OTVORENI GENERATOR IZVEŠTAJA I SKRIPTI ZASNOVAN NA PARSERU WELL-FORMED XML DOKUMENATA

Vladimir Otašević[1,2], Biljana Kosanović[2]

1: Univerzitet u Beogradu – Elektrotehnički fakultet, Beograd, Srbija
2: Računarski centar Univerziteta u Beogradu, Beograd, Srbija
elektronska pošta: vladimir.otasevic@rcub.bg.ac.rs, biljana@rcub.bg.ac.rs

REZIME

Cilj ovog rada je da prikaže mogućnosti otvorenog generator izveštaja i skripti. Alat koji implementira generator nazvan je *WEXA* (skr. *Well-formed XML* parser). Softversko rešenje je nastalo kao rezultat neprekidne potrebe da se isti podaci iznova obrađuju. Kao krajnji proizvod alata, dobijaju se izveštaji različitih formata. Podaci koji su unapred pripremljeni i koji ispunjavaju uslov sintaksne ispravnosti odnosno imaju osobinu da su *well-formed*, mogu se parsirati ovim alatom. Uočena je mogućnost da se isti mehanizam može proširiti kako bi se podaci iskoristili za generisanje računarskih skripti, prevashodno napisanih u formatu *bash* skripti. Pored otvorenosti koda, a kao posledica odabranih tehnologija, omogućeno je transparentno praćenje izvršavanja svake komande ovog alata. Glavna odlika ovog alata ogleda se u jednostavnosti primene i lakoći proširivanja ili prilagođavanja generatora specifičnostima podataka ili korisnika.

Ključne reči: otvoreni generator izveštaja, javascript, otvoreni generator skripti, *well-formed XML*, parser *XML* dokumenta, transparentno izvršavanje, prilagođavanje alata.

1 Uvod

U eri primene računara i računarskih tehnologija, u gotovo svim sferama društva, neophodno je voditi računa kako se podaci čuvaju, ažuriraju i koriste. U najvećem broju slučajeva podaci imaju višestruku primenu i to omogućava korisnicima da ih upotrebljavaju na različite načine. Takođe, treba napomenuti da računari imaju važnu ulogu u tom sistemu. Računari nisu samo sredstvo za obradu podataka, već predstavljaju krajnje korisnike. Zbog postojanja interakcije čovek-računar, u ovom radu su pri implementaciji *WEXA* alata razmatrani različiti oblici mašinski čitljivih podataka.

Mašinski čitljivi podaci su važan deo *WEXA* alata. Za podatke koji ispunjavaju uslov da su mašinski čitljivi, postoji garancija da se kao takvi mogu deliti među mašinama [1][2]. Ako se još obezbedi da su mašinski čitljivi podaci i ljudski čitljivi, onda je interakcija čovek-računar u potpunosti podržana zahvaljujući njihovoj strukturi. Ovo je glavni razlog koji je uticao pri odabiru formata i strukture podataka čije će parsiranje biti predstavljeno.

WEXA je nastao usled svakodnevne potrebe da se podaci obrađuju od strane više korisnika zarad dobijanja različitih izveštaja. Kako su u celom procesu rada uključeni mnogi korisnici, koji se

međusobno razlikuju prema stepenu računarskih veština kao i radnih pozicija, bilo je neophodno realizovati alat koji je dovoljno intuitivan za svakoga od njih. Zbog same različitosti korisnika, različite su i potrebe za generisanjem izveštaja i skripti.

Alat je razvijen tako da svaki korisnik može preuzeti *WEXA* instancu i pokrenuti je lokalno na svom računaru, a jedino što je potrebno je postojanje internet pretraživača kod korisnika. Tehnologije koje su korišćene za realizaciju ovog alat su *JavaScript*, *HTML* i *CSS*. Korišćene tehnologije su osnovne komponente većine današnjih alata.

2 Mašinski čitljivi podaci

Različite su kategorije i podkategorije podataka koje se mogu smatrati mašinski čitljivim. Po pravilu, podaci koji su mašinski čitljivi su zasigurno i struktuirani podaci [3]. Glavna podela koja postoji među mašinskim čitljivim podacima jeste da li imaju osobinu i da su ljudski čitljivi. Podaci koji su zapisani u formatu kao što su *.csv*, *.xml*, *.json* pripadaju samo klasi data fajlova i ne smatraju se ljudski čitljivim. Postoje različiti interpretatori navedenih formata, koji mogu da povećaju stepen ljudske čitljivosti, ali po pravilu ti formati i dalje nisu ljudski čitljivi.

Sa druge strane postoje fajlovi koji mogu biti *.html* ili *.xslt* koji se smatraju ljudski čitljivim podacima. U osnovi ako se posmatra sintaksna struktura *.xml* i *.html* koda, onda ne postoji razlika između ova dva formata. Ipak, zbog semantičke logike koja postoji u *.html* zapisima i preciznije definisanih pravila pisanja *.html* fajlova, smatra se da su *.html* zapisi mašinski i ljudski čitljivi [4].

Navedena osobina koja proističe iz sličnosti *.html* i *.xml* fajlova je prvi razlog zbog kog je *.xml* format uzet u razmatranje za primenu. Razmatrano je korišćenje *.json* formata iz razloga što je taj format jednostavnije koristiti sa strane alata, imajući u vidu da je alat prevashodno napisan na *JavaScript* programskom jeziku. Ipak, *.json* ne podržava koncept metapodataka, pa zbog toga nije dalje razmatran [5].

Fajlovi koji su u *.xml* formatu se smatraju *well-formed* ako ispunjavaju uslov da su sintaksno ispravni [6]. Sintaksna ispravnost je lako proverljiva pomoću internet pretraživača. Ako nešto nije uredu sa fajlom, pretraživač će ukazati precizno na grešku. Pravila koja moraju da se ispoštuju kako bi se *.xml* fajl smatrao da je *well-formed* jesu da je sadržaj razdvojen tagom za početak i kraj, da je definisan sadržaj i da je sadržaj hijerarhijski struktuiran. Navedena pravila su samo najosnovnija pravila koja se moraju ispoštovati. Sa druge strane postoje pravila koja se definišu na nivou entiteta. Neka od tih pravila definišu tagove kao *case-sensitive* elemente, kao i da se vrednosti atributa moraju naći pod znacima navoda i da nema preklapanja tagova [7][8].

Postoji još pravila kao i smernica kako treba pisati mašinski čitljive dokumente kao *.xml* fajlove. Skup pravila, kao i osobine *.xml*, mogu se proširiti zahvaljujućim šemama koje preciznije definišu semantičku i sintaksnu ispravnost *.xml* fajlova. Šeme definišu pravila na osnovu različitih segmenata primene *.xml* fajlova. Za korišćenje *WEXA* alata dovoljno je da je *.xml* fajl *well-formed*, odnosno da prolazi sintaksnu proveru od strane internet pretraživača.

3 Korišćene tehnologije i implementacija

WEXA alat je razvijen kao internet aplikacija pomoću JavaScript, *HTML* i *CSS* tehnologija. Glavni razlog za primenu odabranih tehnologija jeste njihova praktičnost i jednostavnost prilikom razvoja. Korišćene tehnologije su podržane, barem osnovne komponente, kod većine internet pretraživača [9]. Zbog toga su promene uočljive u realnom vremenu. Sa druge strane nije potrebno posebno razvojno okruženje kako bi se napisala aplikacija koja koristi predložene tehnologije. Dovoljan je jednostavan tekstualni editor, kao što je *Notepad* ili *Pluma*.

Zbog jednostavnog koncepta održavanja, nije neophodno napredno poznavanje internet tehnologija i složenih programerskih paradigmi. Ova osobina ujedno omogućava da se alat može specijalizovati za potrebe obavljanja jednog konkretnog posla. Primer je dodavanje posebnih filtera ili čitanje specifičnijih *.xml* fajlova.

Internet pretraživači predstavljaju okruženje za pokretanje *WEXA* alata. To omogućava da proces izvršavanja bude u potpunosti transparentan te krajnji korisnik ima potpunu kontrolu nad programskim kodom. Takođe, zbog prirode korišćenih tehnologije kao i internet pretraživača kao okruženja za izvršavanje, omogućeno je da korisnici mogu *WEXA* alat preuzimati sa interneta, prilagoditi svojim potrebama i izvršavati lokalno na svom internet pretraživaču.

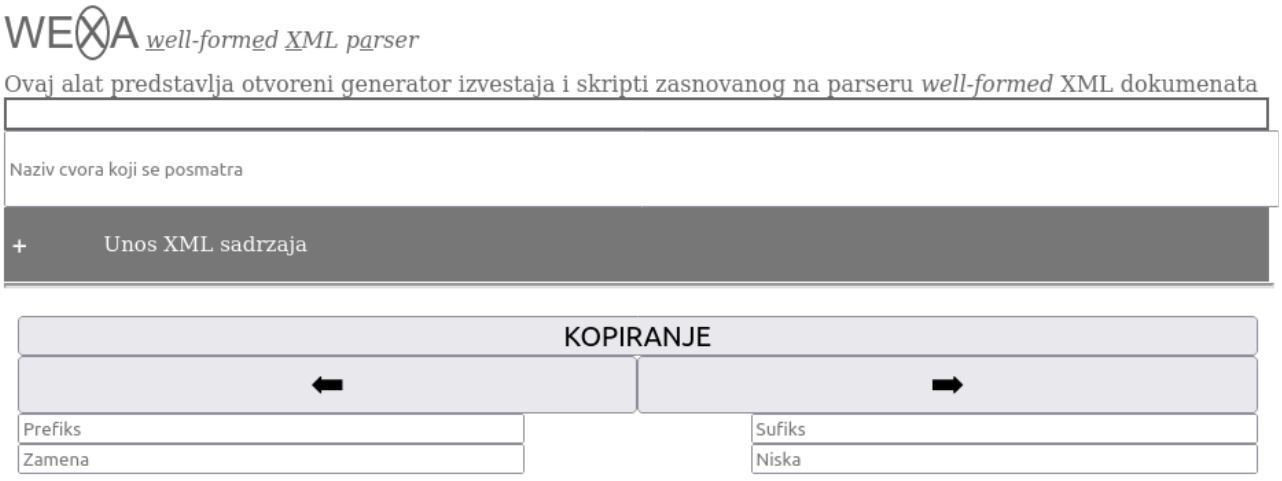
3.1 Pregled funkcionalnosti *WEXA* alata

WEXA alat je podeljen na tri komponente. Svaka komponenta predstavlja zasebnu logičku celinu pri čemu postoji veza između svake komponente. Komponente alata su:

1. Panel za upis *XML* zapisa;
2. Panel sa ekstrahovanim entitetima;
3. Radni okvir za generisanje izveštaja i skripti.

Panel za upis *XML* ne proverava da li je zapis *well-formed*. Od korisnika se očekuje da će uneti zapis koji je valjan. Ako postoji greška, neće doći do ekstrahovanja tagova i neće se nastaviti procedura generisanja izveštaja i skripti. U okviru panela za upis *XML* zapisa postoji polje za unos naziva čvorova koje treba ekstrahovati. Za uspešno parsiranje neophodno je da svaki čvor ima sve tagove koji se ekstrahuju, čak i kada je ono prazno. Svi tagovi koji se ekstrahuju moraju imati atribut pod nazivom *type*, jer on reprezentuje naziv posmatrane vrednosti.

Nakon uspešnog unosa, u okviru panela sa ekstrahovanim entitetima formiraće se tabela. Svako polje tabele sadrži dugme koje reprezentuje sve vrednosti nekog entiteta. Pritiskom na dugme vrednosti se ispisuju u radnom okviru. Vrednosti jednog pojedinačnog taga se ispisuju u zasebnim redovima. Redovi su sortirani na osnovu taga koji je prvi ispisan, odnosno na osnovu vrednosti prvog dugmeta koje je pritisnuto. Sortiranje se vrši u alfabetskom poretku. Izgled početnog ekrana dat je na Slici 1.



Slika1 Izgled početnog ekrana *WEXA* alata.

Radni okvir se koristi za rad sa podacima. Postoji nekoliko funkcionalnosti koje se mogu primeniti kako bi se generisao željeni rezultat, i to:

1. Kopiranje trenutnog stanja;
2. Vraćanja prethodnog stanja i ponovno izvršavanje stanja. Operacije poznate kao *Undo* i *Redo*;
3. Dodavanje prefiksa;
4. Dodavanje sufiksa;
5. Pronalaženje niske i njena zamena novom vrednošću. Operacija poznata kao *Find & Replace*.

Svaka promena uneta kroz bilo koju funkcionalnost je vidljiva u realnom vremenu. Da bi promene bile sačuvane neophodno je dvaput kliknuti na primenjenu funkcionalnost i promena će se izvršiti i sačuvati. Sačuvane promene predstavljaju osnovni tekst. Kako bi se obezbedila logička podela između prefiksa, osnovnog teksta i sufiksa uvedene su zelena, plava i crvena boja teksta, respektivno. Omogućeno je da se prefiks, a što je još važnije, i sufiks sačuvaju kao deo osnovnog teksta. To omogućava da se kombinuje unos kao što je prefiks-tag-sufiks-tag-sufiks.

Slika2 Primer komande za kopiranje koja kombinuje dodavanje taga i sufika.

Alat može generisati različite formate izveštaja. Unosom separatora u polje za sufiks definiše se tip izlaznog fajla. Ako je potrebno generisati *.csv* fajl, onda će se u sufiks uneti znak zareza kao separator.

Kad je reč o skriptima, *WEXA* je veoma koristan u slučaju da je potrebno ažuriranje većeg broja mašina ili alata. Kao dobar primer može poslužiti slučaj kada je potrebno novu verziju nekog računarskog fajla postaviti na nekoliko različitih mašina. Ovakva vrsta skripte može se dobiti u svega nekoliko koraka, a najvažniji su:

1. Dodavanjem prefiksa kojim se definiše komanda i putanja do računarskog fajla;
2. Unos vrednosti taga koji reprezentuje putanju do mašina;
3. Dodavanjem sufiksa kojim se definiše odredišna putanja u okviru ciljane mašine.

Nakon generisanja skripte potrebno je sadržaj sačuvati u okviru lokalnog fajla, dodeliti mu dozvolu za izvršavanje i pokrenuti.

4 Zaključak

Implementirani alat je praktično rešenje realizovano kako bi pojednostavilo rešavanje čestih zadataka koji zahtevaju generisanje različitih izveštaja ili skripti. Najčešće su to slučajevi kada se jedan fajl propagira na različite mašine, kada se sa mašina preuzimaju podaci ili kada se upoređuju različite karakterstike aplikacija. Alat je proširen i prilagođen za korišćenje od strane softverskih inženjera koji neretko imaju potrebu da pokreću najrazličitije skripte za potrebe tehničkog održavanja i unapređenja računarske infrastrukture.

Zbog odabira široko primenjivih tehnologija, alat je lako proširiv. Omogućena je laka prenosivost i instanciranje alata ako se ukaže potreba za dodatnim proširenjima ili prilagođavanjima radi obavljanja specifičnih zadataka. Intuitivan dizajn omogućava praktično korišćenje alata bez obzira

na računarske veštine korisnika. Takođe, odabrane tehnologije podržavaju koncept otvorenosti i transparentnosti računarskog koda kao i njegovog izvršavanja.

Zahvalnica

Autori su zahvalni Milici Ševkušić, bibliotekarki u Tehničkom institutu SANU i prof. Nadici Miljković na komentarima i podršci prilikom pisanja ovog rada.

Rad je napisan u okviru projekta *NI4OS-Europe – National Initiatives for Open Science in Europe (H2020 no. 857645).*

Dostupnost podataka i softverskog koda: *WEXA* je dostupan na adresi wexa.rcub.bg.ac.rs

Literatura

[1] Buchanan EM, Crain SE, Cunningham AL, Johnson HR, Stash H, Papadatou-Pastou M, Isager PM, Carlsson R, Aczel B. Getting Started Creating Data Dictionaries: How to Create a Shareable Data Set. Advances in Methods and Practices in Psychological Science. January 2021. doi:10.1177/2515245920928007

[2] Lakens D, DeBruine LM. Improving Transparency, Falsifiability, and Rigor by Making Hypothesis Tests Machine-Readable. Advances in Methods and Practices in Psychological Science. April 2021. doi:10.1177/2515245920970949

[3] Open Data Handbook. Machine readable. http://opendatahandbook.org/glossary/en/terms/machine-readable/

[4] Philip J, Richard GC, Lennart M, Antony FQ, Chris FT, William D, Henning H, Rolf A, PRIDE: a public repository of protein and peptide identifications for the proteomics community, Nucleic Acids Research, Volume 34, Issue suppl_1, 1 January 2006, Pages D659–D663, https://doi.org/10.1093/nar/gkj138

[5] Wikipedia contributors. "JSON." Wikipedia, The Free Encyclopedia. Wikipedia, The Free Encyclopedia, 5 Jul. 2021. Web. 15 Jul. 2021.

[6] "Well formed XML documents". JCommerce Dev Network. August 22, 2009.

[7] Grijzenhout S, Marx M. The quality of the XML web. Journal of web semantics. 2013 Mar 1;19:59-68.

[8] Extensible Markup Language (XML), http://www.w3.org/TR/REC-xml.

[9] C. Severance, JavaScript: Designing a Language in 10 Days, Computer, vol. 45, no. 2, pp. 7-8, Feb. 2012, doi: 10.1109/MC.2012.57.