ITS za razvoj veština pisanja održivog koda

I. Problem

Inženjeri softvera treba da pišu *kvalitetan* kod koji ispunjava funkcionalne zahteve. Važna karakteristika *kvaliteta* koda je *održivost*. Održivost određuje lakoću razumevanja, izmene, proširivanja i testiranja softverskih komponenti [1]. Kod niske održivosti uvećava cenu razvoja softvera i loše utiče na ostale aspekte njegovog kvaliteta, poput bezbednosti, performansi i pouzdanosti [2][3]. Stoga industrija i akademija ističu da je važno da inženjeri softvera nauče da pišu održiv kod [4][5][6][7].

Postoji mnoštvo heuristika za pisanje održivog koda [8][9][10]. Mnoge su neprecizni opisi podložni subjektivnim interpretacijama [11] ili su previše tesno vezane za programske jezike, tehnološke konvencije ili arhitekturalne šablone [12]. Posledično, veštine pisanja održivog koda se stiču kroz izradu velikog broja zadataka uz podršku eksperta koji daje povratne informacije. Međutim, postoji oskudica eksperata koji bi mogli predavati ovo gradivo [13].

Inteligentni Tutoring Sistemi (*Intelligent Tutoring Systems*, ITS) bi mogli umanjiti ovaj problem simulacijom ljudskog instruktora. Ovaj rad predstavlja ITS specijalizovan za učenje pisanje održivog koda. Glavna komponenta ovog ITS-a je zadužena da prima kod od učenika i vraća personalizovane povratne informacije za ispravku tog koda.

Razvijeni ITS može da pomogne učenicima da savladaju osnove veštine pisanja održivog koda kroz manje izazove. Zahvaljujući tome, instruktori mogu posvetiti više vremena radu sa učenicima na sofisticiranijim problemima. ITS takođe podržava instruktore u otkrivanju miskoncepcija u znanju učenika.

II. Teorijske osnove

Poglavlje II.A opisuje domen pisanja održivog koda. Kroz ovo poglavlje ilustrujemo kako ne postoji precizna definicija održivog koda, te se veština pisanja održivog koda mora učiti kroz primere. Poglavlje II.B analizira ciljeve interesnih grupa i na osnovu njih definiše zahteve koje rešenje treba da ispuni kako bi doprinelo ispunjenju tih ciljeva.

A. *Održivost koda*

NAPOMENA: poglavlje nije napisano, ali je iznad istaknut njegov cilj. Okvirno, organizacija poglavlja bi bila: (1) definicija pojma održivosti koda, (2) konkretan primer problema održivosti koda (npr., *Long Method*). Primer bi pokazivao kako ne postoje precizne (merljive) heuristike koje jednoznačno definišu funkciju kao „dugačku“.

B. *Zahtevi koje mora da ispuni edukativni alat za razvoj veštine pisanja održivog koda*

Zahtevi koje treba da ispuni edukativni alat za razvoj veštine pisanje održivog koda su izvedeni iz našeg iskustva kao instruktora i empirijskih studija koje su istraživale potrebe instruktora za analitikama učenja u učionici [2][5]. Tabela II prikazuje zahteve koje rešenje treba da ispuni i objašnjava kako oni doprinose ispunjenju ciljeva interesnih grupa koje će koristiti rešenje.

C. *Teorijske osnove rešenja koje ispunjava izvedene zahteve*

ITS je tip tehnologije učenja koji modeluje principe efektivne instrukcije (*model instruktora*) da adaptira edukativni sadržaj (*model domena*) specifičnostima učenika (*model učenika*) [2]. ITS-ov *korisnički interfejs* definiše instrumente (*learning instrument*) koji prezentuju edukativni sadržaj i rukuju interakcijom učenika sa sadržajem [2].

Cilj ITS-a je da razvije temeljno znanje učenika u najkraćem mogućem roku [3]. ITS prati interakcije studenata sa edukativnim sadržajem kako bi podržao svoju prilagodljivost i omogućio analitiku učenja [4]. Na osnovu prikupljenih događaja, instruktori mogu da identifikuju učenike kojima je potrebna pomoć, uobičajene miskoncepcije,

Tabela II Zahtevi alata za učenje veštine pisanja održivog koda

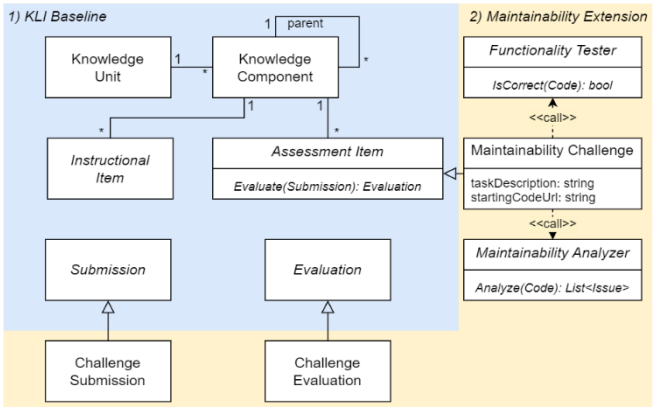
Cilj interesne grupe	Zahtev	Argument kako zahtev doprinosi cilju
Studenti softverskog inženjerstva žele efektivno da savladaju veštinu pisanja održivog koda kako bi unapredili svoje karijere	1. Alat treba da simulira situacije iz realnog sveta koje zahtevaju upotrebu veštine pisanja održivog koda	Realistično okruženje povećava angažovanost u učenju i rezultate učenja, pomažući učenicima da razviju veštine potrebne za dato realistično okruženje [1]. Na primer, pitanje sa višestrukim izborom je manje realistično od programskog zadatka koji se rešava u integrisanom razvojnom okruženju
	2. Alat treba da automatski identifikuje nedostatke u znanju svakog studenta	Rešenje treba da automatski identifikuje nedostatke u znanju svakog studenta kako bi se omogućilo automatsko prilagođavanje instrukcije
	3. Alat treba da se automatski prilagodi nedostacima u znanju studenta odabirom adekvatnih zadataka i povratnih informacija	Rešenje treba da se adaptira nedostacima u znanju svakog studenta kako bi se optimizovali dobici u učenju i time efikasno razvile veštine učenika.
Instruktori teže donošenju odluka zasnovanih na podacima kako bi poboljšali iskustvo učenja za sebe i studente	4. Alat treba da nadgleda i prikazuje progres svakog studenta	Pomoću pregleda aktivnosti svakog učenika, instruktor može da usmeri svoju pažnju na studente kojima treba pomoć, dok alat obučava ostale studente.
	5. Alat treba da prati interakcije studenata sa edukativnim sadržajem da bi podržao analizu kvaliteta sadržaja	Pomoću ove funkcionalnosti, instruktori mogu da analiziraju studentske odgovore na zadatke, vreme provedeno u svakom segmentu ili šablone u učenju. Na taj način instruktori razumeju kojim delovima procesa učenja i edukativnih materijala je potrebno unapređenje. Specifično za održivost koda, instruktor može da analizira predat kod da bi utvrdio kršenje principa, preporuka i heuristika i adekvatno razvio model veština pisanja održivog koda.

preterano izazovne zadatke i edukativni sadržaj koji je neophodno unaprediti [5].

Postoji više pristupa dizajnu ITS [6]. Pristup koji se ističe po svojoj zrelosti i nivou empirijske verifikacije je *Knowledge-Learning-Instruction* (KLI) radni okvir [3]. KLI definiše koncepte koji modeluju znanje, učenje i instrukciju (slika 1, deo). *Instruktivni elementi* (*instructional items*) su varijacije u okruženju učenja koje uzrokuju porast znanja učenika. Na primer, ovde spadaju edukativni materijali prezentovani u formi videa ili teksta koji izlažu gradivo. *Instruktivni elementi* razvijaju nove i rafiniraju postojeće veštine (*knowledge components*) koje učenici moraju da primene da bi rešili probleme iz domena. Domenski problemi su modelovani kroz *zadatke* (*assessment items*). Primeri *zadataka* mogu biti zadaci koji zahtevaju kraći odgovor na pitanje ili pisanje koda koji ispunjava određene zahteve. *Zadaci evaluiraju* (*Evaluate*) učenički *odgovor* (*Submission*) da bi utvrdili stepen razvoja *veština* učenika i vratili adekvatne povratne informacije.

III. REŠENJE

Slika 1 predstavlja domenski model rešenja. Osnova domenskog modela je *Knowledge-Learning-Instruction* (KLI) radni okvir predstavljen u poglavlju III.A. Nad ovom osnovom je izgrađena *Maintainabillity Challege* komponenta ITS-a (poglavlje III.B) koja je fokus ovog rada. Ova komponenta je specijalizovana za proveru koliko je učenika vešt u pisanju održivog koda.



Slika 1 Domenski model inteligentnog tutoring sistema za razvoj veštine pisanja održivog koda.

LITERATURA

[1] R. Weinhandl, T. Houghton, and Z. Lavicza, "A case study on learning basic logical competencies when utilising technologies and real-world objects," *Education and Information Technologies*, vol. 26, no. 1, pp. 639-653, 2021. DOI: 10.1007/s10639-020-10282-5

[2] E. Mousavinasab, N. Zarifsanaiey, S. R. Niakan Kalhori, M. Rakhshan, L. Keikha, and M. Ghazi Saeedi, "Intelligent tutoring systems: a systematic review of characteristics, applications, and evaluation methods," *Interactive Learning Environments*, vol. 29, no. 1, pp. 142-163, 2021. DOI: 10.1080/10494820.2018.1558257

[3] K. R. Koedinger, A. T. Corbett, and C. Perfetti, "The Knowledge-Learning-Instruction framework: Bridging the science-practice chasm to enhance robust student learning," *Cognitive science*, vol. 36, no. 5, pp. 757-798, 2012. DOI: 10.1111/j.1551-6709.2012.01245.x

[4] K. Mangaroska, and M. Giannakos, "Learning analytics for learning design: A systematic literature review of analytics-driven design to enhance learning," *IEEE Transactions on Learning Technologies*, vol. 12, no. 4, pp. 516-534, 2018. DOI: 10.1109/TLT.2018.2868673

[5] K. Holstein, B. M. McLaren, and V. Aleven, "Intelligent tutors as teachers' aides: exploring teacher needs for real-time analytics in blended classrooms," in *Proceedings of the seventh international learning analytics & knowledge conference*, Mar. 2017, pp. 257-266. DOI: 10.1145/3027385.3027451

[6] D. Dermeval, R. Paiva, I. I. Bittencourt, J. Vassileva, D. Borges, "Authoring tools for designing intelligent tutoring systems: a systematic review of the literature," *International Journal of Artificial Intelligence in Education*, vol. 28, no. 3, pp. 336-384, 2018. DOI: 10.1007/s40593-017-0157-9