Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.42:004.738.12
DOI: https://doi.org/10.24867/29BE32Rajtarov

JEZIK ZA SPECIFIKACIJU PRODAJNIH AKCIJA

A LANGUAGE FOR THE SPECIFICATION OF SALES PROMOTIONS

Nataša Rajtarov, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIČKO I RAČUNARSKO INŽENJERSTVO

Kratak sadržaj – *U ovom radu će biti predstavljen jezik za dinamičko kreiranje prodajnih akcija, tokom rada aplikacije, ali i kreiranje same veb aplikacije. Fokus rada je na prikazu mogućnosti i prednosti ovog jezika u odnosu na postojeća rešenja poput Vtex-a, Drools-a i JHipster-a. Cilj je da se pokaže kako kreirani jezik omogućava veću fleksibilnost i smanjuje ograničenja za prodavce prilikom kreiranja akcija, za razliku od postojećih rešenja koja se oslanjaju na šablone.*

Ključne reči: *textX, python, jezik specifičan za domen, prodajne akcije*

Abstract – *This paper presents a language for the dynamic creation of sales promotions during application runtime, as well as for the creation of the web application itself. The focus of the paper is on showcasing the capabilities and advantages of this language compared to existing solutions like Vtex, Drools, and JHipster. The goal is to demonstrate how the developed language provides greater flexibility and reduces limitations for sellers when creating promotions, unlike existing solutions that rely on templates.*

Keywords: *textX, python, domain specific language, prodajne akcije*

1. UVOD

Razvojem tehnologije, mnoge aktivnosti svakodnevnog života, poput trgovine, migrirale su na veb platforme. Prelazak na veb, omogućio je trgovcima da širi broj ljudi može da dobije uvid u njihovu delatnost. Pored mnogih prednosti, veb platforme su donele i ograničenja, kao što je ograničenje u raznolikosti prodajnih akcija. Potreba za rešenjem ovog problema se javlja kada prodavci žele da kreiraju jedinstvene akcije koje nisu unapred definisane ili čije ponašanje nije opisano u potpunosti kodom unutar aplikacije.

Zadatak ovog rada je implementacija jezika koji omogućava dinamičko kreiranje prodajnih akcija tokom rada aplikacije. Jezik se sastoji iz dva ključna dela: prvi deo koji omogućava kreiranje veb aplikacije za prodaju, dok drugi deo omogućava kreiranje akcija u realnom vremenu. Pored toga, rad analizira postojeća rešenja kao što su Drools, VTex koji omogućavaju kreiranje prodajnih akcija uz pomoć šablona.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio prof. dr Igor Dejanović.

Osnovni motiv za razvijanje jezika koji omogućava kreiranje prodajnih akcija u toku rada aplikacije jeste problem potrebe za šablonima koji ograničavaju prodavca prilikom kreiranja prodajnih akcija. Razlog podele jezika na dva dela, jeste omogućavanje korisnicima da efikasno kreiraju aplikaciju koja će im omogućiti trgovinu, kao i obezbediti platformu na kojoj mogu uz jezik opisan u ovom radu da kreiraju akcije.

2. JEZICI SPECIFIČNI ZA DOMEN

Kako bi računar izvršavao zahteve od strane čoveka, razvijen je jezik koji omogućava međusobnu komunikaciju. Postoje jezici opšte namene (JON) i jezici specifični za domen (JSD).

Jezici opšte namene, kao što su Python, Java, i JavaScript, omogućavaju kreiranje struktura i opisivanje ponašanja sistema u različitim aplikacijama i industrijama. Programski jezici opšte namene dizajnirani su da budu dovoljno fleksibilni i ekspresivni da mogu rešavati probleme nezavisno od domena kojem pripadaju. Međutim, njihova složenost može predstavljati izazov za korisnike koji nisu programeri, jer su dizajnirani da rešavaju širok spektar problema, što može otežati njihovu primenu u specifičnim domenima.

Za razliku od jezika opšte namene, jezici specifični za domen su usko orijentisani na jedan domen, što znači da su konstruisani tako da ispunjavaju zahteve samo određenog spektra zadataka, ali to rade uz povećanu produktivnost. Takav jezik bazira se na postojećem jeziku eksperata što olakšava ekspertima samo savladavanje tog jezika. Jezici specifični za domen su ograničene ekspresivnosti i usmereni su na određenu oblast primene, kao što objašnjava Martin Fovler [1].

Istraživanja kao što su [2, 3] pokazuje da korišćenje jezika specifičnih za domen povećava produktivnost kod njenih korisnika. „Sa odgovarajućim JSD-om, možete razviti kompletne aplikativne programe za domen brže i efikasnije nego sa jezikom opšte namene.“ [2] Ovakvi jezici omogućavaju brži razvoj aplikacija jer smanjuju količinu koda potrebnu za izražavanje određenih akcija u poređenju sa jezicima opšte namene. Pored toga, eksperti više nisu u obavezi da uče jezik koji nije u skladu sa njihovim domenom, čime se unapređuje efikasnost i tačnost. Uži domen i jezik prilagođen njemu, smanjuje nivo slobode koju jezik može da ostvari, ali samim tim i smanjuje verovatnoću grešaka, kao i ponavljanje koda. JSD-ovi su lakši za razumevanje nego JON jezici i nude mogućnost analize, verifikacije, optimizacije, paralelizacije i transformacije [4].

2. OSNOVNI KONCEPTI JEZIKA

Proces kreiranja novog jezika specifičnog za domen obuhvata 4 koraka:

- Definisanje domena – potrebno je tačno odrediti domen koji će obuhvatati jezik.
- Dizajn jezika i njegove semantike – U okviru ove tačke potrebo je razmotriti koje tačno funkcionalnosti treba da obuhvati dati jezik, šta će biti izlaz/produkt njegovog rada, ko će ga koristiti i na koji način da mu se prilagodi takav jezik. Modeluju se ograničenja koja će morati da podrži.
- Kreiranje alata koji će obrađivati jezik – Potebno je kreirati alat koji će omogućiti prevod i korišćenje jezika.
- Korišćenje jezika.

Da bismo omogućili efikasno modelovanje, kako jezika specifičnog za domen tako i jezika opšte namene, potrebno je da koristimo 2 nivoa sintakse, a to su aprstraktna i konkretna sintaksa. Apstraktna sintaksa se odnosi na strukturu jezika, nezavisno od specifičnih detalja konkretne sintakse. Ona opisuje šta se izražava u jeziku, a ne kako se izražava. Jedna apstraktna sintaksa može imati više svojih konkretnih sintaksi. Apstraktno sintaksno stablo je glavna struktura koja omogućava predstavu apstraktne sintakse. konkretna sintaksa omogućava ljudima da pišu i čitaju dati jezik. Ona obuhvata konkretne operatore, ključne reči, simbole, ili može imati i grafičku sintaksu. Ona određuje pravila kako se apstraktni pojmovi izražavaju kroz nju. Jezik se opisuje gramatikom jezika. Gramatika jezika predstavlja formalnu definiciju konkretne sintakse.

3. JEZIK ZA KREIRANJE E-KOMERC SAJTOVA I DEFINISANJE PRODAJNIH AKCIJA

Budući da kompanije mogu da imaju različite domene trgovine, neće svima odgovarati isti tip akcije. Da bi se omogućilo fleksibilno kreiranje akcija, stvoren je jezik koji je opisan u ovom radu. Sam jezik se sastoji iz dva dela:

- Deo za kreiranje prodajnih akcija
- Deo za kreiranje web sajtova koji podržavaju kreiranje akcija pomoću datog jezika

Za implementaciju jezika korišen je textX metajezik [5] koji omogućava istovremeno definisanje i konkretne i apstraktne sintakse upotrebom jedne gramatike. Izlaz iz celog procesa je generisan kod koji je spreman da obavlja svoju namenu. Faze kroz koje prolazi program/model su:

1. Leksička analiza
2. Sintaksna analiza
3. Semantička analiza
4. Generisanje koda

Leksička analiza podrazumeva obradu ulaznog koda, tako da se razbije na lekseme, manje jedinice koda. Ona identifikuje ključne reči, identifikatore, literale i druge simbole. Izlaz su lekseme koje se u sintaksnoj analizi obrađuju. Proverava se sintaksa jezika i lekseme se organizuju u sintaksno stablo. U okviru semantičke analize obrađuje se i proverava sintaksno stablo da li je semantički ispravno. U ovoj fazi se proveravaju tipovi podataka, provere opsega i ostale provere vezane za sintaksu. Poslednja faza omogućava generisanje koda.

Prevođenje koda i generisanje u ovom projektu, opisano je na programskom jeziku Python. Međutim, za generisanje koda veb aplikacije, korištena je Jinja, budući da omogućava kreiranje koda kroz šablone. Generisana aplikacija se sastoji iz dva sloja: serverski i klijentski sloj. Za serverski sloj je korišten Springboot sa javom. Za komunikaciju sa bazom je korišten Nibernit. Klijentski sloj aplikacije je pravljen u React-u. Komunikacija između serverskog i klijentskog sloja je obavljana pomoću REST-a.

3.1. TextX

textX je alat razvijen u Python-u sa glavnim ciljem brzog razvoja jezika specifičnih za domen [5]. Zasnovan je na Arpeggio PEG parseru, što mu omogućava neograničeno gledanje unapred sa linearnim vremenima parsiranja i rad u stilu interpretera. textX nasleđuje prirodu Arpeggio parsera, ali i gramatiku višeg nivoa [5]. Nastao je kao implementacija Xtext meta-jezika u Python-u, ali se vremenom razvio i danas sadrži neke karakteristike drugačije od Xtext jezika. Osnovni radni proces i arhitektura textX alata uključuju sledeće korake:

1. Parsiranje gramatike jezika i dinamičko stvaranje meta-modela i Arpeggio parsera.
2. Parsiranje programa/modela u datom jeziku i kreiranje grafa Python objekata, gde je svaki objekat instanca određene klase iz meta-modela jezika.
3. Model može kasnije biti korišćen za interpretaciju i/ili generisanje izvornog koda.

textX gramatika se sastoji od skupa pravila koja opisuju koncepte iz ciljnog jezika (tj. meta-klase) i sintaksne strukture koncepata predstavljenih korišćenjem tekstualne konkretne sintakse. Sva pravila rezultiraju kreiranjem python klase u vreme izvršavanja. Kreira se graf python klasa, koje će biti korištene za instanciranje datog koncepta tokom parsiranja modela. Model se kasnije može koristiti za interpretiranje ili generisanje koda.

3.2. Gramatika za modelovanje veb aplikacija

Gramatika definiše pravila koja su dovoljna da se opiše cela aplikacija. Kako bi se kreirala veb aplikacija sa osnovnim CRUD operacijama(create-kreiraj, read-čitaj, update-obnovi, delete-obriši), dovoljno je uneti podatke o domenskim klasama na osnovu kojih bi se definisali ostali slojevi aplikacije, repozitorijum, servis, kontroler, kao i klijentska strana. Zbog toga ovaj deo gramatike omogućava definisanje klasa, veza između njih, prostih atributa i enumeracija. Kako bi omogućili kreiranje veb prodavnice, omogućeno je opisivanje određenih klasa anotacijama koje omogućavaju proširenje tih klasa kroz dodate atribute i metode. Anotacije su:

- Bill - Anotacija će označiti klasu, čiji objekat se kreira prilikom potvrde kupovine. Izvršiće se obračun kupljenih stvari, pronaći će se odgovarajuća akcija i primeniti na cenu. Anotacija dobija nazive funkcija koje će se pozivati prilikom primene akcija.

- Bying - Anotacija će označiti klasu, čiji objekti će se naći u računu i čija cena će se obračunati prilikom kupovine. Anotacija može dobiti dve vrednosti: one ili more što omogućava kupovinu samo jednog ili više artikala.
- Item - Anotacija će označiti klasu produkta. Svaki produkt imaće listu objekata koji sadrže cenu, tj. onih objekata koji su pod anotacijom Bying.
- Basket – Anotacija će označiti klasu koja čuva objekte koji će se naći u korpi u toku kupovine. Kada se kreira račun, korpa će se isprazniti.
- Action – Anotacija će omogućiti oznaku klase koja će biti akcija. Dobiće podatke od kad do kad važi akcija i kod koji je menadžer/prodavac uneo kao i atribut prevedeni kod.

Proizvodna akcija se opisuje kao niz instrukcija koje dovode do promene cene. Da bi se obavila akcija, moguće je posmatrati listu proizvoda koji su trenutno na akciji, proizvode koje potrošač trenutno kupuje i samog potrošača. Proizvodi koji su trenutno na akciji mogu se naći u klasi koju je kreator aplikacije označio sa anotacijom Action. Jedna akcija predstavlja funkciju od nula ili više iskaza. Kroz ovaj jezik moguće je kreirati promenljive, kreirati jednolinijske uslove, if-else uslove, nove objekte, ali ključni deo jezika su „kratke petlje“ koje omogućavaju iteriranje kroz liste elemenata uz različite uslove. Na primer, „kratka petlja“ max će naći najveći element u listi bez da se eksplicitno piše standardna for petlja sa uslovima kao u JON jezicima. Gramatika dela jezika za kreiranje akcija, napisana u textX nalazi se na slici Slika 1.

```
ChaneId: ID ('.' ID)*;

Sum:        'sumOf'       PosibleLists             (('as:' (ChaneId)   (UsingFormula)?   WherePart)?);
Average:    'averageOf'   PosibleLists             ('as:' ( ChaneId  ) (UsingFormula)?   WherePart)?;
Product:    'productOf'   PosibleLists             ('as:' ( ChaneId  ) (UsingFormula)?   WherePart)?;

SelectTop:  'selectTop'   INT 'of' PosibleLists    'as:'  ID           (WherePart)?;
SelectIn:   'selectIn'    PosibleLists             'as:'  ID            WherePart;
Count:      'countFrom'   PosibleLists             ('as:' ID)?         (WherePart)?;
All:        'allOf'       PosibleLists             ('as:' ID)?          WherePart ;
Any:        'anyOf'       PosibleLists             ('as:' ID)?          WherePart;
NoOne:      'noneOf'      PosibleLists             ('as:' ID)?          WherePart;
Min:        'min'         PosibleLists             ('as:' (ChaneId))?  (WherePart)?;
Max:        'max'         PosibleLists             ('as:' (ChaneId))?  (WherePart)?;
Size:       'sizeOf'      PosibleLists ;
Add:        'add'         (ChaneId)                'in:' PosibleLists;
ForEach:    'for'         ( ChaneId)            'as:' ( ChaneId) (ForOperation)?  ShortFunctions;

ForOperation: ('sumFrom'|'averageFrom'|'productFrom' );
PosibleLists: ( ChaneId | SelectIn);
WherePart: 'where'  (Conditions | All| Any | NoOne);
UsingFormula: 'formula' Operations;
ShortFunctions: ( Size| Product | Count | All | Any | NoOne | Min | Max | Add |SelectTop| Sum |Average | SelectIn | ForEach);
value: (INT | STRING | FLOAT | BOOL );
```

Slika 1*"kratke petlje" gramatike opisane u textX-u*

4. IZLAZNA APLIKACIJA

Izlazna veb-shop aplikacija sadrži tri uloge: administratora, prodavca i kupca. Administrator može da vrši CRUD operacije nad svim entitetima, prodavac može da rukuje sa objektima vezanim za proizvode, da kreira cene i akcije, dok kupac može da naručuje proizvode, pregleda korpu, svoje podatke i sl.

Proces kreiranja akcije se sastoji od unosa podataka o akciji, kao što su datumi od kad do kad traje i slično, ali i koda akcije. Kod akcije napisan u jeziku za kreiranje akcija, šalje se na python serveru koji vrši prevod akcije. Prevedeni kod se vraća na serverski sloj aplikacije. Gde se čuvaju podaci o akciji, kao i sam prevedeni kod. Korisnik se obaveštava o uspešnosti kreiranja akcije.

Sam proces naručivanja obuhvata odabir proizvoda i potvrdu kupovine od strane kupca. Podaci se šalju na serversku stranu gde se vrši obračun cena proizvoda i obračunavaju se akcje. Prilikom obračuna akcije, kreira se fajl koji čuva klasu sa vraćenim kodom funkcije. Klasa se čita i pomoću refleksije se izvršava kod koji je preveden. Refleksija omogućava da program pregleda, poziva metode ili modifikuje strukturu i ponašanje objekata u toku izvršavanja programa. Budući da se šalju reference na objekte prevedonj funkciji, objekat račun se vraća sa novom cenom, nastalom primenom akcije. Podaci se čuvaju u bazi i šalje se obaveštenje korisniku.

5. PREGLED STANJA U OBLASTI

5.1. Drools

Drools se opisuje kao sistem za upravljanjem poslovnim pravilima, koji omogućava ulančavanje unapred i unazad (forward-chaining i backward-chaining) [22]. Koristi DRL (Drools Rule Language) za definisanje pravila koja omogućavaju rezonovanje na osnovu činjenica. DRL jezik može opisati pravila za bilo koji domen primene, ne ograničavajući se samo na trgovinu. Pored kreiranja pravila, Drools omogućava i kreiranje šablona, koji služe kao šabloni za unos pravila kroz aplikaciju, bez potrebe za učenjem samog Drools jezika. Međutim, ovaj pristup može biti kompleksan ako je potrebno definisati više šablona za različite tipove akcija. Drools je najbolje rešenje kada korisnik već zna DRL jezik ili kada postoji

ograničen broj akcija koje se mogu definisati šablonima. Problemi nastaju kada korisnik mora da uči DRL jezik ili želi da definiše širok spektar akcija.

5.2. Vtex

Vtex je onlajn platforma koja omogućava upravljanje online trgovinom. Koristi cloud tehnologiju, što omogućava skalabilnost, tj. da prodavnice mogu da porastu i još uvek sačuvati svoje performanse. Omogućava povezivanje sa drugim platformama kao što su Amazon i eBay. Pored svega toga, Vtex omogućava kreiranje promocija i akcija. Postoji više vrsta akcija koje podržava, a neke od njih su: popusti na proizvode, „Kupite jedan, dobijete jedan besplatno.“, količinski popust, promo kodovi itd. Prilikom kreiranja, kupac bira jedan od tipova akcije gde mu se nudi šablon sa podacima u vezi akcije, koji popunjava i čuva akciju. Ono što razlikuje kreiranje akcija u ovoj aplikaciji od jezika koji je opisan u ovom radu, je to da postoji set podataka koji prodavci mogu da iskoriste za svoje akcije. Taj skup podataka koji se unosi jeste širok i omogućava kreiranje različitih prodajnih akcija, ali nema slobodu koju nudi jezik. Pomoću jezika je moguće opisati širi skup akcija, jer je moguće pristupiti većem broju podataka o samom korisniku, njegovoj korpi sa proizvodima.

5.3. Jhipster

JHipster je generator koda i podržava kreiranje veb aplikacija i mikroservisa gde za serversku stranu generiše kod u Spring okviru, a klijentsku u Angular-u i React-u. Podržava i MVC šablon na serverskoj strani tako da je izgenerisana aplikacija je podeljena na model, kontroler, dok je za izgled zadužena klijentska strana. Jhipster koristi JDL (JHipster Domain Language) za modelovanje entiteta i odnosa u aplikaciji. JDL jezikom je moguće opisati entitete koje korisnik želi da ima u svojoj aplikaciji i da definiše atribute i veze između entiteta. Dok JDL omogućava osnovno modelovanje entiteta, jezik opisan u ovom radu nudi anotacije koje pružaju detaljnije opise entiteta, čime se omogućava bolji opis aplikacija čiji je domen trgovina. JHipster i njegov JDL pružaju dobar okvir za brzo kreiranje veb aplikacija, ali novi jezik i njegove anotacije predstavljaju korak napred u prilagođavanju alata specifičnim potrebama trgovinske industrije.

6. ZAKLJUČAK

Rad obuhvata razvoj jezika za modelovanje aplikacija za online trgovinu, koji omogućava kreiranje prodajnih akcija. Ovaj jezik je dizajniran kao jezik specifičan za domen kreiranja e-komerc sajtova. Rad opisuje proces kreiranja jezika, uključujući definisanje domena, modelovanje jezika, razvoj alata za podršku jezika, te generisanje aplikacija i akcija korišćenjem jezika. Korišćeni su različiti alati i tehnologije kao što su Python, textX, Java, Spring Boot i React, za realizaciju sintakse, semantike, i generisanje koda. Jezik omogućava modelovanje klasa i relacija, dodavanje specifičnih anotacija za proširenje funkcionalnosti, i generisanje web aplikacija koje sadrže serverski i klijentski sloj. Jezik omogućava kreiranje različitih prodajnih akcija, sa većim stepenom fleksibilnosti u odnosu na postojeća rešenja poput Drools, Vtex, i Shopify. Dalji razvoj jezika usmeren je ka praćenju novih trendova u online trgovini i unapređenju tehnologija korišćenih u razvoju.

7. LITERATURA

[1] Fowler, Martin. „*Domain-specific languages“* Pearson Education, 2010.

[2] Hudak, Paul. *"Domain-specific languages."* Handbook of programming languages 3.39-60 (1997): 21.

[3] Sujeeth, Arvind Krishna. *„Productivity and Performance with Embedded Domain-Specific Languages*.“ Stanford University, 2014.

[4] M. Mernik, J. H. „*When and how to develop domain-specific.“* (2005).

[5] Dejanović, I. Vaderna, R., Milosavljević, G., Vuković, Ž. „*Textx: a python tool for domain-specific languages implementation. Knowledge-based systems.“* (2017).

Kratka biografija:

Nataša Rajtarov rođena je u Zrenjaninu 2000. god. Radi kao saradnik u nastavi na Fakultetu tehničkih nauka. Master rad na Fakultetu tehničkih nauka iz oblasti Softverskog inženjerstva o informacionih tehnologija odbranila je 2024.god.
kontakt:
rajtarovnatasa@gmail.com