Neki autorskopravni aspekti automatskog generisanja koda

Haris Hasić
Pravni fakultet Univerziteta u Travniku
Kiseljak, Bosna i Hercegovina
haris.hasic@pfk.edu.ba

Vladimir Vujović
Elektrotehnički fakultet
Istočno Sarajevo, Bosna i Hercegovina
vladimir_vujovich@yahoo.com

Sadržaj—**Radi povećanja produktivnosti, razvoj softvera se sve više oslanja na korištenje sistema za automatsko generisanje koda. Ovaj zadatak može biti obavljen koristeći CASE alate ili namjenski kreirane generatore koda. Kao rezultat, nameće se niz pitanja o pravnoj prirodi automatskih generatora koda i autorstvu rezultirajućih računarskih programa. Rad razmatra stanje tehnike po pitanju automatskog generisanja objektnog i izvornog koda, te implikacije koje se povlače korištenjem takvih mehanizama po pitanju autorskopravne zaštite. Prvo je razmotreno pitanje da li automatski generatori koda predstavljaju intelektualnu tvorevinu koja je sama računarski program ili je bliža programskom jeziku. Zaključeno je da su automatski generatori koda računarski programi i stoga autorskopravno zaštićeni. Onda su razmotreni razni odgovori i napravljena skala autorstva nad računarskim programima koji su rezultat istog.**

Ključne riječi – Domain-Specific Language; Domain-Specific Model; Model-Driven Architecture; generatori koda; razvoj softvera; programski jezici; funkcionalnost; autorstvo; koautorstvo;

I. UVOD

Držimo da je razvoj softvera u konstantnim naporima ka povećanju efikasnosti i produktivnosti developera i programa, a naglasak se stavlja na brže pisanje optimizovanog programskog koda. Sa pojavom asemblerskog jezika, korištenje kompajlera dolazi do izražaja, čime se pisanje programa u mašinskom kodu potpuno zamjenjuje alatima za generisanje seta instrukcija. Sa pojavom GPL (General-Purpose Language) jezika treće generacije, i uvođenjem objektne orijentacije, nastaje eksplozija CASE (Computer Aided Software Engineering) alata koji pokrivaju svaki aspekt u životnom ciklusu razvoja softvera, što omogućava mnogostruko povećanje produktivnosti.

Detaljnije razumijevanje prirode problema, dovodi do pojave domenski-specifičnih jezika (DSL) i domen-specifičnih modela (DSM) na osvnovu kojih se proširuju i nadopunjuju CASE alati, čime se mijenja pristup rješavanju problema softverskog inženjerstva. U ovom slučaju alati pružaju punu podršku developeru da pristupi modelom upravljavljanom razvoju softvera odnosno grafičkom predstavljanju domena problema i domena rješenja. CASE alati za ispunjavanje ovog zadatka moraju ponuditi ne samo okruženje za rad, već i mogućnost generisanja koda na osnovu datih modela. Jedan od predstavnika ovakvog razvoja softvera jeste UML (Unified Modeling Language) koji kroz svoj objektni model predstavlja objekte i odnose između objekata te alat za modelovanje PowerDesigner koji omogućava da se UML objektni model direktno mapiraju na programski kod, odnosno da se generiše programska struktura. Korak dalje jeste primjena specifičnih generatora koda namjenjenih za određeni problem, kao i korištenje radnih okvira (frameworka), biblioteka i generičkih objekata.

Autorsko pravo je grana prava koja štiti subjekte prava – autore kao fizička lica koja su napravila određeno autorsko djelo i njihove pravne sljednike, kao lica koja su povodom nekog pravnog osnova stekli ličnopravna ili imovinskopravna ovlaštenja na određenom djelu.

Autorsko djelo se definiše kao ljudska tvorevina koja je prvenstveno originalna, u smislu da je jedinstvena i da je duhovna tvorevina, te da ima određenu formu.

Autorsko pravo, u Zakonu o autorskim i srodnim pravima Bosne i Hercegovine (Sl. Glasnik BiH 63/10) (u ostatku teksta ZAISP) [1] , član 102. štiti računarske programe kao književna djela.

Kao univerzalno pravilo autorskog prava navodi se nešto što se zove ideja-ekspresija dihotomija. Ona podrazumijeva da su autorskim pravom sigurno zaštićeni izrazi slobodnih, kreativnih izbora autora kojim on iskazuje svoju kreativnost, a sigurno nisu zaštićeni, odnosno iz pravne zaštite su izuzte ideje kao takve. Razlog zašto su ideje izuzete je da bi se omogućilo svim drugim autorima kao i svim subsekventnim pokoljenjima da mogu slobodno da stvaraju nova djela, a ne da se stvori monopol nad idejama. Naglasak kod autorskih prava je na zaštiti izražaja autora.

Kod računarskih programa izražaj predstavlja izvorni i objektni kod računarskog programa, te u svakom slučaju izvorni i objektni kod računarskog programa je izvjesno zaštićen. Pored izvornog i objektnog koda, po Zakonu o autorskim i srodnim pravima BiH u autorskopravnu zaštitu su po članu 102. St.1 [1] uključeni i pripremni materijali iz kojih se može dobiti računarski program. Ova odredba će posebno biti intersantna u daljnjoj analizi.

Iz autorskopravne zaštite po članu 102. St. 2, drugim propisima ZAISP-a [1] i po Direktivi 2009/24/EC (Direktiva 209/24/EC Evropskog Parlamenta i Vijeća od 23. Aprila, 2009. Godine o pravnoj zaštiti računarskih programa) [2] (u ostatku teskta Direktiva 2009/24/EC) su pored ideja izuzeti i koncepti, postupci, matematske operacije, algoritmi i drugi elementi. Po

presudi SAS v. WPL (Slučaj C-406/10) [3][4] iz autorskopravne zaštite su izuzeti i funkcionalnost računarskog programa, programski jezik i format podataka.

Po Zakonu o autorskim i srodnim pravima BiH autorsko pravo traje sedamdeset godina nakon života autora. Ovo je izuzetno dug period trajanja za računarske programe, čiji životni ciklus se rjetko mjeri inkrementima od po pet godina.

Autor računarskog programa ima isključivo pravo na svake oblike reprodukcije računarskog programa, na prilagođavanje i prevođenje autorskog djela, te na iznajmljivanje djela.

Korisnik računarskog programa ima isto izvjesna prava. Ta prava su da može izvrši bilo šta što je neophodno da bi pokrenuo program, ukloniti greške iz programa, da napravi sigurnosu kopiju, da posmatra rad programa kako bi shvatio njegovu ideju i principe na kom je zasnovan.

Autorskopravna zaštita je uslovljena činom stvaranja. Čin stvaranja podrazumijeva da autor donosi niz slobodnih i kreativnih odluka kojiima utiska svoju ličnost na određeno autorsko djelo. Automatski generatori koda donose dosta tih izbora za i u ime autora. Kao rezultat ovo insinuira određena pravna pitanja.

Neka od pravnih pitanja koja su analiziramo u ovom radu, a koja nastaju kao rezultat implikacija korištenja sistema za automatsko generisanje koda računarskog programa su:

1. Da li su sistemi za automatsko generisanje koda računarskog programa (u ostatku rada AGCS) sami računarski programi, koji su autorskopravno zaštićeni ili su bliži pojmu programskog jezika koji nije?
2. U slučaju da su AGCS-ovi računarski programi, tko će se smatrati autorom računarskog programa koji je nastao njihovim korištenjem, u kom slučaju postoji pet eventualnosti:

   a) autor je računar,
   b) autor je autor AGCS-a(u ostatku teksta izvorni autor),
   c)izvorni autor i autor korisnik su koatuori na rezultirajućem računarskom programu,
   d)autor korisnik stvara djelo prerade koristeći AGCS-ove, te
   e)autor korisnik je autor rezultirajućeg računarskog programa?

U razmatranju drugog pitanja će se uzeti i u obzir implikacije AGCS-ova na samo autorstvo računarskih programa.Međutim, prvobitno će se skrenuti pažnja na strukturu i razvoj softverskih sistema, te će biti izvršena podjela generatora.

II. RAZVOJ SOFTVERA

Razvoj softvera je blisko povezano sa softverskim inženjerstvm koje je danas vec prepoznato kao legitimna inženjerska disciplina i profesija [5]. Nametanje računara i softvera za rješavanje problema nije nužno i neophodno, ali često je opravdano kao sredstvo za implementaciju rješenja [2]. Ovakav pristup, razvoj softvera stavlja u sam fokus razmatranja. Softversko inženjersvo je tokom godina evoluiralo uvodeći nove pristupe kreiranja softvera, te alate koji daju podršku developerima i naručiocima, a sve u svrhu poboljšanja produktivnosti. Dinamička priroda softvera, u literaturi iz softverskog inženjerstva, dovela je do pojave mnogih modela koji opsuju način na koji se stvarno odvija, odnosno recepte za sprovođenje procesa razvoja softvera [2]. Izrada modela procesa i razmatranje njegovih podprocesa pomažu projektantskom timu da bolje razumiju prirodu problema te shvate jaz između onoga "*šta bi trebalo*" i onoga "*šta jeste*" softver [2]. U [2] su prikazani najčešći modeli razvoja softvera:

- model vodopada,
- V model,
- prototipski model,
- transformacioni model,
- inkrementalni i iterativni model,
- spiralni model te
- agilni model razvoja softvera.

Međutim, striktna primjena ovih modela može rezultirati neefikasnšću samog procesa izrade softvera, smanjenju produktivnosti pa čak rezultirati i odustajanjem od realizacije istog. Upravo iz navedenih razloga moralo se pribjeći iznalaženju novih metoda i načina razvoja softvera.

Prelaskom sa asemblera na kompajlere, prije 30 godina, uvodeći GPL jezike treče generacije, povečalo je produktivnost za oko 500% [7][8]. Recept za ovakav uspjeh je bilo u povečavanju nivoa apstrakcije, i automatskom generisanju asemblerskog koda iz koda GPL-a [8]. Uvođenjem DSL-a ova brojka postaje mnogostruko veća.

Razlozi koji su doveli do promene pristupa razvoja softvera i uvođenje DSL-a su veoma racionalni i dati u [7]:

- velika ponovljivost zadataka,
- automatizacija dijelova koda koji se višestruko ponavlja,
- veča efikasnost prilagođenih rješenja naspram onih generičke prirode.

kao i same beneficije korištenja DSL-a date u [8]:

- problem se rješava jednom, na najvećem nivou apstrakcije, dok se finalni kod automatski generiše,
- fokus developera se prebacuje sa koda na dizajn, odnoso rješavanje samog problema pri čemu se sakriva kompleksnost,
- modeli su bazirani na domenu problema, što daje mogućnost da naručioc sudjeluje u razvijanju rješenja,
- konzistentnost i smanjenje grešaka koda napravljene u implementaciji,
- ako postoje alati za modelovanje i rješavanje problema domenske prirode, znatno se povečava produktivnost.

Prepoznavajući prednosti povečavanja nivoa apstrakcije, nastala je "eksplozija" CASE alata za transformaciju i generisanje objektno orijentisanog koda. Međutim u nastojanju da se napravi generičko rješenje, dolazi samo do parcijalnog rješavanja problema, što i jeste najveći problem ovakvog rješenja [9]. Ovakav pristup je zastupljen u najvećem broju alata opšte namjene, a polazna tačka je najčešće UML. Da bi se izbjegla uopštenost i prevladali problemi nastali korištenjem CASE alata, a opet zadržala jednostavnost, pristupilo se

razvijanju posebne tehnologije, odnosno modelom upravljanom razvoju softvera (MDE) koje kombinuje [9]:

- DSL –opisuje formalizme aplikacije, strukture i ponašanja. U kombinaciji sa DSM-om dobija se set alata za modelovanje sitema u domenu problema.
- generatore koda – analiziraju modele, optimizuju ih i vrše njihovu transformaciju u ciljni kod, XML specifikaciju ili neki drugi model.

Generatori koda predstavljaju programe koji pišu programe [10], pri čemu je veoma bitno naglasiti da developer i dalje koristi svoje vještine za rješavanje problema, dok generator kraira sam softver. Beneficije korištenja generatora koda uveliko su slične beneficijama korištenja DSL-a, međutim, naglasak se sad prebacuje na sam kod. Tako su u [10] date osnovne beneficije korištenja generatora koda:

- kvalite,
- konzistentnost,
- brza promjena i prilagodba na svim nivoima razvoja,
- povečanje vremena dizajna i produktivnosti,
- mogućnost brzih promjena dizajna koda.

U [10] generatori koda su podjeljeni u dvije skupine:

- pasivne i
- aktivne.

Pasivni generatori koda predstavljaju generatore koji nemaju nikakvu odgovornost za nastali kod. Najčešće služe za generisanje radnih okvira (frameworka) i često su sastavni dio integrisanih razvojnih alata (IDE – Integrated Development Environment) kao "čarobnjaci".

Aktivni generatori koda predstavljaju generatore koji mogu višestruko da obrađuju izlazne podatke, pri čemu ih djelimično ili potpuno modifikuju. Kada se ustanovi potreba za promjenom, generator se ponovno pokreće čime se sinhronizuju nastale promjene. U [10] je dato 6 modela aktivnih generatora koda:

- *Code munging* – korisit template koda za prebacivanje ulaznog u izlazni oblik koda,
- *Inline-code expander* – koristeći oznake mijenjaju se dijelovi koda sa produkcionim kodom
- *Mixed-code generator* – kao i kod prethodnog modela, sa razlikom da se oznake mijenjaju kodom direktno u ulaznom fajlu.
- *Partial-class generator* – generiše se set klasa koje je naknadno potrebno dopuniti.
- *Tier or layer generator* – generiše se kompletan sloj. Nije potrebno naknadno mijenjanje i dopunjavanje koda.
- *Full-domain language* – kreiraju se posebni jezici i modeli koji imaju mogućnost generisanja kompletnih programa.

Generatori koda danas imaju široku primjenu, kako u razvoju softvera, tako i u njegovom testiranju [11] te kreiranju dokumentacije. Poboljšanja osnovnih principa i varijacije su teme koje će u budućnosti sve više biti razmatrane, pogotovo sa stajališta primjene vještačke inteligencije i ekspertskih sistema i programa koji pišu programe [12].

III. PRAVNA PRIRODA AGCS-OVA

Koja je pravna priroda AGCS-ova pruža odgovor i na pitanje da li su isti računarski programi ili su bliži u pojmu programskih jezika. Računarski programi su autorskopravno zaštićeni, programski jezici nisu. Iz tog razloga odgovor na ovo pitanje je veoma relevantan kako za autora AGCS-ova tako i korisnika po pitanju obima prava koja imaju nad istim, kao i na rezultirajućim proizvodima.

Član 102. St. 1 ZAISP-a [1] definiše računarske programe kao: „Kompjuterski program, u smislu ovog zakona, jeste program u bilo kojoj formi, uključujući pripremne materijale za njegovu izradu.“ Ovo je izuzetno široka definicija. Ona je preuzeta iz Direktive 2009/24/EC Evropskog parlamenta i Vijeća od 23. aprila 2009. godine o pravnoj zaštiti računarskih programa [2], koja u tački 7. Preambule između ostalog određuje da: „pojam „računarski program“ će uključivati programe u bio kom obliku, uključujući i one koji su inkorporisani u hardveru.“ Ova definicija je namjerno ostavljena što širom u obimu, a da bi mogla da uključi sve razvoje u stvarnosti pojma računarskog programa bez mijenjanja. Jedini zahtjev koji ona postavlja na program je da bude „program“ što se može izuzetno široko tumačiti. Iz ove definicije se takođe vidi da program u objektnom i izvornom kodu je zaštićen na jednak način. Jedini kriterij koji Direktiva 2009/24/EC prepoznaje je kriterij originalnosti, odnosno kako tačka 8. Preambule određuje da se ne smiju primjenjivati testovi kvalitete ili estetike programa.

Dakle, strogo gledajući zakonski tekst definicija AGCS su računarski programi. Međutim po članu 102. St. 3 ZAISP-a [1] „Kompjuterski programi zaštićeni su kao pisana autorska djela ako predstavljaju vlastitu intelektualnu tvorevinu njihovog autora.“ Ovo je preuzeta odredba člana 1. St. 3 Direktive gdje je određeno da „računarski program će biti zaštićen samo ukoliko je autorova vlastita intelektualna tvorevina.“ Samo radi razjašnjenja, po članu 1. St. 1 Direktive i po članu 4. St. 2 tačka a) ZAISP-a računarski programi se štite kao književna djela u smislu značenja Bernske Konvencije za zaštitu književnih i umjetničkih djela. Računarski programi općenito su dakle identični po pravnoj zaštiti bilo kojim drugim književnim djelima, i po pravu nema razlike između romana Dostojevskog i Windows operativnog sistema, ali samo ukoliko zadovolje uslov člana 102. St. 3 ZAISP-a.

Direktiva niti Zakon ne daju pobliže određenje šta to znači „intelektualna tvorevina njihovog autora“, tako da se za pojašnjenje značenje mora potražiti u sudskoj praksi. U BiH sudska praksa, u opsegu znanja autora, nije dala definitivno tumačenje značenja ovog stava. Budući da je uredba preuzeta iz Direktive Evropske unije, potrebno se okrenuti ka njihovoj relevatnoj sudskoj praksi. Tijelo koje je nadležno da tumači značenje Direktiva EU je Sud pravde Evropske unije. Taj sud je u svojim odlukama ostavio nacionalnim sudovima da odrede da li djelo zadovoljava ovaj uslov. Kao što je konstatovano, nacionalni sudovi još nisu odlučili po ovom pitanju.

Da bi se došlo do prijedloga tumačenje ovog člana mora se sagledati definiciju autorskog djela iz člana 4. St. 1 ZAISP-a koja je „Autorskim djelom smatra se individualna duhovna tvorevina iz oblasti književnosti, nauke i umjetnosti bez obzira

na vrstu, način i oblik izražavanja, ako ovim zakonom nije drugačije određeno.“ Budući da je izvan pitanja da AGCS su u polju nauke, ovaj kriterij se smatra zadovoljenim. Djelo mora da bude individualno i mora da bude duhovna tvorevina. Autori su u prošlosti argumentovali da riječ „individualno“ se treba tumačiti kao jedinstveno, u smislu da djelo ne smije biti prepisano odnosno da ne može biti istovjetno već postojećem djelu. „Duhovni“ karakter tvorevine autori su u prošlosti tumačili kao odraz individualnost odnosno osobenosti autora.Može se zapaziti da se koristi izraz „duhovni“ a ne „intelektualni“. To je jedan od razloga za nedoumicu po ovom pitanju. Taj odraz individualnosti Sud pravde tumači, a onda se prihvata u vidu teorije izbora autora.

Teorija izbora podrazumjeva da autor kada stvara djelo donosi tri vrste izbora. Prvi su izbori tehničke prirode, koji se odnose na konkretni medij stvaranja i koji nisu slobodni budući da su diktirani eksternim konsideracijama. Tehnički izbori nisu zaštićeni. Drugi izbori su funkcionalni izbori, koji su takvi izbori koji su diktirani utilitarnom prirodom djela. Oni su takođe eksterno diktirani i kao takvi nisu zaštićeni. Treća i najvažnija vrsta izbora su kreativni izbori. Autor pri stvaranju autorskog djela donosi kontinuirane i mnogobojne izbore. Takvi izbori su kreativni i slobodni. Ti izbori odražavaju individualitet autora. Oni su to što razlikuje jedno autorsko djelo od drugog autorskog djela, jer dva autora radeći na istu temu će na osnovu svog individualiteta donjeti različite kreativne izbore. Kao rezultat, statistički je neizvjesno da će se pojaviti dva potpuno identična autorska djela. Da bi bili relevatni ovi izbori moraju biti slobodni, u smislu da su diktirani unutarnjim a ne eksternim uslovima i moraju biti kreativni, u smislu da mora se zahtjevati određeni intelektualni rad koji odražava individualitet stvoritelja, te koji kao takvi ne mogu biti proceduralno ili provizorno provedeni od strane objekta koji nema intelekt.

Nikakvi drugi kriteriji se ne smiju postaviti za ustanovljenje da li AGCS-ovi predstavljaju autorsko djelo.

Potrebno je definisati i programske jezike. Programske jezike možemo shvatiti kao skup svih mogućih dozvoljenih uputstava i pravila koja programer može koristiti da stvori komunikaciju sa računarom radi rješenja određenog konkretnog problema. U programskom jeziku svako uputstvo je u vidu jedne ili više riječi ili drugih simbola, a sve sa tačno i precizno definisanim značenjem. Pravila u pitanju upravljaju korištenjm svakog uputstva.

Advocate General Yves Bot Suda pravde Evropske unije, pišući u Mišljenju povodom slučaja C-406/10, a iznešenog 29. Novembra, 2011. godine, u paragrafu 70 [13]. Citira Patrick Roussel-a kada kaže: „programski jezik kao takav je sličan naučnom djelu, teoretska konstrukcija čija je svrha da organizuje, definiše i prenese značenje sa ciljem pisanja softverskih izvora takvim riječima koje može razumjeti ljudsko biće i koji mogu lako biti transformisani u uputstva koje izvodi računar. Računarski program sačinjava posebne metode koje će se koristiti i potpomaže neophodno razmišljanje da bi se mogli napisati i formalizovati izvorni programi računara. Njegova svrha, različita onoj od programa nije da navede računar da proizvede posebni rezultat već da ustanovi pravila za formulisanje programa koji će omogućiti da se izvrši rezultat.“

Relevantno je da programski jezik kao takav može biti objektivno originalan, u smislu jedinstven, ali da ne zadovoljava subjektivni test karaktera intelektualne tvorevine, u smislu karaktera duhovne tvorevine, budući da programski jezik ni na koji način ne odražava ličnost autora i predstavlja više medij stvaranja nego stvarnu duhovnu tvorevinu, te ne može biti originalan. Programski jezik predstavlja skup svih mogućih pravila i svih mogućih riječi koje računar može da razumije. On predstavlja isti neograničeni potencijal kao i svjež ljudski um ili prazan papir. Kao što ljudski um i prazan papri nisu zaštićeni, tako nije ni programski jezik.

Ovo je jedan od razloga zbog kojih je Sud pravde u slučaju C-406/10 SAS Institute Inc. v. World Programming Ltd, donešenoj 02.05.2012 [4]. godine u stavu 1. presude odlučio: „ni funkcionalnost računarskog programa ni programski jezik i format data fajlova korištenih u računarskom programu da bi se iskoristile izvjesne njegove funkcije sačinjava oblik izražavanja tog programa i, kao takvi, nisu zaštićeni autorskim pravom u računarskom programu za svrhe te direktive.“

Razlog zašto dolazi do zabune i pitanja da li AGCS-ovi predstavljaju računarski program ili programski jezik jeste percepcija da i oni predstavljaju efektivno prazan papir za korisnika autora.

Pomnim razmatranjem drži se da ovo nije tačno. Naime, prvenstvena karakteristika računarskih programa je njihova utilitarnost. Za razliku od tradicionalnih autorskih djela koji su pasivni i gdje smjer prenosa podataka ide isključivo od autora djela na korisnika, to nije slučaj sa računarskim programima, jer korisnici po prirodi računarskog programa su u kontinuiranom kontaktu sa računarskim programom, koji opetovano, reaguje na impute korisnika.

Utilitarnost programa nije razlog za njegovo izuzeće iz autorskopravne zaštite. Knjige vodiči i kuhari su takođe utitarni, u jednom vidu, iako ne kao računarski programi.

Programi za kreativno izražavanje autora su česti. Ekosistem softvera je pun primjera. Autori ovog djela upravo stvaraju ovo djelo na jednom takvom programu. I kao što nitko ne osporava status autorskog djela Microsoft Word programu ili Adobe Photoshop programu tako, drži se da autorskopravna zaštita za AGCS-ove ništa drugačije ne bi trebala biti izvjesna.

Samo zato što je svrha djelovanja takvih programa stvaranje računarskog koda, odnosno samo zato što predstavljaju intelektualnu tvorevinu koja radi na stvaranju, u varijabilnom stepenu autonomije, drugih računarskih programa, drži se da nije razlog za isključenje iz autorskopravne zaštite.

Autor AGCS-ova kao i autor svakog drugog računarskog programa je morao donijeti skup kreativnih, slobodnih odluka o neophodnim elementima da bi se postigao konkretni rezultat, pa je zadovoljen kriterij originalnosti. Programski jezici isto mogu biti originalni, ali oni nisu autorska djela jer nemaju potrebni „duhovni“ karakter.

AGCS-ovi imaju „duhovni“ karakter jer imaju i konkretnu svrhu, funkciju koja se njima želi postići. Ta funkcija je generisanje računarskog koda, što je validna funkcija za računarski program.

Iz svega izrečenog, držimo da su AGCS-ovi validni računarski programi i kao takvi da su zaštićeni autorskim pravom BiH.

IV. AUTORSTVO NA RAČUNARSKIM PROGRAMIMA KOJI SU REZULTAT KORIŠTENJA AGCS-OVA

Razmatranje ovog pitanja bi trebalo pružiti odgovor na to ko se sve može smatrati autorom računarskog programa koji je nastao kao rezultat korištenja AGCS-ova.Ovo je izuzetno bitno za odrediti jer (osim po pitanju zaposlenja i naloga) pruža se odgovor i na pitanje tko će biti nosilac autorskih i drugih prava povodom tih računarskih programa.

Autori su jedino mogli naći da zakoni Ujedinjenog Kraljevstva uređuju pitanje autorstva nad računarskim programima koji su rezultat automatskog generisanja koda, i to u Sekciji 4(3) Autorskog, dizajnerskog i patentnog zakona Ujedinjenog Kraljevstva iz 1988. godine, koji glasi: „U slučaju književnih, dramskih, muzičkih ili umjetničkih djela koji su generisani od strane računara, autor će biti lice od strane koga su aranžmani neophodni za stvaranje djela poduzeti".

Ovo je izuzetno široka i nezgrapna odredba, koja nam ne daje mnogo pojašnjenja niti konkretni odgovor.

Po našem razmatranju, nudi nam se pet mogućih odogovora:

a) *Autor je sam računar*
b) *Autor je lice koje je načinilo AGCS*
c) *Autor AGCS-a i korisnik autor su koautori na novom računarskom programu,*
d) *Korisnik autor prerađuje izvorno autorsko djelo, te*
e) *Korisnik autor je autor novog računarskog programa.*

A. Računar kao autor računarskog programa

Po članu 9. ZAISP-a autor djela je „fizičko lice koje je stvorilo djelo.“ Pravni poredak je jedinstven da fizičko lice je svako ljudsko biće. Dakle, ex lege, autorom računarskog programa ne može biti pravno lice (definisano kao društvena tvorevina fizičkih i/ili drugih pravnih lica, koja ima stabilnu unutarnju strukturu, koja ima imovinu koja je različita od imovine članova, koja je registrovana u nadležnom tijelu, te kojoj je pravni poredak dao pravnu sposobnost što je sposobnost biti nositeljem prava i obaveza), takođe, ex lege autorom računarskog programa ne može biti ništa drugo što nije fizičkim licem. „Umjetne inteligencije“ u najširem mogućem obimu tog pojma, samim tim, su isključene od autorstva. Međutim, postoje izvjesni pravni sistemi koji dozvoljavaju pravnim licima da budu autori računarskih programa.

Pitanje pravnog statusa „umjetnih inteligencija“ je svakako izuzetno zanimljivo i može imati velikog značaja za buduća pokoljenja. Međutim ta polemika prevazilazi opseg ovog rada. Lično, autori smatraju da eventualno „umjetne inteligencije“ koje u potpunosti mogu zadovoljiti turingov test, te koji emuliraju u metafizičkom smislu uvriježene kontencije statusa fizičkih lica će biti ravnopravni subjekti prava, a ne kao sada objekti prava. U tim slučajevima vjerovatno je da će autorskopravna zaštita biti pružena i njihovim intelektualnim ili duhovnim tvorevinama. Autori i to podržavaju.

Činjenica, međutim stoji da računari, kao naprave za procesovanje podataka, nisu sposobni da donose kreativne i slobodne odluke ili izbore, nisu sposobni da razmišljaju i da imaju individualnost da prenesu na autorska djela. Jasno je i univerzalno prihvaćeno da računar ne može da bude autorom djela već samo objektom prava na kom autor djeluje u stvaranju autorskog djela. Uzimajući sve u obzir, apsolutno je jasno da u pozitivnom pravu računar, odnosno „umjetna inteligencija“ ne može biti autorom računarskog programa, pa se ova opcija u potpunosti isključuje.

B. Kriterij za određivanje autora AGCS rezultirajućeg koda

Moramo konstatovati da iako koristimo termin AGCS da označimo sve automatske generatore koda, to ne znači da su svi automatski generatori koda identični. Upravo suprotono, svi su na neki način različiti, što je osnova njihovog statusa kao autorskih djela.

Pored što su različiti, AGCS-ovi se razlikuju po nivou i detaljnosti instrukcija od strane korisnika za generisanje koda. AGCS-ovi se razlikuju i po detaljnosti i obimu završnog koda kog su sposobni da generišu autonomno, bez imputa korisnika. Sve ovo su važni faktori u određenju tko od lica koji su učestvovali u stvarnju računarskog programa korištenjem generatora koda su autori.

Držimo da ne postoji jedan ili jedinstven odgovor na ovo pitanje. Taj odgovor će zavisiti u svakom konkretnom slučaju. Te, da bi se utvrdilo koje od mogućih ponuđenih odgovora će biti primjenjen u kom slučaju postoji potreba da se analizira prvo AGCS gdje je potrebno da se upozna sa njegovom prirodom, a posebno je od velike važnosti da se utvrdi koliki je opseg autonomskog djelovanja generatora koda. Ilustrativno bi svakako bilo da se pogleda koliki je opseg autor generatora zamislio i implementirao u generator. Zanimljivo je i pogledati da li je rezultat unutar tih granica ili ih je na neki način prevazišao, jer bi to bio svakako veoma jak indikator da je za taj konkretni generator rješenje pod tačkom e)

Nakon toga potrebno je analizirati prirodu i obim doprinosa korisnika. Posebno, ovdje je potrebno pogledati koliki je nivo apstrakcije i konkretizacije ideje korisnika koji je unesen u generator, odnosno koliko je programa proizvedeno kao izravni rezultat izražaja ličnosti korisnika a koliki autora djela.

Ukoliko je veliki postotak rezultirajućeg koda automatski i autohtono generisan, te ukoliko je imput korisnika minimalna, onda bi, u odsustvu drugih eskternih čimbenika zaključili da je autor lice koje je načinilo AGCS.

Ukoliko je korisnik unjeo veliki dio konkretizacije i svog individualiteta u generator, a gdje je generator pratio jednostavan set uputstava, te je proizveo kod na osnovu pukih proceduralnih algoritama, velike su indikacije da je autor korisnik AGCS-a. Kao što je autor knjige napisane na Wordu autor djela a ne programer u Microsoftu, ili autor slike obrađene u Photoshopu korisnik a ne Adobe, tako je i u velikoj većini slučajeva autor korisnik a ne lice koje je načinilo AGCS. Držimo da je ovo pravilo, a drugi slučajevi izuzetci. Držimo da se treba napraviti oboriva pretpostavka autorstva za korisnika,

te da teret dokazivanja treba biti na sva druga lica koja tvrde da su autori.

Druge dvije eventuelnosti će zavisiti od nivoa interakcije korisnika i kroz AGCS autora programa. Ukoliko AGCS automatski i proceduralno generiše većinu neophodnog koda i korisnik modifikuje taj kod za vlastite potrebe u manjem obimu, rezultirajući kod se može smatrati djelom prerade u jednom obliku. Djelo prerade samo od sebe je autorsko djelo, koje je zaštićeno u obimu u kom je originalno i pod uslovom da je određeno od strane autora da je takva modifikacija dozvoljena.

Ukoliko su doprinosi autora AGCS-a i korisnika takvi da je nemoguće iz finalnog proizvoda izčlaniti njihove doprinose, odnosno ukoliko je nemoguće odstraniti ili rasčlaniti kod na njegove sastavne komponente i dobiti upotrebljive elemente, u tom slučaju vjerovatno je da će se rezultirajuće djelo smatrati koautorskim djelom.

V. ZAKLJUČAK

Automatski generatori koda se koriste da bi olakšali i napravili proces izrade računarskih porgrama učinkovitijim. Budući da je izvorni kod zaštićen autorskim pravom kao književno djelo, nameće se nekoliko autorskopravnih pitanja vezanih za njihovu upotrebu.

Na pitanje da li su AGCS-ovi računarski programi ili su po pravnoj prirodi bliži nezaštićenim programskim jezicima, zaključili smo da je evidentno da su računarski programi zbog svoje funkcionalnosti, odnosno činjenice da izvršavaju i imaju konkretnu svrhu i zbog svoje originalnosti.

Na pitanje autorstva rezultirajućeg koda zaključili smo da je objektivno nemoguće i pravno neopravdano dati jedinstven odgovor. Umjesto toga smo predložili elemente i aspekte na koje je potrebno obratiti pažnju prilikom svake individualne ad-hoc analize.

Kao rezultatse može dati predlog izmjene Zakona o autorskim i srodnim pravima BiH sa odredbom da se kod automatski generisanog koda autorom istog ima smatrati, po principu oborive pretpostavke ili *presumptio iuris tantum*, korisnik generisanog koda, s tim što se može argumentovati i da je priroda autorstva i njegova osoba drugačiji od toga.

LITERATURA

[1] Zakon o autorskim i srodnim pravima Bosne i Hercegovine (Službeni glasnik Bosne i Hercegovine, 63/10)

[2] Direktiva 2009/24/EC Evropskog parlamenta i Vijeća od 23. Aprila, 2009. godine o pravnoj zaštiti računarskih programa

[3] Mišljenje AG Yvies Bot-a povodom slučaja C-406/10 SAS Institute v. World Programming Ltd.

[4] Presuda Suda pravde Evropske unije povodom slučaja C-406/10 od 02. Maja, 2012. godine

[5] A. Abran, J. W. Moore, "Guide to the Software Engineering Body of Knowledge", IEEE Computer Society 2004.

[6] S. L. Pfleeger, J. Atlee, "Software Engineering: Theory and Practice", Prentice Hall 2006, New Jersey

[7] S. Kelly, J.P. Tolvanen, "Domain-Specific Modeling: Enabling full code generation", Wiley-IEEE Computer Society Press, 2008.

[8] J. Luoma, "Automating software development with domain-specific modelling", Medical Device Technology [Med Device Technol] 2004 Nov; Vol. 15 (9), pp. 46-7.

[9] D. C.Schmidt, "Model-Driven Engineering", Computer; Feb2006, Vol. 39 Issue 2, p25-31, 7p

[10] J. Herrington, "Code Generation in Action", Maning 2003

[11] X. Dianxiang, X. Weifeng, E. W. Wong, "Automated test code generation from class state models", International Journal of Software Engineering & Knowledge Engineering; Jun2009, Vol. 19 Issue 4, p599-623, 25p

[12] T. Rompf, M. Odersky, "Lightweight Modular Staging: A Pragmatic Approach to Runtime Code Generation and Compiled DSLs", Communications of the ACM; Jun2012, Vol. 55 Issue 6, p121-130, 10p

[13] Mišljenje AG Yves Bot-a povodom slučaja C-393/09 BSA v. Ministersto kulturny

[14] Presuda Suda pravde Evropske unije povodom slučaja C-393/09 BSA v. Ministerstvo kulturny

[15] Alija, I. (2011). Pravo intelektualnog vlasništva u Bosni i Hercegovini, Zakoni iz oblasti autorskog i srodnih prava i kolektivno ostvarivanje autorskog i srodnih prava Bosne i Hercegovine sa objašnjenjima i registrom pojmova. Travnik: Univerzitet u Travniku.

[16] Bengtsson, H. (2012, 06). *EU harmonisation of the copyright originality criterion.* Retrieved 06 22, 2012, from World Services Group: http://www.worldservicesgroup.com/publications.asp?action=article&artid=4597

[17] Besarović, V. (2005). Intelektualna svojina, Industrijska svojina i autorsko pravo, Četvrto, dopunjeno i izmenjeno izdanje. Beograd: Centar za publikaije Pravnog fakulteta u Beogradu.

[18] Marković, S. (2007). *Pravo intelektualne svojine.* Sarajevo: Magistrat.

[19] Sinodinou, T.-E. (n.d.). *Directive 91/250/EEC on software copyright protection: computer programs created by employees.* Retrieved 07 01, 2012, from

http://www.tatianasinodinou.eu/docs/Sinodinou_computer.pdf?PHPSESSID=9aacedc0019fdf7e19fd04854c388f90

ABSTRACT

The work covers the topic of automatic code generating systems or AGCS's. AGCS's are used to aid in the design and to create computer programming engineering efficiencies. However such application creates certain copyright implications. The work examines the legal nature of AGCS's. It asks the question weather they are actual computer programs or more akin to programming languages, which are not copyright protected. It concludes that they are computer programs because of their function and originality. The work also examines authorship issues related to AGCS use. It provides a test for determining authorship and makes a suggestion as to a creation of a legal presumption as to authorship in favor of the user of the AGCS.

SOME COPYRIGHT LAW ASPECTS OF AUTOMATIC CODE GENERATION

Haris Hasić, Vladimir Vujović