Alati za suradničko učenje programiranja

Igor Škorić
Financijska agencija, podružnica Pula
Giardini 5, Pula
skoric.igor@gmail.com

Sažetak **Učenje programiranja predstavlja težak zadatak za studente, jer od njih zahtjeva usvajanje mnogih vještina. Osim sintakse programskog jezika oni moraju usvojiti i algoritamski način rješavanja problema, razumjeti način na koji računao radi, imati predodžbu o tome kako će se program izvršavati itd. Tako kompleksan problem može predstavljati prevelik teret za početnika i djelovati na njega obeshrabrujuće, a posljedica toga je niska prolaznost, velik broj odustanaka i zaziranje studenata od programiranja. Problemi vezani uz učenje programiranja potaknuli su brojna istraživanja koja različitim intervencijama nastoje poboljšati taj proces. Kolaborativno učenje jedna je od najpopularnijih metodologija poučavanja i često se koristi i kod poučavanja programiranja. Da bi primjena bilo kakve intervencije bila što uspješnija važno je pri tome koristiti pravi alat koji će potencirati njene pozitivne efekte.**

U ovom radu dan je pregled relevantne literature i istraživanja o korištenju suradničkog učenja pri učenju programiranja i alatima koji se pri tome koriste. Prikazani su i komentirani su rezultati tih istraživanja i prezentirane smjernice za daljnja istraživanja.

Ključne riječi - učenje programiranja; suradničko učenje; kolaborativni alati

I. UVOD

Usvajanje vještine programiranja jedna je od temeljnih kompetencija koje mora usvojiti svaki student koji svoju profesionalnu budućnost vidi u informatici. Nažalost, istraživanja pokazuju da ovladavanje programiranjem predstavlja težak zadatak za studente [1]. Kolegiji vezani uz programiranje imaju nisku prolaznost i visoku stopu odustajanja [2][1]. Ta prolaznost je niska bez obzira na brojna istraživanja koja pokušavaju naći rješenje tog problema [3]. Razlozi tome su brojni [4][5]. Programiranje sadrži kompleksne apstraktne koncepte [6]. Sintaksa programskih jezika je kompliciranа i nije prilagođena edukaciji [5]. Studenti nemaju razvijen učinkovit mentalni model računala [7]. I ako ovladaju sintaksom pojedinih naredbi studenti ih ne znaju ukomponirati u smisleni program [8]. Karakteristike studenata isto značajnu utječu na njihov uspjeh u učenju programiranja [9], pa neki od njih mogu imati problem s motivacijom ili zaziru od programiranja. Osim sintakse programskog jezika oni moraju usvojiti i algoritamski način rješavanja problema, razumjeti način na koji računalo radi, imati predodžbu o tome kako će se program izvršavati itd. Ne postoji univerzalni odgovor na sve ove probleme, ali se kroz istraživanja pokušavaju naći rješenja koja mogu odgovoriti na bar neka od njih primjenom različitih strategija poučavanja i kreiranjem novih alata prilagođenih tim strategijama s ciljem poboljšanja njihova učinka.

Neke od istraživanih strategija pokazuju potencijal da poboljšaju rezultate poučavanja programiranja. Porter i ostali [10] kao primjer takvih uspješnih pristupa izdvajaju programiranje u paru, vršnjačko poučavanje i medijsko računarstvo. Oni navode rezultate koji pokazuju da ova tri pristupa pomažu studentima u učenju i zadržavanju znanja i da se njihovim kombiniranjem efekt pojačava. Vihavainen, Airaksinen i Watson [11] u svom sustavnom pregledu pristupa učenju programiranja proučavaju efekte korištenja trinaest različitih pristupa poučavanju programiranja. Na osnovu analize primarnih studija zaključuju da pedagoške intervencije poboljšavaju rezultate učenja programiranja. Nakon što su grupirali intervencije u pet grupa (u koje su stavljene srodne intervencije) pokazuje se da najveće poboljšanje rezultata učenja pokazuje grupa u kojoj su intervencije zasnovane na suradničkom učenju (kooperativno učenje, učenje bazirano na timu i programiranje u paru).

Možemo zaključiti da strategije učenja koje uključuju suradnju kroz grupni rad imaju potencijal da olakšaju i poboljšaju poučavanje programiranja.

II. SURADNIČKO UČENJE

Suradničko (kolaborativno) učenje "je situacija u kojoj dvoje ili više ljudi uče ili pokušavaju učiti zajedno" [12]. Za ovakvo grupno učenje često se u literaturi pojavljuju dva imena: suradničko (kolaborativno) učenje i kooperativno učenje. Neki autori ne prave razliku među ovim nazivima, a neki ih jasno razlikuju. Panitz [13] definira kolaboraciju kao " filozofiju interakcije i osobnog načina života u kojem su pojedinci odgovorni za svoje postupke, uključujući učenje i poštovanje sposobnosti i doprinos njihovih vršnjaka“, a kooperaciju kao "strukturu interakcije osmišljenu kako bi se olakšalo postizanje određenog krajnjeg proizvoda ili cilja kroz ljude koji rade zajedno u grupama". Slavin [14] temeljnu razliku vidi u tome što je kooperacija od strane predavača definirana i strukturirana aktivnost s točno određenim ciljem. Neki autori [15] ukazuju na to da su kolaborativno i kooperativno učenje izvorno razvijeni za ljude različite dobi, iskustava i nivoa međusobne ovisnosti. Međutim oba tipa učenja imaju puno više sličnosti nego razlika [16] i zato ih brojni autori poistovjećuju.

Razlog zašto se u ovom radu ne pravi razlika između kolaborativnog i kooperativnog učenja je taj što se iste pedagoške tehnike (npr. programiranje u paru) odnosno isti alati za podršku grupnom učenju (npr. kolaborativni editori) mogu upotrijebiti i u kolaborativnom i u

kooperativnom učenju. Razliku čini način na koji strukturiramo tu aktivnost, pa je onda smatramo kolaborativnom ili kooperativnom. Kako ovaj rad obrađuje sve vrste grupnog učenja programiranja i za jedan i za drugi oblik u ovom se radu koristi naziv suradničko učenje.

Suradničko učenje se temelji na konstruktivizmu - znanje grade i transformiraju studenti. Suradničko učenje koristi se socijalnim procesima koji podupiru učenike na svom putu prema samostalnom rješavanju problema. Korijen suradničkog učenja je u teoriji učenja koju je razvio Lev Vygotsky . On razvija koncept "zona proksimalnog razvoja" koje su ključne za razvoj složenih vještina. Prema toj teoriji složeni koncepti (poput vještine programiranja) se razvijaju kroz zajedničke socijalne aktivnosti, prije nego što se razviju u vještine koje se mogu samostalno primijeniti.

Suradničko učenje se danas uspješno primjenjuje u širokom rasponu predmeta uključujući i učenje programiranja [17].

III. SURADNIČKO UČENJE PROGRAMIRANJA

Problemi u učenju programiranja i nedostatak informatičara na tržištu rada stvaraju potrebu da se proces njihovog školovanja oblikuje što efikasnije. S tim ciljem u obrazovni proces uvode se različite strategije i pedagoške intervencije s ciljem njegovog poboljšanja. Zbog toga je suradničko učenje kao uspješna tehnika u drugim područjima učenja uvedena i u učenje programiranja.

Kako je kolaborativno učenje u biti bilo kakvo učenje koje uključuje dva ili više sudionika u tom procesu ono obuhvaća mnoštvo različitih (imenovanih i bezimenih) tehnika s brojnim varijacijama. Nemoguće je u jednom članku obraditi i spomenuti svaku od njih, pa ćemo se ograničiti samo na neke (češće spominjane u literaturi).

Najpoznatija i najčešće korištena kolaborativna tehnika u učenju programiranja je programiranje u paru. Programiranje u paru je jedan od temelja metodologije ekstremnog programiranja čiji je kreator Beck [18]. Ona je nastala kao odgovor na krizu razvoja softvera korištenjem klasičnih razvojnih metodologija, s ciljem ubrzanja razvoja softvera kroz iterativni proces. Kod programiranja u paru dvije osobe razvijaju program koristeći jedno računalo. Jedna osoba je "driver" i on oblikuje i unosi kod. Druga osoba je "navigator" i on kontrolira oblikovanje programa i ispravlja greške u njemu. Uloge se izmjenjuju u pravilnim vremenskim razmacima [19]. Rastom popularnosti agilnih metoda ovakav način programiranja danas je široko raširen. U svojoj meta-analizi učinkovitosti programiranja u paru, Hannay i ostali [20] zaključuju da ovakav način programiranja u odnosu na klasičan način donosi poboljšanja u kvaliteti koda, brzini i uloženom trudu. Ovi rezultati mogu varirati u ovisnosti o različitim moderatorima (npr. kompleksnost zadatka).

Iz industrije, ova tehnika programiranja vrlo brzo se proširila i u obrazovni kontekst. Istraživanja o njenom korištenju u učenju programiranja pokazuju da iako ne pokazuje značajno poboljšanje u ocjenama studenata na finalnom ispitu [21], ipak pokazuje brojne pozitivne efekte u kvaliteti koda [22], učinkovitosti [23], zadovoljstvu [24], brzini [25], te ne donosi nikakav štetan efekt u odnosu na pojedinačno programiranje [21]. Možemo zaključiti da je programiranje u paru djelotvorni pedagoški alat u učenju programiranja[26].

U učenju programiranja često se koriste i različite tehnike grupnog programiranja. Primjer za to je Coding Dojo. To je tehnika grupnog učenja kod koje se grupa sastoji od dvojice studenata koji programiraju u paru i publike. U pravilnim vremenskim razmacima jedan član para odlazi u publiku, a jedan član publike postaje dio para. Svaki sudionik na kraju bar jednom isproba obje uloge u paru. Istraživanja pokazuju da studenti ovu tehniku doživljavaju kao zabavnu [27], opuštajuću [28], koja pomaže u učenju programiranja baziranog na testiranju [29].

Istraživana je i primjena tehnike studijskog učenja (Studio-based learning - SBL) u učenju programiranja. U ovoj tehnici studenti najprije pojedinačno rješavaju zadan problem, a onda grupno diskutiraju i analiziraju ta pojedinačna rješenja. Provedena istraživanja pokazuju da ova tehnika pozitivno utječe na motivaciju studenata [30] i njihov angažman [31].

U kontekstu učenja programiranja korištena je i tehnika suradničkog podučavanja(engl. peer instruction) . To je pedagoška tehnika u kojoj studenti najprije pojedinačno odgovaraju na pitanja, zatim diskutiraju s kolegama o odgovorima dok ne postignu grupni konsenzus, a zatim opet odgovaraju. Pokazalo se da studenti imaju pozitivan stav prema tehnici i da poboljšava razumijevanje koncepata iz programiranja [32], te studenti na testiranju imaju bolje rezultate iz gradiva koje su učili na taj način [33].

Pedagoški pregledi koda temelje se na grupnom pregledu i diskusiji o programu koji je napisao svaki od članova grupe. Nakon pregleda na osnovu povratnih informacija o svom radu student može prepraviti svoj program[34].

Izokrenuta nastava (engl. flipped classroom) je strategija poučavanja kod koje je proces učenja okrenut naopačke. Umjesto da student dolazi na nastavu kako bi slušao predavanje, on ga gleda u video obliku prije nastave. Student na nastavu mora doći pripremljen, da bi tamo rješavao probleme u interakciji s profesorom i ostalim studentima. U slučaju učenja programiranja na nastavi se obično pri rješavanju problema koristi računalo.

Programiranje uživo (engl. live coding) je praksa kod koje predavač uživo piše program pred učenicima [35]. To nije linearni proces već on tijekom tog pisanja komentira kod, ispravlja ga i prepravlja, objašnjava strategije. S druge strane studenti tijekom te aktivnosti postavljaju pitanja i sudjeluju u diskusiji. Isto ime koristi se i za tehniku koja spada pod medijsko računarstvo gdje studenti grupno programiraju manipulirajući zvukovima, te kao rezultat proizvode muziku.

Neke od navedenih tehnika povezuju se u literaturi sa POGIL (engl. Process Oriented Guided Inquiry Learning) strategijom [36] podučavanja koja se koristi u različitim STEM disciplinama, a fokusira se na simultani razvoj znanja i procesnih vještina (npr. rješavanje problema, timski rad). Kod ovog pristupa studenti rade u timovima

na strukturiranoj aktivnosti koja je specifično oblikovana da ih vodi prema osobnoj konstrukciji njihovog razumijevanja ključnih koncepata i u isto vrijeme im omogućava da razviju važne procesne vještine (kritičko razmišljanje, komunikacija...). Organiziran i projekt koji se bavi primjenom POGIL-a u edukaciji o računalstvu (http://cspogil.org).

Nisu sve strategije kolaborativnog učenja privukle jednaku pažnju istraživača. Programiranje u paru najčešće je korištena i najviše istraživana tehnika suradničkog učenja u programiranju. Iako postoji određeni prostor za daljnja istraživanja i u programiranju u paru licem u lice (npr. kako faktori kompatibilnosti između članova para utječu na učinkovitost [21]) najveći prostor za daljnja istraživanja nalazi se u području distribuiranog programiranja u paru [37]. Kod ostalih tehnika opseg istraživanja povezan je s raširenošću njihove primjene. Za sve opisane tehnike istraživanja bilježe neke pozitivne efekte pri korištenju u učenju programiranja. Kako su povezane s grupnim radom (koji je po prirodi kompleksan fenomen) radovi pokazuju potrebu da se njihova implementacija pažljivo oblikuje, uzimajući u obzir brojne faktore koji utječu na uspješno oblikovanje grupe i njenu dinamiku. Važan element uspješne primjene su i alati koji će se pri tome koristiti. Pravilno odabran (ili kreiran) alat može bitno doprinijeti pozitivnim efektima pri procesu učenja, zato je velik dio istraživanja o učenju programiranja posvećen alatima koji se pri tome koriste.

IV. ALATI ZA UČENJE PROGRAMIRANJA

Alate koji se koriste u učenju programiranja možemo podijeliti na dvije grupe. Prvu grupu čine alati koji su kreirani za razvoj softvera u poslovnom okruženju (kao Visual Studio ili NetBeans). Oni nisu posebno kreirani za edukaciju, ali ovladavanje takvim alatom olakšava studentu početak rada u profesionalnoj karijeri. Drugu grupu čine alati koji su kreirani posebno za edukaciju s ciljem da na neki način olakšaju studentu učenje programiranja.

A. Integrirano razvojno okruženje i njegova evolucija

Osnovni alat svakog programera je integrirano razvojno okruženje (engl. integrated development environment - IDE). Različita integrirana razvojna okruženja imaju različite mogućnosti i različite setove alata, ali minimalno sadrže bar editor, prevoditelj (engl. compiler) i program za ispravljanje pogrešaka (engl, debugger). Programeru je skup alata koji sadrži integrirano razvojno okruženje dovoljan da samostalno razvije softver.

Međutim razvoj softvera u većini slučajeva nije poduhvat pojedinca, već je to grupna aktivnost u kojoj sudjeluje cijeli razvojni tim sastavljen od ljudi različitih specijalnosti. S ciljem što uspješnijeg razvoja, članovi tima moraju komunicirati, usklađivati se i surađivati da bi kreirali kvalitetan proizvod u što kraćem roku. Tijekom tog razvoja oni razmjenjuju pitanja i odgovore, ideje, planove i kod. Osim toga, danas nije neuobičajeno da članovi tima nisu na jednoj lokaciji, pa govorimo o distribuiranom razvoju softvera, kod kojeg je izazov ostvarivanja uspješne suradnje među članovima tima još teži. Suradnja postaje ključni element pri razvoju softvera, jer kolaboracija u razvojnom timu bitno utječe na uspješnost projekta [38] i članovi tima značajan dio vremena troše na nju.

Da bi se ta kolaboracija među članovima razvojnog tima što više olakšala koriste se brojni alati. Neki od tih alata su uobičajeni, široko prihvaćeni komunikacijski alati kao elektronska pošta (e-mail) ili razmjena izravnih poruka (engl. instant messaging - IM), dok su drugi alati nastali upravo kao podrška razvoju softvera kao što je sustav nadzora inačica (engl. version control systems - VCS). Vremenom, alati za kolaboraciju i komunikaciju počeli su se ugrađivati u integrirana razvojna okruženja kao što je Eclipse [39] ili Visual Studio [40]. Integracija kolaborativnih alata s razvojnim alatom omogućuje programeru da ostane cijelo vrijeme u istom kognitivnom kontekstu (jer ne napušta razvojnu okolinu) i čini razvoj i kolaboraciju jednom cjelinom. Takve alate koji su napravljeni s ciljem da poboljšaju učinkovitost cijelog razvojnog tima (za razliku od IDE koji služi poboljšanju učinkovitosti pojedinca) nazivamo kolaborativne razvojne okoline [41] (engl. collaborative development environment - CDE). Zadnja velika promjena koja se dešava u IT industriji je premještanje s dektopa na web. Pojava Weba 2.0 s Ajax tehnologijom, uvođenje HTML-a 5 i napredak u brzini JavaScript strojeva omogućilo je nastanak nove platforme. Web okolina nudi brojne prednosti u odnosu na desktop: jedinstveno sučelje dostupnost uvijek i s bilo kojeg mjesta, trenutnu kolaboraciju, laku integraciju s drugim servisima, eliminira potrebu za instalacijom, konfiguriranjem i održavanjem softvera [42]. Stoga gotovo svaki desktop softver danas ima i svoj web ekvivalent. Alati za razvoj softvera slijede taj isti trend. Danas možemo pronaći brojne razvojne alate kao gotove web servise. Taj proces transformacije još nije gotov i najveća industrijska razvojna sučelja (npr. Visual Studio) još uvijek se baziraju na desktopu (ali s brojnim ugrađenim kolaboracijskim alatima koji se oslanjaju na web). Ipak, i danas postoje kompletna integrirana razvojna sučelja bazirana na webu kao Codeanywhere (https://codeanywhere.com/) ili Cloud9 (https://c9.io/). Ta sučelja nemaju sve mogućnosti svojih desktop ekvivalenata ali ipak omogućavaju kompletan razvoj softvera na webu. Ovaj trend dodavanja kolaborativnih elemenata i postepeni prelazak prema web platformi prate i svi ostali alati za programiranje.

B. Alati za učenje programiranja

Postoji mnoštvo različitih alata namjenski stvorenih za učenje programiranja. Kelleher i Pausch su predložili taksonomiju [43] takvih alata koja ih dijeli na jedanaest skupina i trinaest podskupina. Pears i ostali [44] ih u pregledu literature o učenju programiranja dijele u četiri skupine:

- Vizualizacijski alati - kroz vizualne metafore prikazuju određene aspekte izršavanja programa
- Alati za automatsko ocjenjivanje - omogućuju studentima da predaju svoje programe , a alat im dodjeli ocjenu. Ovi alati

najčešće koriste unaprijed pripremljene testne slučajeve pomoću koji provjeravaju radi li program ispravno

- Programske okoline - su različiti oblici integriranog razvojnog sučelja posebno prilagođenog za edukacijski kontekst
- Alati za podršku programiranju - su svi ostali alati koji ne spadaju i prije opisane grupe

Značajan broj tih sadrži osnovne elemente integriranog razvojnog okruženja. U njima se može kreirati, ispraviti i pokrenuti program. Kao što se i IDE vremenom mijenjao i razvijao i danas često sadrži i kolaborativne elemente, tako su se mijenjali i alati za učenje programiranja. Rastom popularnosti web platforme i prelaskom alata u to okruženje naglo raste broj onih koji podržavaju kolaborativni rad. Jedan od razloga za to je priroda weba - on je socijalna platforma i stvoren je da bi ljudi mogli komunicirati, surađivati i dijeliti. Za alate u takvom okruženju prirodno je da posjeduju svojstva koja to omogućuju. Sljedeći razlog je razvoj učenja na daljinu i popularnost masivnih otvorenih online tečajeva (engl. Massively Open Online Courses - MOOCs). Kolaboracija i osjećaj zajedništva su važni da bi MOOC-ovi bili učinkoviti [45]. Kolaborativne funkcionalnosti alata poželjne su i u učenju licem u lice. One omogućuju manji broj nastavnog osoblja, odnosno pružanje edukacije većem broju korisnika (uz pozitivne efekte spomenute u ranijim poglavljima).

C. Alati za kolaborativno učenje programiranja

Iako brojna istraživanja pokazuju pozitivne efekte korištenja kolaborativnog učenja u obrazovanju [46], njegovo korištenje ne mora uvijek dati poželjne rezultate [47][48]. Korištenje kolaboracije u učenju može biti vrlo zahtjevno i potrebno je pažljivo odabrati strategiju i elemente implementacije da bi se dobio optimalan efekt [49]. Jedna od tehnologija koja može poslužiti kao katalizator u poboljšanju učinkovitosti korištenja suradničkog učenja je računalo.

Ideja o korištenju računala u suradničkom učenju javlja se krajem 80-ih godina pojavom koncepta računalom podržanog kolaborativnog učenja (engl. Computer Supported Collaborative Learning – CSCL).

Računalom podržano kolaborativno učenje koristi računala kao primarni izvor za dijeljenje znanja i njegovu izgradnju. Korištenjem računalnih tehnologija u procesu učenja povećavamo efikasnost tog procesa .

Pri učenju programiranja (kao i pri programiranju) gotovo uvijek koristimo računalo. Dakle, pošto već imamo na raspolaganju računalo logična je ideja da ga upotrijebimo da bi olakšali taj proces učenja. Računalo ima dugu povijest korištenja u edukaciji kao alat koji olakšava učenje, ali i kao alat za komunikaciju. Zato u literaturi o suradničkom učenju kao sredstvo učenja nalazimo i standardne programe za komunikaciju i kolaboraciju (e-pošta, diskusije, forumi...), ali i programe posebno oblikovane za obrazovanje u nekom području proširene s dodatnim funkcionalnostima za grupno učenje.

Postoje primjeri istraživanja o uspješnoj primjeni wikija [50], blogova [51], podcasta [52], razmjena izravnih poruka [53] i drugih sličnih alata u učenju programiranja. U slučaju korištenja ovakvih alata pojavljuje se problem promjene konteksta kad korisnik osim programa za komunikaciju koristi i program u kojem kodira. Neprekidno prebacivanje iz jednog u drugi program ometa njegov rad. Zato se ovakav pristup koristi za one grupne aktivnosti kad korisnik ne piše program (npr. diskusija u okviru MOOC-a).

Ako govorimo o slučajevima kad želimo dodati komunikacijske elemente u aktivnosti koje uključuju programiranje onda se te komunikacijske funkcionalnosti ugrađuju unutar alata koji se koristi za takvu aktivnost.

Kolaborativni elementi mogu se dodavati kao nadogradnje unutar drugih alata koji se koriste za učenje programiranja kao što su:

- Sangam [54] - koji je plug-in za popularni IDE Eclipse i omogućava da više korisnika prati rad jednog od njih u razvojnoj okolini.
- CollabVS [40] - proširenje Visual Studia koje u to programsko okruženje dodaje alate za komunikaciju i grupni rad.

Najbrojnija skupina alata su integrirana razvojna sučelja sa mogućnošću grupnog editiranja i neki dodatnim alatima za komunikaciju među članovima grupe. Neki od primjera takvih alata su:

- Collabode [55] - kolaborativni editor za sinkrono grupno editiranje u realnom vremenu. Izgrađen je na temelju EtherPad (http://etherpad.org/) online kolaborativnog editora otvorenog koda. Ovaj program postoji i kao nadogradnja za Eclipse razvojno okruženje
- RIPPLE [56] - alat za distribuirano programiranje u paru. U njemu studenti modu istovremeno raditi na istom kodu, i maju i mogućnost razgovora (chat). Istraživanje pokazuje da studenti imaju pozitivan odnos prema korištenju alata i ima potencijal za dugoročno učenje, motivaciju i zadržavanje znanja. To je ujedno i alat za prikupljanje podataka o kolaborativnom programiranju, jer bilježi akcije korisnika
- COLLECE [57] - alat za grupni rad koji podržava sinkrono programiranje, prevođenje, pokretanje programa i mogućnost razgovora među korisnicima.
- CodeWave [58] - kolaborativni IDE koji nudi funkcionalnosti klasične razvojne okoline sa ugrađenim alatima za komunikaciju. Svi alati potrebni studentu za grupni rad su jednom sučelju. Kroz poruke se drugim korisnicima mogu slati i veze koje upućuju na dio koda. Za edukatore alat nudi opciju praćenja povijesti akcija korisnika. Sastoji se od poslužiteljske i klijent komponente.

- IDE 2.0 [59] - kolaborativni IDE na web platformi, nudi komunikaciju među korisnicima, praćenje rada studenata, podjelu studenata u grupe i obavijesti
- IDEOL [60] - kolaborativni web IDE sa brojnim mogućnostima za suradnički rad
- CodePilot [61] - kolaborativni web IDE za pisanje web aplikacija (HTML/CSS/JS). Studenti smatraju da olakšava komunikaciju i preferiraju ga u odnosu na klasično razvojno sučelje
- CodeR [62] - kolaborativni web radno okružje sa alatima za izradu, izvršavanje. Sadrži i terminal i podršku za razgovor (engl. chat)
- Jimbo [63][64] - kolaborativni web IDE sa velikim brojem ugrađenih komunikacijskih alata (sadrži i audio komunikaciju koja olakšava rad na kodiranju jer student može istovremeno kodirati i pričati)
- CoRED [65][66] - web IDE koji omogućava kolaborativno kodiranje, ali ne sadrži nikakve dodatne alate za komunikaciju

Za razliku od uobičajenih razvojnih okolina velik broj ovih alata podržava rad sa više programskih jezika. Razlog za to je što su ti alati obično nastajali na bazi nekog od kolaborativnih editora otvorenog koda poput Atoma (https://atom.io/). Posljedica takve arhitekture je mogućnost da u takav alat relativno jednostavno dodamo podršku za dodatni programski jezik.

Neki od alata nisu proširenja uobičajenih razvojnih sučelja sa dodatnim kolaborativnim funkcionalnostima već su nastali nadgradnjom postojećih vizualizacijskih sustava ili virtualnih okolina poput ili kombiniranje više njih:

- JeCo (Jeliot Collaborative) [67] - alat nastao proširenjem vizualizacijskog alata za programiranje Jeliot 3 sa kolaborativnim elementima iz alata za suradnički rad Woven Stories
- AliCe-ViLlagE [68] - kolaborativna verzija poznate virtualne okoline za učenje programiranja Alice .sa podrškom za programiranje u paru. Unutar ove okoline korisnik piše programe koji kreiraju virtualne svjetove i stvara pravila interakcije među objektima u tim svjetovima

Postoje i primjeri alata koji nemaju svoj ekvivalent u pojedinačnom načinu učenja jer koriste kolaborativne tehnologije u učenju programiranja na inovativan način kao što je :

- Codeopticon [69] - kreiran da omogući pružanje instrukcija jedan-na-jedan unutar rada sa većom grupom studenata. Svaki student piše program na svom računalu. Nastavnik na računalu ima drugačije sučelje na kojem može vidjeti rad na kodu svakog studenta. Unutar sučelja nastavnik može gledati studentov kod i pratiti kako ga on mijenja. Kad uoči da neki student ima problema nastavnik mu može pomoći preko tekstualnog razgovora. Alat se može koristiti za učenje licem u lice i za udaljeno učenje

Literatura koja istražuje prednosti sinkronog editiranja u realnom vremenu je ograničena [70], ne daje definitivne odgovore o prednostima ovakvog pristupa [54] i zahtjeva daljnja istraživanja. Neke studije pokazuju pozitivan efekt kolaborativnog kodiranja na performanse studenata [71], a kod nekih je taj efekt neutralan [72] što ukazuje na potrebu pažljivog planiranja implementacije tog načina učenja [73]. Istraživanja navode pozitivne efekte na angažman [74][75], brzinu rješavanja zadataka [63], motiviranost [64] i zadovoljstvo [72] studenata. Nažalost većina studija koje predstavljaju takve alate ne sadrži analize efekata njihovog korištenja.

V. ZAKLJUČAK

Kolaborativno učenje danas je jedna od najpopularnijih metoda u poučavanju programiranja. Neki oblik kolaboracije integralni je dio gotovo svakog kolegija/tečaja programiranja. Uspon weba kao nove računalne platforme mijenja paradigmu korištenja računala. Temelj tog novog okruženja je socijalna interakcija, komunikacija i kolaboracija. Iako se kolaborativno učenje koristi već dugo u učenju programiranja, te recentne tehnološke promjene nude mu nove perspektive napretka. S jedne strane imamo već etablirane kolaboracijske tehnike poput programiranja u paru koje iako počiva na kolaboraciji i pri radu se koristi računalo, tu kolaboraciju ne ostvaruje preko računala. Razvoj e-učenja i popularnost MOOC-a dovodi do pojave distribuiranog programiranja u paru. Taj novi oblik korištenja te tehnike zahtjeva korištenje računala radi ostvarenja kolaboracije i otvara prostor za nova istraživanja. S druge strane neki alati poput kolaborativnih editora omogućuju pojavu novih tehnika poput grupnog kodiranja (licem u lice ili distribuirano), kao i postizanje novog nivoa kolaboracije u nekim starim kolaborativnim tehnikama kao što je grupni pregled koda (engl. group code review). Iako u literaturi možemo pronaći dosta članaka o kolaborativnim editorima i njihovom korištenju, u poučavanju programiranja nedostatan je broj empirijskih istraživanja o uspješnosti primjene tih alata.

Da bi se ti alati u edukacijskom okruženju iskoristili na optimalan način i maksimalno iskoristili potencijali koje nudi kolaboracija u učenju programiranja, potrebno je detaljno istražiti okolnosti koje određuju njihovu uspješnu primjenu i moderatore koji utječu na nju. Postoji prostor za istraživanje o modelu prihvaćenosti alata za suradničko učenje programiranja na temeljima TAM modela [76] (engl. Technology Acceptance model) i modela uspješnosti gdje kao osnova može poslužiti IS Success model [77].

LITERATURA

[1] B. Simon, R. Lister, and S. Fincher, “Multi-Institutional Computer Science Education Research: A Review of Recent Studies of Novice Understanding,” In Proceedings of the 36th Annual Frontiers in Education Conference, FIE ’06, pages 12–17. IEEE, 2006.

[2] J. Bennedsen and Caspersen, M.E., "Failure Rates in Introductory Programming", SIGCSE Bull, vol. 39, 2, 2007.

[3] Watson, C., & Li, F. W. "Failure rates in introductory programming revisited". In Proceedings of the 2014 conference on Innovation & technology in computer science education (pp. 39-44). ACM. June, 2014.

[4] Tan, P. H., Ting, C. Y., & Ling, S. W. "Learning difficulties in programming courses: undergraduates' perspective and perception". In Computer Technology and Development, 2009. ICCTD'09. International Conference on (Vol. 1, pp. 42-46). IEEE. November, 2009.

[5] Gomes, A., & Mendes, A. J., "Learning to program-difficulties and solutions". In International Conference on Engineering Education–ICEE (Vol. 2007). September, 2007.

[6] Lahtinen, E., Ala-Mutka, K., & Järvinen, H. M., "A study of the difficulties of novice programmers". In Acm Sigcse Bulletin (Vol. 37, No. 3, pp. 14-18). ACM. June, 2005.

[7] Ben-Ari, M., "Constructivism in computer science education". In Acm sigcse bulletin (Vol. 30, No. 1, pp. 257-261). ACM. March, 1998.

[8] Winslow, L. E. (1996). "Programming pedagogy—a psychological overview". ACM Sigcse Bulletin, 28(3), 17-22.

[9] Jenkins, T., & Davy, J. "Diversity and motivation in introductory programming". Innovation in Teaching and Learning in Information and Computer Sciences, 1(1), 1-9., 2002.

[10] Porter, L., Guzdial, M., McDowell, C., & Simon, B., "Success in introductory programming: What works?". Communications of the ACM, 56(8), 34-36., 2013.

[11] Vihavainen, A., Airaksinen, J., & Watson, C., "A systematic review of approaches for teaching introductory programming and their influence on success". In Proceedings of the tenth annual conference on International computing education research (pp. 19-26). ACM. July, 2014.

[12] Dillenbourg, P. (1999). "Collaborative learning: Cognitive and computational approaches. advances in learning and instruction series". Elsevier Science, Inc., PO Box 945, Madison Square Station, New York, NY 10160-0757.

[13] Panitz, T. "Collaborative versus Cooperative Learning: A Comparison of the Two Concepts Which Will Help Us Understand the Underlying Nature of Interactive Learning". 1999.

[14] Slavin, R. E., Educational psychology: theory and practice (5th ed.). Needham Heights, MA: Allyn & Bacon. 1997.

[15] Bruffee, K. A., "Sharing our toys: Cooperative learning versus collaborative learning". Change: The Magazine of Higher Learning, 27(1), 12-18. 1995.

[16] [-38] Kirschner, P. A., "Using integrated electronic environments for collaborative teaching/learning".Research Dialogue in Learning and Instruction, 2(1), 1–10. 2001.

[17] Beck, L., & Chizhik, A. "Cooperative learning instructional methods for CS1: Design, implementation, and evaluation". ACM Transactions on Computing Education (TOCE), 13(3), 10., 2013.

[18] Beck, K., Extreme programming explained: embrace change. addison-wesley professional, 2000.

[19] [-24] L. Williams and R.R. Kessler, "Pair Programming Iluminated". Addision-Wesley Longman Publishing Co., Inc., 2002.

[20] Hannay, J. E., Dybå, T., Arisholm, E., & Sjøberg, D. I. "The effectiveness of pair programming: A meta-analysis". Information and Software Technology, 51(7), 1110-1122, 2009.

[21] Salleh, N., Mendes, E., & Grundy, J., "Empirical studies of pair programming for CS/SE teaching in higher education: A systematic literature review". IEEE Transactions on Software Engineering, 37(4), 509-525. 2011.

[22] L. Williams, R.R. Kessler, W. Cunningham, R. Jeffries,"Strengthening the Case for Pair Programming," IEEE Software, vol. 17, no. 4, pp. 19-25, July/Aug. 2000.

[23] E. Mendes, L. Al-Fakhri, and A. Luxton-Reilly, "InvestigatingPair-Programming in a 2nd-Year Software Development and Design Computer Science Course," ACM SIGCSE Bull., vol. 37,pp. 296-300, 2005.

[24] T. Bipp, A. Lepper, and D. Schmedding, "Pair Programming inSoftware Development Teams—An Empirical Study of Its Benefits," Information and Software Technology, vol. 50, pp. 231-240, 2008.

[25] C. McDowell, L. Werner, H.E. Bullock, J. Fernald, "The Impact of Pair Programming on Student Performance, Perceptionand Persistence," Proc. 25th Int'l Conf. Software Eng., pp. 602-607, 2003.

[26] Umapathy, K., & Ritzhaupt, A. D., "A Meta-Analysis of Pair-Programming in Computer Programming Courses: Implications for Educational Practice". ACM Transactions on Computing Education (TOCE), 17(4), 16. 2017.

[27] D.T Sato, H. Corbucci and M.V Bravo, "Coding Dojo: An Environment for Learning and Sharing Agile Practices," Agile Conference, (Toronto, Canada), pp.459-464, IEEE, 2008.

[28] K. Heinonen, K. Hirvikoski, Matti Luukkainen, and Arto Vihavainen, "Learning agile software engineering practices using coding dojo," Proceedings of the 14th annual Conference on Information Technology Education (SIGITE '13), (New York, USA), pp. 97-102, 2013.

[29] R.B. Da Luz, A.G Neto and R. Noronha, "Teaching TDD, the Coding Dojo Style," International Conference Advanced Learning Technologies (ICALT), (Beijing, China), pp.371-375, IEEE, 2013.

[30] Hendrix, D., Myneni, L., Narayanan, H., & Ross, M. "Implementing studio-based learning in CS2". In Proceedings of the 41st ACM technical symposium on Computer science education (pp. 505-509). 2010.

[31] Reardon, S., & Tangney, B., "Smartphones, studio-based learning, and scaffolding: Helping novices learn to program". ACM Transactions on Computing Education (TOCE), 14(4), 23. 2015.

[32] Simon, B., Kohanfars, M., Lee, J., Tamayo, K., & Cutts, Q. "Experience report: peer instruction in introductory computing". In Proceedings of the 41st ACM technical symposium on Computer science education (pp. 341-345). ACM. March, 2010.

[33] Cao, Y., & Porter, L., "Evaluating Student Learning from Collaborative Group Tests in Introductory Computing". In Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education (pp. 99-104). ACM. March, 2017.

[34] Hundhausen, C., et al. "Integrating pedagogical code reviews into a CS 1 course: an empirical study." ACM SIGCSE Bulletin. Vol. 41. No. 1. ACM, 2009.

[35] Gaspar, A., & Langevin, S. "Active learning in introductory programming courses through Student-led "live coding" and test-driven pair programming". In International Conference on Education and Information Systems, Technologies and Applications, Orlando, FL. 2007.

[36] Hu, H. H., Kussmaul, C., Knaeble, B., Mayfield, C., & Yadav, A., Results from a survey of faculty adoption of Process Oriented Guided Inquiry Learning (POGIL) in Computer Science. In Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education (pp. 186-191). ACM. July, 2016

[37] da Silva Estácio, B. J., & Prikladnicki, R., "Distributed pair programming: A systematic literature review". Information and Software Technology, 63, 1-10. 2015.

[38] Cook, C. L. R., Towards computer-supported collaborative software engineering, 2007.

[39] Frost, R., "Jazz and the eclipse way of collaboration". IEEE software, 24(6), 2007.

[40] Hegde, R., & Dewan, P., "Connecting programming environments to support ad-hoc collaboration". In Proceedings of the 2008 23rd IEEE/ACM International Conference on Automated Software Engineering (pp. 178-187). IEEE Computer Society, September, 2008.

[41] Booch, G., & Brown, A. W., "Collaborative development environments". Advances in computers, 59, 1-27. 2003.

[42] Kats, L. C., Vogelij, R. G., Kalleberg, K. T., & Visser, E. "Software development environments on the web: a research agenda". In Proceedings of the ACM international symposium on New ideas, new paradigms, and reflections on programming and software (pp. 99-116). ACM., October, 2012.

[43] Kelleher, C., & Pausch, R., "Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers". ACM Computing Surveys (CSUR), 37(2), 83-137. 2005.

[44] Pears, A., Seidman, S., Malmi, L., Mannila, L., Adams, E., Bennedsen, J., ... & Paterson, J., "A survey of literature on the teaching of introductory programming". ACM SIGCSE Bulletin, 39(4), 204-223. 2007.

[45] Hew, K. F., & Cheung, W. S., "Students' and instructors' use of massive open online courses (MOOCs): Motivations and challenges". Educational research review, 12, 45-58. 2014.

[46] Springer, L., Stanne, M. E., & Donovan, S. S., "Effects of small-group learning on undergraduates in science, mathematics, engineering, and technology: A meta-analysis". Review of Educational Research, 69(1), 21–51. 1999.

[47] Kirschner, F., Paas, F., & Kirschner, P. A., "A cognitive load approach to collaborative learning: United brains for complex tasks". Educational Psychology Review, 21, 31–42. 2009.

[48] Kreijns, K., Kirschner, P. A., & Jochems, W. "Identifying the pitfalls for social interaction in computer-supported collaborative learning environments: a review of the research". Computers in human behavior, 19(3), 335-353. 2003.

[49] Nokes-Malach, T. J., Richey, J. E., & Gadgil, S., "When is it better to learn together? Insights from research on collaborative learning". Educational Psychology Review, 1–12. 2015.

[50] Kim, S. H., Han, H. S., & Han, S., The study on effective programming learning using wiki community systems. Innovative Approaches for Learning and Knowledge Sharing, 646-651. 2006.

[51] Jones, D., "Enhancing the learning journey for distance education students in an introductory programming course". Central Queensland University. 2006.

[52] Saeed, N., Yang, Y., & Sinnappan, S., "Emerging web technologies in higher education: A case of incorporating blogs, podcasts and social bookmarks in a web programming course based on students' learning styles and technology preferences". Educational Technology & Society, 12(4), 98-109. 2009.

[53] Bigham, J. P., Aller, M. B., Brudvik, J. T., Leung, J. O., Yazzolino, L. A., & Ladner, R. E., "Inspiring blind high school students to pursue computer science with instant messaging chatbots". In ACM SIGCSE Bulletin (Vol. 40, No. 1, pp. 449-453). ACM. March, 2008.

[54] Brodahl, C., Hadjerrouit, S., & Hansen, N. K., "Collaborative writing with Web 2.0 technologies: education students' perceptions". 2011.

[55] Goldman, M., Little, G., & Miller, R. C., "Real-time collaborative coding in a web IDE". In Proceedings of the 24th annual ACM symposium on User interface software and technology (pp. 155-164). ACM. October, 2011.

[56] Boyer, K. E., Dwight, A. A., Fondren, R. T., Vouk, M. A., & Lester, J. C., "A development environment for distributed synchronous collaborative programming". In ACM SIGCSE Bulletin (Vol. 40, No. 3, pp. 158-162). ACM. June, 2008.

[57] Bravo, C., Duque, R., & Gallardo, J., "A groupware system to support collaborative programming: Design and experiences". Journal of Systems and Software, 86(7), 1759-1771. 2013.

[58] Vandeventer, J., & Barbour, B., "CodeWave: a real-time, collaborative IDE for enhanced learning in computer science". In Proceedings of the 43rd ACM technical symposium on Computer Science Education (pp. 75-80). ACM. February, 2012.

[59] Itahriouan, Z., Aknin, N., Abtoy, A., & El Kadiri, K. E., "An experimental study of software engineering learning using IDE 2.0". In Information Science and Technology (CiSt), 2016 4th IEEE International Colloquium on (pp. 559-563). IEEE. October, 2016.

[60] Tran, H. T., Dang, H. H., Do, K. N., Tran, T. D., & Nguyen, V., "An interactive Web-based IDE towards teaching and learning in programming courses". In Teaching, Assessment and Learning for Engineering (TALE), 2013 IEEE International Conference on (pp. 439-444). IEEE. August, 2013.

[61] Warner, J., & Guo, P. J., "CodePilot: Scaffolding End-to-End Collaborative Software Development for Novice Programmers". In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (pp. 1136-1141). ACM. May, 2017.

[62] Kurniawan, A., Soesanto, C., & Wijaya, J. E. C., "CodeR: Real-time Code Editor Application for Collaborative Programming". Procedia Computer Science, 59, 510-519. 2015.

[63] Ghorashi, S., & Jensen, C., "Integrating Collaborative and Live Coding for Distance Education". Computer, 50(5), 27-35. 2017.

[64] Palviainen, J., Kilamo, T., Koskinen, J., Lautamäki, J., Mikkonen, T., & Nieminen, A., "Design framework enhancing developer experience in collaborative coding environment". In Proceedings of the 30th Annual ACM Symposium on Applied Computing (pp. 149-156). ACM. April, 2015.

[65] Lautamäki, J., Nieminen, A., Koskinen, J., Aho, T., Mikkonen, T., & Englund, M., "CoRED: browser-based Collaborative Real-time Editor for Java web applications". In Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work (pp. 1307-1316). ACM. February, 2012.

[66] Nieminen, A., Lautamäki, J., Kilamo, T., Palviainen, J., Koskinen, J., & Mikkonen, T., "Collaborative coding environment on the web: A user study". Developing Cloud Software Algorithms, Applications, and Tools,(60), 275-300. 2013.

[67] Moreno, A., Myller, N., & Sutinen, E., "JeCo, a collaborative learning tool for programming". In Visual Languages and Human Centric Computing, 2004 IEEE Symposium on (pp. 261-263). IEEE. September, 2004.

[68] Al-Jarrah, A., & Pontelli, E. " AliCe-ViLlagE" Alice as a Collaborative Virtual Learning Environment. In Frontiers in Education Conference (FIE), 2014 IEEE (pp. 1-9). IEEE. October, 2014.

[69] Guo, P. J. "Codeopticon: Real-time, one-to-many human tutoring for computer programming". In Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (pp. 599-608). ACM. November, 2015.

[70] Kato, Y., Matsuzawa, Y., & Sakai, S. "Promoting collaborative programming for introductory programming courses through an "individual work branch and real-time sharing" approach". In IFIP TC3 Working Conference "A New Culture of Learning: Computing and next Generations" (p. 121).

[71] Othman, M., Othman, M., & Hussain, F. M., "Designing prototype model of an online collaborative learning system for introductory computer programming course". Procedia-Social and Behavioral Sciences, 90, 293-302. 2013.

[72] Nguyen, V., Dang, H. H., Do, N. K., & Tran, D. T. "Enhancing team collaboration through integrating social interactions in a Web - based development environment". Computer Applications in Engineering Education, 24(4), 529-545. 2016.

[73] Zhou, W., Simpson, E., & Domizi, D. P., "Google Docs in an Out-of-Class Collaborative Writing Activity". International Journal of Teaching and Learning in Higher Education, 24(3), 359-375. 2012.

[74] Hickey, T. J., Langton, J., & Alterman, R., "Enhancing CS programming lab courses using collaborative editors". Journal of Computing Sciences in Colleges, 20(3), 157-167. 2005.

[75] Harding, T. D. "Experiences of Using a Collaborative Programming Editor in a First-Year Programming Course". 2014.

[76] Venkatesh, V., & Bala, H., "Technology acceptance model 3 and a research agenda on interventions". Decision sciences, 39(2), 273-315. 2008.

[77] DeLone, W. H., & McLean, E. R. "Information systems success: The quest for the dependent variable". Information systems research, 3(1), 60-95. 1992.