zanemarim pomanjkljiv odnos do varnosti, ki ga ima kolega.
	POI2_v*	Če bi opazil, da se moj kolega ne zmeni za varnostna pravila, ne bi ukrepal.
	POI3_z*	Poročanje o varnostnih incidentih je neobvezno.
	POI3_s	Tvegano je zanemariti varnostne incidente, tudi če se mi zdijo nepomembni.
	POI3_v	Če bi opazil varnostni incident, bi ga prijavil.

Delovni okvir za pretočne podatke s standardom common information model (CIM)

Maja Savinek[1,2], Matjaž Kukar[1]
[1] Univerza v Ljubljani, Fakulteta za računalništvo in informatiko, Večna pot 113, 1000 Ljubljana
[2] Elektro Ljubljana, Slovenska cesta 56, 1000 Ljubljana
maja.savinek@elektro-ljubljana.si, matjaz.kukar@fri.uni-lj.si

Izvleček

Napredno merjenje električne energije je tehnična rešitev za avtomatiziran obračun električne energije. Pametno omrežje je zgolj paradigma, kjer je električno omrežje z njegovimi uporabniki povezano z rešitvami informacijskih in komunikacijskih tehnologij za izboljšanje vseh funkcionalnosti omrežja. Za doseganje tega, je potrebno uporabiti razpoložljive podatke merilnih naprav, senzorjev, polnilnic in drugih naprav v omrežju. Vključevanje teh podatkov predstavlja številne izzive, saj niso vse naprave povezane s tradicionalnimi sistemi zajema podatkov.

Predlagani delovni okvir omogoča zajem podatkov v skoraj realnem času z ustreznim komunikacijskim standardom. Slednje se doseže s pretvorbo podatkov v skladu s standardi za merjenje električne energije na podlagi XSD sheme in transformacije podatkov v formatu CIM. Dognali smo, da s predlaganim delovnim okvirjem čas intervala od zajema, transformiranja do končne razpoložljivosti, je štirikrat učinkovitejši od obstoječih.

Ključne besede: Common information model, napredne merilne naprave, podatkovni tok, upravljanje podatkov

Abstract

Smart metering is a technical solution for emerging markets and billing system. The smart grid is a paradigm where the power grid and its users are connected to information and communication technology solutions, which envisage the improvement of all network functionalities. To achieve this, it is necessary to use the increasingly available data from smart meters, sensors, chargers and other devices. The incorporation of this data is associated with many challenges as not all devices are connected to a traditional data capture system.

The proposed framework enables near real-time data capture with appropriate communication standards. This is achieved by converting values in accordance with the standards for electricity measurement based on the XSD scheme and values transformed into a CIM model. We have found that the usual length of the time interval from capturing the current data on the measuring device to recording the raw current is four times more efficient than the existing frameworks.

Keywords: Common information model, data management, smart meter, stream data

1 UVOD

Elektroenergetska podjetja se soočajo s pospešeno digitalizacijo in posledično z iskanjem novih rešitev za obvladovanje obnovljivih virov energije, dinamičnega razvoja energetskih trgov, energetskih storitev ter iskanjem naprednih pristopov upravljanja omrežja. Nekateri izzivi, ki so povezani z obvladljivostjo energije, so stohastično vedenje proizvodnje obnovljivih virov energije, spremembe smeri pretoka energije v omrežjih ali proizvodnja energije v različnih obdobjih. Prisotnost pametnih števcev v nizkonapetostnem omrežju omogoča povezavo končnih deležnikov z omrežjem in sočasno generiranje podatkov. Kljub naprednim tehnologijam in pospešenemu uvajanju naprednih merilnih naprav, predstavlja enovito povezovanje med naprednimi merilnimi sistemi in pametnim omrežjem velik izziv.

V večini primerov napredne merilne naprave komunicirajo preko PLC (Power Line Communication) ali GPRS/3G. Običajni protokol za odčitavanje merilnih podatkov je DLMS/COSEM (standard za izmenjavo merilnih podatkov o električni energiji). Podatki kot so napetost, tok, delovna moč, jalova energiji se lahko generirajo skoraj v realnem času. Ti podatki predstavljajo velik potencial za izvedbo napredne analitike in poglobljen pogled v distribucijsko omrežje ter predstavljajo enega od pomembnih vhodov za sistema ADMS (Advanced Distribution Management Systems) in SCADA (Supervisory Control And Data Acquisition), ki sta poglavitna sistema za upravljanje in nadziranje distribucijskega omrežja.

Iz različnih razlogov kot so zagotavljanje podatkov v standardni obliki za izmenjavo podatkov, nezanesljivosti komunikacije v realnem času z naprednimi merilnimi napravami, zagotavljanje kakovosti masovnih podatkov in izvajanje imputacije manjkajočih vrednosti, smo raziskavo usmerili v razvoj delovnega okvirja, ki omogoča zajem, transformacijo in imputacijo s komunikacijskim standardom za izmenjavo paketnih sporočil v skoraj realnem času.

2 SORODNO DELO

Nekateri avtorji so raziskali področje zajemanja podatkov pametnih števcev in svoje delo objavili v prispevkih. Pri pregledu del smo izpostavili ključne točke, ki jih bomo upoštevali pri razvoju delovnega okvirja za zajem in obdelavo merilnih pretočnih podatkov. Avtorji člankov se problematike niso lotili kot celovitega problema, ampak zgolj posameznih področij.

2.1 Delovni okvir za zajem pretočnih podatkov

Koncept velepodatkov je prisoten že več kot desetletje. Kljub temu, da je dobro opredeljen in da na splošno razumemo njegove prednosti, podjetja ne morejo učinkovito izkoristiti podatkov in vseh prednosti, ki jih ta koncept ponuja. Tradicionalne metode upravljanja podatkov, kot so relacijske podatkovne baze, niso sposobne zajeti večletnih, sekundnih in milisekundnih dogodkov, kar vodi do problematike zagotavljanja informacij o dejanskem stanju pametnih omrežij v realnem času [1].

V prispevku [2] avtorji opisujejo problem zajemanja podatkov, ki se generirajo na robu omrežja. Vključevanje novih podatkov predstavlja veliko izzivov, saj niso vse naprave povezane s komunikacijskim omrežjem. Glavna ideja, ki jo obravnavajo avtorji, je paralelno združiti različne pretočne podatke z zgodovinskimi podatki. Z idejo o virtualnem sistemu SCADA želijo združiti različne formate podatkov na standardiziran vir.

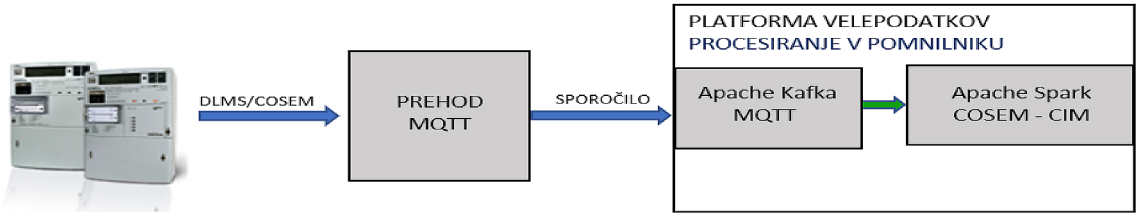
2.2 Izmenjava pretočnih podatkov v formatu CIM

V prispevku avtorji izpostavljajo težave, s katerimi se morajo spoprijeti pri prehodu iz distribucijskega in prenosnega omrežja na pametna omrežja. Pametna omrežja se bodo v prihodnosti soočila s standardizacijo standardnih protokolov, če želijo v celoti izkoristiti ponujene prednosti in funkcionalnosti. Nacionalni inštitut za standarde in tehnologijo (NIST) [4] je opredelil 16 ključnih standardnih protokolov [5] za pametna omrežja, nato dopolnil na 77. Interoperabilnost standardov je mogoče doseči z ontološkim konceptom semantične piramide. To dosežemo z informacijskim modelom, ki združuje definicije, koncepte in povezave med njimi. Prav tako je definiran UML podatkovni model električnega omrežja za CIM (Common Information Model), ki je razširjen na ustrezne protokole in kombiniran z ontologijami drugih protokolov za doseganje interoperabilnosti. V prispevku je poudarek na standardu IEC 60870 [6], kjer v večini primerov izvajajo preslikavo podatkov le za doseganje interoperabilnosti. Predstavili so pristop uporabe SPARQL za merilne naprave, ki ustrezajo formatu CIM.

3 DELOVNI OKVIR ZA PRETOČNE PODATKE

Osnovna dilema predlaganega delovnega okvira je, ali je mogoče vzpostaviti delovni okvir, ki bo v skoraj realnem času zajemala neobdelane podatke, jih prenese v platformo za obvladovanje velepodatkov, kjer se neobdelani podatki v pomnilniku sprocesirajo, imputirajo in transformirajo v format CIM. Delovni okvir vsebuje več komponent, ki so ponazorjene s shematskim prikazom na sliki 1, torej napredne merilne naprave, sistem zajema podatkov (prehod) in procesiranje podatkov v pomnilniku.

Napredne merilne naprave so opremljene s komunikacijskim modulom, ki v skladu s standardom DLMS / COSEM komunicira s sporočilnim sistemom MQTT. MQTT je protokol za prenos podatkov IoT (internet stvari). Na desni strani slike 1 je prikazana obdelava podatkov v pomnilniku na velepodatkovni platformi, ki je sestavljena iz zajemanja pretočnih podatkov s komponento Apache Kafka MQTT in raz-



Slika 1: **Semantična predstavitev komponent predlaganega okvira za zajem in obdelavo podatkov v pomnilniku s COSEM-CIM razčlenjevalnikom v formatu CIM**

členjevalnikom Apache Spark COSEM-CIM. Apache Kafka je prek protokola MQTT povezan s prehodom, kjer pridobi neobdelane pretočne podatke in skrbi za prenos teh podatkov v Apache Spark za procesiranje v pomnilniku. Ko se podatki nahajajo v Apache Spark, se v pomnilniku izvede obdelava z razčlenjevalnikom COSEM-CIM.

Razčlenjevalnik COSEM-CIM vsebuje metodo za pretvorbo neobdelanih podatkov v strukturirane podatke, matematični algoritem za preoblikovanje ter združevanje vrednosti, algoritem za imputacijo manjkajočih vrednosti in komponento za pretvorbo v format CIM. Tako obdelani podatki se v zadnjem koraku zapišejo v platformo za obvladovanje velepodatkov. Razčlenjevalnik COSEM-CIM uporablja standardizirano obliko podatkov CIM XML v standardu IEC61968 [3].

3.1 Opredelitev testnega okolja

Za potrebe vrednotenja delovanja predlaganega delovnega okvirja pretočnih podatkov (COSEM-CIM parser) se je vzpostavilo testno okolje, ki je vsebovalo 50 naprednih merilnih naprav MT880, opremljenih s komunikacijskim modulom UMTS, MQTT sporočilno infrastrukturo ter platformo za upravljanje velepodatkov in procesiranje v pomnilniku. Celotno okolje je delovalo na operacijskem sistemu Linux CentOS z 64 GB RAM, 16 vCPU in 1TB diskovnega prostora.

Testno okolje AMI je vsebovalo tri ključne komponente in sicer 50 naprednih merilnih naprav MT880, sistem za upravljanje merilnih podatkov MDMS (Meter Data Management System) in relacijsko podatkovno bazo. Okolje je delovalo na operacijskem sistemu Windows server 2016 in SQL strežniku z enako konfiguracijo strojne opreme kot delovni okvir COSEM-CIM parser.

RDF SPARQL okolje je delovalo na enaki infrastrukturi kot COSEM-CIM parser. Odstopanje je bilo v konfiguraciji hranjenja in obdelovanja podatkov v Apache Spark.

Delovanje testnih okolij na različnih operacijskih sistemih bi lahko privedlo do odstopanj v hitrosti delovanja posameznih komponent. Faktor operacijskih sistemov bi bil zelo pomemben pri izvajanju meritev v milisekundah. Izvedba naših meritev je temeljila na najnižji časovni enoti sekundi, ki je zavedena v sistemskih zapisih. Na podlagi teh dejstev smo faktor vpliva operacijskih sistemov zanemarili.

4 REZULTATI IN RAZPRAVA

V tem razdelku predstavljamo primerjavo med okvirji, in sicer AMI (Advance Metering Infrastructure), RDF SPARQL in razčlenjevalnikom COSEM-CIM. Izhodiščna vprašanja raziskovanja so bila:

- primerjava zagotavljanja podatkov v realnem času in
- vpliv parametrov na čas obdelave in pretvorbo podatkov v realnem času v formatu CIM.

Vsaka napredna merilna naprava vsebuje frekvenco prenosa podatkov in registre merilnih vrednosti toka (I) in napetosti (U). Frekvenca intervalov zajema podatkov je dosegla največ 1 minuto. Glavno vprašanje je, ali je mogoče vzpostaviti infrastrukturo, ki bo sposobna v realnem času zajemati neobdelane podatke in jih prenašati v velepodatkovno platformo, kjer se bodo podatki v pomnilniku obdelali in transformirali v format CIM. Test je bil zasnovan tako, da je upošteval natančen čas zajema za vseh 50 števcev točno takrat, ko so se v posredniku pojavili neobdelani podatki, čas obdelave in pretvorbe ter čas ponovne poizvedbe. Testiranje se je ponovilo v 100 iteracijah. Izračunani povprečni čas iz sistemskih zapisov zajema podatkov od posrednika do ponovne poizvedbe, omogoča oceno dejanskega doprinosa obdelave in transformacije podatkov. S standardnim odklonom smo izmerili razpršenost izračunanih povprečnih časov okoli aritmetične sredine vzorcev.

Tabela 1: **V tabelarični obliki so prikazani rezultati testiranja časovne primerjave posameznih delovnih okvirjev od zajema do transformacije podatkov. Spremenljivka Stream definira tip pretočnosti podatka. Vrednost t_1 prikazuje čas zajema podatkov, t_2 prikazuje čas obdelave in transformacije v format CIM, t_3 čas, ko so podatki ponovno na voljo v formatu CIM. Skupni čas obdelave je zaveden v stolpcu t_s.**

Delovni okvir	Pretočnost podatkov	t_1	σ_1	t_2	σ_2	t_3	σ_3	t_s	σ_s
AMI	ne	6,27 min	0,35	–	–	–	–	–	–
RDF SPARQL	da	0,518 s	0,18	5,527 s	0,23	5,98 s	0,20	12,61 s	0,21
COSEM - CIM	da	0,492 s	0,13	2,648 s	0,19	0 s	–	3,14 s	0,17

Tabela 1 vsebuje primerjavo rezultatov testiranja zajem in transformiranja podatki v skoraj realnem času s tremi različnimi pristopi.

Prvi delovni okvir AMI, zasnovan z inženirskega vidika, ni omogočal zajema podatkov v skoraj realnem času, kar je privedlo do časa zajema podatkov 6,27 min s standardnim odklonom 0,35. Drugi delovni okvir RDF SPARQL, zasnovan z znanstvenega vidika, omogoča zajem podatkov v realnem času. Povprečen čas zajema podatkov je 0,518 s s standardnim odklonom 0,18. V tretjem predlaganem okvirju COSEM-CIM parser je povprečni čas zajema podatkov 0,492 s standardnim odklonom 0,13.

V naslednjem koraku testiranja smo preverili koliko časa izbrana delovna okvirja RDF SPARQL in COSEM-CIM potrebujeta za obdelavo ter transformacijo podatkov v format CIM in omogočanje razpoložljivosti podatkov za nadaljnjo poizvedovanje. RDF SPARQL je potreboval enak povprečni čas za zajem neobdelanih podatkov kot COSEM-CIM parser. V naslednjem koraku, kjer se podatki obdelajo in transformirajo, je COSEM-CIM parser porabil manj časa kot RDF SPARQL, saj slednji neobdelane podatke obdela, transformira, shrani in jih na zahtevo pretvori v CIM format. COSEM CIM parser podatke paralelno obdela in hkrati transformira. Predlagani okvir je štirikrat učinkovitejši od RDF SPARQL, kar je razvidno v tabeli 1, torej COSEM CIM parser porabi 3,14 s s standardnim odklonom 0,17 in RDF SPARQL 12,61 s s standardnim odklonom 0,21. Prednost našega predlaganega okvirja je, da se vsi matematični algoritmi in transformacije izvedejo paralelno v pomnilniku.

4.1 Razprava

Na podlagi izvedenih meritev smo prišli do zaključka, da je s predlaganim delovnim okvirjem izvedljivo zajeti in transformirati pretočne podatke v format CIM v realnem času.

Pri raziskavi in definiranju delovnega okvirja smo se uprli na ugotovitve v sorodnih raziskavah, predvsem na članek [4], kjer so avtorji uporabili hranjenje neobdelanih podatkov in dostop v RDF SPARQL-u. Slabost RDF SPARQL pristopa je povpraševanje po podatkih, saj je časovno zelo potratno. S pristopom COSEM-CIM se podatki zajamejo, vzporedno obdelajo, transformirajo pomnilniku in shranijo v formatu CIM, kar omogoča štirikrat učinkovitejše delovanje predlaganega delovnega okvira.

5 ZAKLJUČEK

Raziskovalno delo je usmerjeno v razvoj delovnega okvirja za zajem, obdelavo in procesiranje v pomnilniku pretočnih podatkov v skoraj realnem času. Naš cilj je ponuditi obdelane in standardizirane podatke za shranjevanje ali integracijo z različnimi aplikacijami v realnem okolju.

Ugotovitve temeljijo na rezultatih testiranj in primerjave z drugimi okvirji, ki so pokazale, da predlagan okvir zmanjša čas celotne obdelave za štirikrat. Pri pregledu znanstvenih del s področja predlaganega okvirja ni bilo zaznati, da bi raziskana literatura obravnavala področje kot celoto, ki je predstavljeno v našem predlaganem okvirju.

Na podlagi pregleda znanstvenih člankov smo nekatere rešitve uporabili, jih nadgradili ter jih ponudili kot izboljšane celote. Za testiranje okvirja smo uporabili tehnologijo, komponente in strojno opremo, kar ne pomeni, da je bila optimalno izbrana.

REFERENCE

[1] Tabue, B., et al.: Proactive transmission and distribution asset management (2016).
[2] Avo Sevlian, R., et al.: Visualization and Analytics for Distributed Energy Resources, Cornell University (2017).
[3] Simmons, J.: Common Information Model Primer, EPRI (2015).
[4] Crapo, A., et al: The semantically enabled smart grid, in Proceedings of the Grid-Interop Forum, pp. 177-185 (2009).

[5] NIST Framework and Roadmap for Smart Grid Interoperability Standards, 1st ed., National Institute of Standards and Technology (2009).

[6] Penya, K., et al: Semantic integration of IEC 60870 into CIM, IEEE (2011).

[7] Rohr, M., et al: Using CIM for Smart Grid ICT integration, BTC, pp. 45-61 (2011).

▪

Maja Savinek je doktorska študentka na Fakulteti za računalništvo in informatiko Univerze v Ljubljani. Zaposlena je v podjetju Elektro Ljubljana, d. d., kjer se ukvarja z razvojem napredne analitike in algoritmov na področju elektroenergetike, s poudarkom na tehnologijah velepodatkov.

▪

Matjaž Kukar je docent na Fakulteti za računalništvo in informatiko, član Laboratorija za kognitivno modeliranje. Leta 2001 je doktoriral iz računalništva in informatike, leta 1996 magistriral in leta 1993 diplomiral na Fakulteti za računalništvo in informatiko Univerze v Ljubljani. Njegove raziskave so na področju umetne inteligence, predvsem strojnega učenja, podatkovnega rudarjenja in analize podatkov.

Pozitivno in neoznačeno učenje z generativnimi nasprotniškimi mrežami

Aleš Papič, Igor Kononenko, Zoran Bosnić
Univerza v Ljubljani, Fakulteta za računalništvo in informatiko, Večna pot 113, 1000 Ljubljana, Slovenija
{ales.papic, igor.kononenko, zoran.bosnic}@fri.uni-lj.si

Izvleček

Količina ustvarjenih podatkov otežuje njihovo obdelavo. V primeru nadzorovanega učenja lahko označevanje učnih primerov predstavlja dolgotrajno in drago nalogo. V dvorazrednih problemih primere označimo kot pozitivne ali negativne. V tem delu predpostavljamo, da imamo na voljo majhno število pozitivnih primerov in večje število neoznačenih primerov. Cilj generativnega pozitivnega in neoznačenega učenja je ustvariti označene primere, kar predstavlja strategijo za reševanje tega problema. Kljub temu generativni pristopi prinašajo pomanjkljivosti, kot so visoka računska zahtevnost, nestabilno učenje in nezmožnost ustvarjanja popolnoma označenih podatkovnih množic. V prispevku predlagamo nov generativni pristop, ki temelji na pomožnih klasifikacijskih generativnih nasprotniških mrežah. Nenegativno pozitivno in neoznačeno tveganje integriramo kot pomožno funkcijo izgube, da se naučimo porazdelitve pozitivnih in negativnih primerov. Na priznanem naboru podatkov za pozitivno in neoznačeno učenje prikažemo najsodobnejše rezultate. Rezultati kažejo, da naš pristop dosega primerljivo točnost z obstoječimi pristopi, kljub preprostejši arhitekturi, ima visoko učno stabilnost in generira nabor označenih podatkov.

Ključne besede: pozitivno in neoznačeno učenje, delno nadzorovano učenje, generativne nasprotniške mreže, globoko učenje

Abstract

The quantity of the data generated makes them difficult to process. In the case of supervised learning, labelling training examples may represent an especially tedious and costly task. In binary classification problems, examples are labelled as positive or negative. In this work, we assume that we have at our disposal a small number of positive and a larger number of unlabelled examples. Generative positive and unlabelled learning aims to generate labelled data, which constitutes a strategy to address this issue. Nonetheless, the generative approaches bring shortcomings, such as high computational cost, training instability and the inability to generate fully labelled datasets. We propose a novel generative approach based on an auxiliary classifier generative adversarial network. We integrate non-negative positive and unlabelled risk as an auxiliary loss to learn the distribution of positive and negative examples. We demonstrate the state-of-the-art performance on a common positive and unlabelled learning benchmark dataset. The results show that our approach achieves comparable performance as existing approaches despite its simple architecture, has high training stability and generates fully labelled data.

Keywords: Positive and unlabelled learning, partially supervised learning, generative adversarial networks, deep learning

1 UVOD

Naloge nadzorovanega učenja zahtevajo, da so učni primeri označeni, *npr.* kot pripadniki pozitivnega ali negativnega razreda, pred učenjem modela. Vsakodnevno pridobivanje označenih podatkov ni vedno samoumevno zaradi pomanjkanja človeških strokovnjakov, s tem povezanih stroškov, časovnih omejitev ali celo omejitev postopka zajemanja podatkov. Zaradi omenjenih praktičnih pomanjkljivosti je postalo podpodročje delno nadzorovanega učenja, pozitivno in neoznačeno (*angl.* positive and unlabeled, PU) učenje, vse bolj proučevano.

PU učenje je binaren, pozitiven in negativen klasifikacijski problem, kjer so na voljo le označeni pozitivni in neoznačeni primeri. Elkan in Noto [6] sta neoznačene podatke obravnavala kot utežene pozitivne in negativne hkrati, kar je pozneje spodbudilo razvoj nepristranskih tehnik [4, 3, 8]. S tem predobdelava neoznačenih podatkov ni potrebna, kar poenostavi učni proces. Ne glede na to z omejenim

Tabela 1: **Primerjava prednosti predstavljenih najsodobnejših PU pristopov. Oznaka (✓) poudarja, da metoda izpolnjuje dano merilo na levi [1].**

Metoda	GenPU	PGAN	D-GAN
Predznanje ni potrebno		✓	✓
Primeren za zapletene nabore podatkov		✓	✓
Ustvari ustrezne pozitivne primere	✓		
Ustvari ustrezne negativne primere	✓		✓
Stabilnost učenja z uporabo SGD		✓	✓
Izvirna arhitektura GAN		✓	✓

naborom označenih podatkov obstaja nevarnost prevelikega prileganja, kadar se uporabljajo prilagodljivi modeli, kot so globoke nevronske mreže. Zato se v zadnjem času raziskave osredotočajo na uporabo generativnih nasprotniških mrež (*angl.* generative adversarial networks, GAN) zaradi rasti razpoložljivih neoznačenih podatkov [7, 2, 1]. GANi bogatijo učne primere z umetnimi, da je na voljo večji nabor označenih primerov, ki jih lahko uporabimo za nadzorovano učenje. Kljub temu imajo generativni pristopi PU učenja veliko računsko zahtevnost oziroma generirajo izključno negativne učne primere. To pa lahko predstavlja težavo med učenjem klasifikatorja v primerih, ko primanjkuje označenih pozitivnih primerov.

V ta namen predlagamo novo generativno PU ogrodje za generiranje označenih učnih primerov. Ogrodje uporablja pogojno latentno predstavitev prostora in je opremljeno s pomožno ocenitveno funkcijo za učenje iz PU podatkov. Poleg tega je predlagani pristop mogoče uporabiti na različnih znanih arhitekturah GAN. S poskusi na resničnih podatkih pokažemo, da naš pristop uspešno prepozna porazdelitev tako pozitivnih kot negativnih primerov.

2 SORODNA DELA

Du Plessis in sod. [3] so definirali konveksno PU učenje (uPU), toda Kiryo in sod. [8] so kasneje pokazali, da se prilagodljivi modeli, kot so nevronske mreže, pretirano prilegajo podatkom. To se zgodi, ko vrednost funkcije izgube postane negativna, zato so predlagali nenegativno različico (nnPU). Vendar pa uPU in nnPU zahtevata predhodno poznavanje porazdelitve podatkov. Chiaroni in sod. [1] so poudarili, da se porazdelitev primerov razlikuje med paketi (*angl.* batch), zaradi česar lahko stohastične tehnike optimizacije postanejo nestabilne, zlasti pri majhnih paketih. Zato se nedavne raziskave osredotočajo na generativne nasprotniške mreže (GANe), da bi podatke obogatili z umetnimi učnimi primeri in tako povečali označen učni nabor.

Mirza in sod. [10] so predlagali strategijo pogojnega GANa (CGAN), ki generatorju in diskriminatorju poda oznako razreda ter tvori z razredom pogojene umetne primere. Odena in sod. [11] so predlagali drugo različico, imenovano *pomožni klasifikator* GAN (AC-GAN), kjer ima diskriminator pomožno nalogo, da napove pogojno latentno informacijo. Skozi nasprotniški proces učenja se generator nauči skupne latentne predstavitve. Kljub temu oba pristopa potrebujeta označen nabor podatkov, česar pri PU učenju nimamo in predstavlja njuno glavno pomanjkljivost.

Velike količine neoznačenih podatkov otežujejo učni proces obstoječih PU metod. Zato so Hou in sod. [7] predlagali prvo generativno PU ogrodje (GenPU), ki vključuje vrsto generatorjev in diskriminatorjev, zaradi česar je računsko zahteven. Chiaroni in sod. [2] so predlagali Pozitiven-GAN (PGAN), ki temelji na originalni arhitekturi GAN. Zaradi pomanjkanja nadzora pa se ne more naučiti resnične negativne porazdelitve podatkov, kar so kasneje naslovili z metodo Divergentni-GAN (D-GAN) [1]. Ne glede na to je še vedno omejen na generiranje izključno negativnih primerov. To pa lahko predstavlja težavo pri učenju končnega klasifikatorja, ko je na voljo majhen nabor označenih primerov. Tabela 1 povzema ključne lastnosti generativnih PU pristopov.

3 POGOJNO GENERATIVNO PU UČENJE

Naše delo naslavlja omejitvi ogrodja GenPU, visoko računsko zahtevnost in nestabilnost učenja, ki sta posledici zapletene arhitekture. Izvirno GAN arhitekturo dopolnimo z dodatnim pomožnim klasifikatorjem v novem pogojnem generativnem PU ogrodju. Le-ta je zmožen generiranja primerov obeh razredov, kar pa ne velja za obstoječa PGAN in D-GAN. Omenjena lastnost je potrebna za probleme, pri katerih nam primanjkuje označenih pozitivnih primerov.

Naš pristop je zgrajen na obstoječem AC-GANu, ki lahko generira označen nabor podatkov. Pomožni klasifikator optimizira log-verjetje pravilnega razreda L_C [11], zato potrebuje označene primere iz

vsakega razreda. Da pa bi se lahko učili iz PU nabora, uvedemo nenegativno PU tveganje (L_{nnPU}) [8] kot dodatno pomožno funkcijo izgube. L_{nnPU} omogoča oceno izgube za neznane negativne primere, ki se nahajajo v vhodnih podatkih. Funkcijo izgube pomožnega klasifikatorja tako sestavlja utežena vsota $\lambda_1 L_C + \lambda_2 L_{nnPU}$, kjer je $\lambda_1 = 1 - \lambda_2$. L_C merimo na umetnih primerih, saj so njihove oznake na voljo v latentni predstavitvi, medtem ko L_{nnPU} merimo izključno na učni množici. Pri empiričnem ovrednotenju modela je parameter $\lambda_1 = 0.5$.

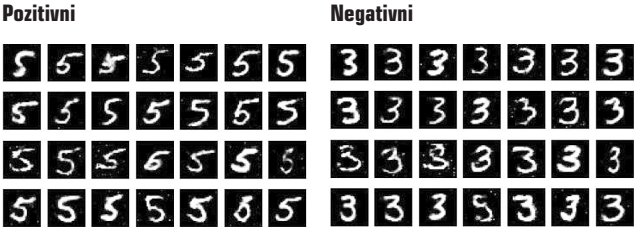
Cilj pomožnega klasifikatorja je usmeriti generator k pravilni uporabi podane pogojne informacije. Ne glede na to moramo vedeti, kakšna je porazdelitev neoznačenih podatkov, kar lahko ocenimo neposredno iz podatkov samih [5]. Ogrodje nam omogoča, da generiramo popolnoma označen nabor pozitivnih in negativnih primerov, iz pozitivnih in neoznačenih. Predlagan pristop je dvostopenjski, kar pomeni, da se najprej naučimo generirati umetne primere, le-te pa uporabimo za učenje poljubnega obstoječega binarnega klasifikatorja. Ob tem ima naš pristop manj računskih omejitev in večjo stabilnost.

4 REZULTATI IN RAZPRAVA

Učinkovitost generativnih pristopov pokažemo z empiričnim ovrednotenjem na podatkovni množici MNIST [9]. Med vrednotenjem se domneva, da je znana porazdelitev podatkov (π_p). Izbrali smo dve števki s po 5000 slikami, ki smo jih uporabili za izdelavo nabora podatkov PU. Da bi bil problem bolj zahteven, smo izbrali števki 3 in 5 zaradi njune vizualne podobnosti. Izvedli smo štiri ponovitve, vsako z različnim številom označenih primerov, in ocenili uspešnost binarnega klasifikatorja, naučenega na umetnih primerih s 3-kratnim prečnim preverjanjem (glej Tabelo 2). Za oceno učinkovitosti je bil uporabljen binarni klasifikator z dvema skritima plastema, vsaka z 256 nevroni. Aktivacijska funkcija za vsako plastjo je ReLU, razen zadnje, kjer je uporabljen sigmoid. Splošna arhitektura GAN in učni parametri so bili vzeti iz literature [7].

Rezultati kažejo, da klasifikator, naučen na podatkih predlaganega pristopa, dosega primerljivo točnost kot tisti, ki je bil naučen na podatkih PGANa in D-GANa. Na naše presenečenje, ko je na voljo le deset primerov z oznako, PGAN deluje najbolje. Hou in sod. [7] so ugotovili, da se nnPU začne pretirano prilegati, ob manjšem številu označenih primerov, zato se obnaša nestabilno. V našem ogrodju se je pojavila enaka nestabilnost, kar je vplivalo na generator, ki mu ni uspelo pravilno uporabiti podane pogojne informacije.

Že s 50 označenimi primeri pa je naš pristop presegel obstoječe. GenPU se je izkazal najslabše zaradi nestabilnega učenja in posledično nezmožnosti se naučiti porazdelitve negativnih primerov. Slika 1 prikazuje generirane števke s predlaganim pristopom.



Slika 1: **Vizualizacija umetnih pozitivnih in umetnih negativnih števk, ustvarjenih s predlaganim pristopom.**

Tabela 2: **Klasifikacijska točnost klasifikatorja na nalogi razlikovanja števk, 3 in 5 ob različnem številu označenih podatkov.**

PODATKI			**METODA**			
N_p	N_u	π_p	**PGAN**	**D-GAN**	**GenPU**	**Pog. gen. PU učenje (naš)**
1000	9000	0.444	**0.93 ± 0.01**	**0.93 ± 0.01**	0.47 ± 0.00	0.92 ± 0.00
100	9900	0.495	0.80 ± 0.06	0.82 ± 0.04	0.47 ± 0.00	**0.86 ± 0.04**
50	9950	0.497	0.72 ± 0.08	0.65 ± 0.13	0.47 ± 0.00	**0.74 ± 0.03**
10	9990	0.499	**0.69 ± 0.03**	0.61 ± 0.07	0.46 ± 0.00	0.59 ± 0.05

5 ZAKLJUČEK

Količina ustvarjenih podatkov se dnevno povečuje, kar otežuje njihovo obdelavo. Označevanje učnih primerov predstavlja še posebej dolgotrajno in drago, včasih tudi nemogočo nalogo. V članku predstavimo novo generativno ogrodje, ki zmanjšuje računsko zahtevnost in posledično povečuje učno stabilnost. Ob tem je primeren za širok nabor obstoječih arhitektur GAN in daje označen nabor podatkov. Rezultati so pokazali, da obstoječe metode največkrat dosežejo nižjo točnost kot predlagan pristop, še posebej, ko je število označenih primerov omejeno. Vendar pa je naš pristop pokazal težave pri zelo majhnem številu označenih primerov. To je posledica čezmernega prileganja pomožnega klasifikatorja, kar bomo obravnavali v prihodnje. Ne glede na to je predlagani pristop s sestavljeno kriterijsko funkcijo preprostejši, saj nadomešča kompleksno arhitekturo ogrodja GenPU.

Predlagan pristop je primeren za številne probleme v medicini, *npr.* pri odkrivanju redkih bolezni, v priporočil- nih sistemih za prepoznavanje zavajajočih ocen in v bančništvu za zaznavanje rizičnih posojil. Pomaga lahko tudi pri anonimizaciji občutljivih podatkov, kjer je mogoče nadomestiti vhodne podatke z umetnimi pri tem pa jih loči v za nas bolj (pozitivno) in manj (negativno) zanimivo skupino.

V prihodnje nameravamo pristop uporabiti za odkrivanje nenapovedanih obrokov pri bolnikih s sladkorno boleznijo tipa 1, saj je problem primeren za PU učenje. Sedanja umetna trebušna slinavka nima te zmožnosti, a bi obstoj takšnega sistema zelo pripomogel bolnikom.

ZAHVALA

To delo podpira Javna agencija za raziskovalno dejavnost Republike Slovenije (ARRS), sredstva za mlade raziskovalce (53630).

LITERATURA

[1] Florent Chiaroni, Ghazaleh Khodabandelou, Mohamed-Cherif Rahal, Nicolas Hueber, and Frédéric Dufaux. Generating relevant counter-examples from a positive unlabeled dataset for image classification. *arXiv preprint arXiv:1910.01968*, 2019.

[2] Florent Chiaroni, Mohamed-Cherif Rahal, Nicolas Hueber, and Frédéric Dufaux. Learning with a generative adversarial network from a positive unlabeled dataset for image classification. In *IEEE ICIP*, pages 1368–1372. IEEE, 2018.

[3] Marthinus Du Plessis, Gang Niu, and Masashi Sugiyama. Convex formulation for learning from positive and unlabeled data. In *ICML*, pages 1386–1394, 2015.

[4] Marthinus C Du Plessis, Gang Niu, and Masashi Sugiyama. Analysis of learning from positive and unlabeled data. In *NIPS*, pages 703–711, 2014.

[5] Marthinus Christoffel Du Plessis and Masashi Sugiyama. Class prior estimation from positive and unlabeled data. *IEICE*, 97(5):1358–1362, 2014.

[6] Charles Elkan and Keith Noto. Learning classifiers from only positive and unlabeled data. In *KDD'08*, pages 213–220, 2008.

[7] Ming Hou, Brahim Chaib-Draa, Chao Li, and Qibin Zhao. Generative adversarial positive-unlabeled learning. In *IJCAI*, pages 2255–2261. AAAI Press, 2018.

[8] Ryuichi Kiryo, Gang Niu, Marthinus C Du Plessis, and Masashi Sugiyama. Positive-unlabeled learning with non-negative risk estimator. In *NIPS*, pages 1675–1685, 2017.

[9] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[10] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[11] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *ICML*, pages 2642–2651. Proceedings of Machine Learning Research, 2017.

▪

Aleš Papič je mladi raziskovalec in doktorski študent na Fakulteti za računalništvo in informatiko, Univerze v Ljubljani. Njegovo raziskovalno področje vključuje strojno učenje, globoko učenje in generativne nasprotniške mreže. Prav tako je asistent pri predmetu Osnove umetne inteligence.

▪

Igor Kononenko je doktor računalniških znanosti in redni profesor na Fakulteti za računalništvo in informatiko Univerze v Ljubljani ter predstojnik Laboratorija za kognitivno modeliranje. Njegova raziskovalna področja so umetna inteligenca, strojno učenje, nevronske mreže in kognitivno modeliranje. Je (so)avtor 225 člankov na teh področjih ter 13 učbenikov (dve knjigi izšli v Angliji).

▪

Zoran Bosnić je profesor na Fakulteti za računalništvo in informatiko Univerze v Ljubljani. Raziskovalno se ukvarja z umetno inteligenco, zlasti s strojnim učenjem. Osredotoča se pretežno na učenje iz podatkovnih to- kov in na interdisciplinarne aplikacije strojnega učenja. Na tem področju je tudi (so)avtor okoli 70 znanstvenih člankov.

Iz Islovarja

Islovar je spletni terminološki slovar informatike, ki ga objavlja jezikovna sekcija Slovenskega društva INFORMATIKA na naslovu http://www.islovar.org. Slovar je javno dostopen za vpoglede in vnašanje novih izrazov.

aktívna napráva -e -e ž *(angl. active device)* naprava, ki za delovanje potrebuje poseben vir energije; prim. pasivna naprava

datotéka napráve -e -- ž *(angl. device file)* vmesnik za dostop do vhodno-izhodnih naprav, ki v datotečnem sistemu izgleda kot datoteka; sin. vmesniška datoteka

inêrcijska merílna napráva -e -e -e ž *(angl. inertial measurement unit, IMU)* naprava za določanje položaja, orientacije in hitrosti objekta, pri čemer navadno uporablja kombinacijo pospeškometra, žiroskopa in včasih magnetometra; prim. inercijski navigacijski sistem

kazálna napráva -e -e ž *(angl. tracking device)* naprava za upravljanje premikanja kazalca na računalniškem zaslonu

naménska napráva -e -e ž *(angl. appliance)* naprava, namenjena za določena opravila

navídezna napráva -e -e ž *(angl. pseudo device, virtual device)* programska koda, ki simulira delovanje fizične naprave, gonilnika, krmilnika; sin. virtualna naprava

navídezni računálnik -ega -a m *(angl. VM, virtual machine, virtual computer)* programska oprema, ki posnema delovanje stvarnega računalnika; sin. navidezna naprava; prim. posnemovalnik, simulator

navíti -ijem dov. *(angl. overclock)* s tehničnimi prilagoditvami povečati takt naprave nad tistega, za katerega je bil projektiran

níčelna napráva -e -e ž *(angl. null device)* datoteka, ki se izbriše ob sporočilu, da je ukaz izvršen

omréžje pomnílniških napráv -a -- -- *(angl. storage area network, SAN)* pomnilniške naprave in strežniki, povezani v funkcionalno celoto za hrambo podatkov

osnôvna številka napráve -e -e -- ž *(angl. major device number)* številka naprave, ki določa gonilnike za krmiljenje naprave

pasívna napráva -e -e ž *(angl. passive device)* naprava, ki za delovanje ne potrebuje posebnega vira energije; prim. aktivna naprava

pomóžna številka napráve -e -e -e ž *(angl. minor device number)* številka ene od naprav, ki jo upravlja gonilnik

profíl mobílne napráve -a -- -- -- m *(angl. mobile information device profile, MIDP)* nabor javanskih programskih vmesnikov za mobilne naprave

prostórska merílna napráva -e -e -e ž *(angl. coordinate measuring machine, CMM)* naprava za merjenje geometrijskih značilnosti predmeta

róbna napráva -e -e ž *(angl. edge device)* naprava, ki omogoča dostop do omrežja; prim. jedrno omrežje

sledílna napráva -e -e ž *(angl. tracking device)* naprava za določitev ali sledenje položaja ali gibanja na daljavo

Izpitni centri ECDL

ECDL (European Computer Driving License), ki ga v Sloveniji imenujemo evropsko računalniško spričevalo, je standardni program usposabljanja uporabnikov, ki da zaposlenim potrebno znanje za delo s standardnimi računalniškimi programi na informatiziranem delovnem mestu, delodajalcem pa pomeni dokazilo o usposobljenosti. V Evropi je za uvajanje, usposabljanje in nadzor izvajanja ECDL pooblaščena ustanova ECDL Fundation, v Sloveniji pa je kot član CEPIS (Council of European Professional Informatics) to pravico pridobilo Slovensko društvo INFORMATIKA. V državah Evropske unije so pri uvajanju ECDL močno angažirane srednje in visoke šole, aktivni pa so tudi različni vladni resorji. Posebno pomembno je, da velja spričevalo v 148 državah, ki so vključene v program ECDL. Doslej je bilo v svetu izdanih že več kot 11,6 milijona indeksov, v Sloveniji več kot 17.000, in podeljenih več kot 11.000 spričeval. Za izpitne centre v Sloveniji je usposobljenih osem organizacij, katerih logotipe objavljamo.

AGORA

Micro Team

Znanstveni prispevki

Kratki znanstveni prispevki

Informacije

ISSN 1318-1882
9 771318 188001