Зборник радова Факултета техничких наука, Нови Сад

UDK: 004.9
DOI: https://doi.org/10.24867/29BE19Akik

НАМЕНСКИ ЈЕЗИК И ОКРУЖЕЊЕ ЗАСНОВАНО НА ИНЖЕЊЕРСТВУ ВОЂЕНОМ МОДЕЛИМА ЗА ПРИСТУП РАЗЛИЧИТИМ ВЕКТОРСКИМ БАЗАМА ПОДАТАКА

DOMAIN-SPECIFIC LANGUAGE AND MODEL-DRIVEN ENVIRONMENT FOR ACCESSING DIVERSE VECTOR DATABASES

Елена Акик, *Факултет техничких наука, Нови Сад*

Област – ЕЛЕКТРОТЕХНИЧКО И РАЧУНАРСКО ИНЖЕЊЕРСТВО

Кратак садржај – *У овом раду представљено је окружење за униформни приступ различитим векторским базама података, формирано кроз моделом вођен развој софтвера. У оквиру окружења подржано је повезивање са циљном базом података, дефинисање структура података, манипулација објектима базе података, постављање упита и векторска претрага сличности. Централни елемент окружења је наменски језик uniVEC, са текстуалном синтаксом. Важан део окружења су генератори кôда, који трансформишу моделе креиране помоћу језика uniVEC у извршив програмски кôд. Генератори кôда генеришу одговарајуће Python скрипте за подршку рада са одабраном циљном векторском базом података, као што су: Pinecone, Milvus, Chroma, Weaviate или Qdrant. Предложено окружење има циљ да поједностави рад са векторским базама података, смањи време потребно за учење концепата из домена и обезбеди миграције између различитих векторских база података.*

Кључне речи: *векторске базе података, наменски језици, развој софтвера вођен моделима, масивни подаци, машинско учење*

Abstract – *In this paper, a model-driven environment for uniform access to various vector databases is presented. The environment supports connection to the target vector database, definition of data structures, manipulation of database objects, querying, and vector similarity search. The central element of the environment is the domain-specific language uniVEC, with a textual syntax. An important part of the environment is code generators, that transform models created by using uniVEC into executable programming code. The code generators generate Python scripts to support work with one of the chosen vector databases, such as: Pinecone, Milvus, Chroma, Weaviate, or Qdrant. The proposed environment aims to simplify work with vector databases, reduce learning time and support migrations between different vector databases.*

Keywords: *Vector Database, Domain-Specific Language, Model-Driven Software Development, Big Data, Machine Learning*

НАПОМЕНА:

Овај рад проистекао је из мастер рада чији је ментор био др Милан Челиковић, доцент

1. УВОД

Масивни подаци (енгл. *Big Data*) мењају начин живота и рада људи и кључни су покретач развоја у различитим областима. У дигиталном свету, количина генерисаних података расте великим интензитетом. Масивни подаци користе се за генерисање знања и боље доношења одлука [1]. Пут од генерисаног знања до формиране одлуке је комплексан и дуготрајан, а кључну улогу имају предикције, које уочавају обрасце, на основу којих је могуће формирати одлуке.

Потенцијални изазови приликом управљања великим количинама података јесу где складиштити и како обрадити податке. Додатно, како људи нису у стању ефикасно обрадити масивне податке и на основу њих генерисати знање, као подршка, деценијама уназад, коришћена је вештачка интелигенција (енгл. *Artificial Intelligence*). Руковање масивним подацима може бити изазовно са становишта меморијског простора за складиштења података, као и перформанси читања и ажурирања података.

При одабиру начина складиштења структуираних и неструктуираних података, чест избор су векторске базе података, погодне за складиштење података датих типова, али и за семантичку претрагу и индексирање података, обезбеђење скалабилности система и добрих перформанси приликом извршавања упита над подацима у бази података [2].

Упркос предностима које пружају, сложеност употребе векторских база података један је од кључних изазова. Примарни изазов представља овладавање концептима векторских база података, због њихове сложености и разноврсности. Тренутно не постоји униформни језика за рад са различитим векторским базама података. Због тога, крајњи корисници морају овладати техникама приступа и начинима интерпретације жељених акција над базом података, специјализованим за приступ одабраној векторској бази података, реализујући их кроз програмски кôд.

У овом раду, предложено је окружење за униформну интеракцију са различитим векторским базама података, где је кључни елемент наменски језик *uniVec*. Помоћу језика *uniVec* могуће је креирати моделе за приступ векторским базама података у виду скрипти, док креиране моделе генератори кôда трансформишу у извршив програмски кôд одабране векторске базе података. Предложено окружење има за циљ да обезбеди: поједностављење рада са циљном

векторском базом података кроз стандардизацију приступа; смањено време за учење концепата из домена и тестирање различитих векторских база података, као и редуковану сложеност приликом миграција између различитих векторских база података.

Након увода, у поглављу 2 дат је теоријски осврт на развој софтвера вођен моделима и наменске језике. Поглавље 3 усмерено је на преглед начина рада векторских база података. У поглављу 4 описан је наменски језик *uniVec,* а у поглављу 5 приказани су генератори кôда. У поглављу 6 сумиран је општи допринос имплементираног окружења..

2. ТЕОРИЈСКЕ ОСНОВЕ РАЗВОЈА СОФТВЕРА ВОЂЕНОГ МОДЕЛИМА И НАМЕНСКИХ ЈЕЗИКА

Развој вођен моделима (енгл. *Model-Driven Development – MDD*) представља парадигму која користи моделе као примарне артефакте процеса развоја система, укључујући аутоматизовано или полу-аутоматизовано генерисање имплементационих артефакта из апстрактних модела [3]. Начин креирања и управљања моделима зависи од конкретних потреба које треба да разреше. Развој софтвера вођен моделима (енгл. *Model-Driven Software Development – MDSD*) представља конкретизацију парадигме *MDD*, где су модели искоришћени за развој софтверских система.

Језгро парадигме *MDSD* јесу језици који омогућавају крајњим корисницима да концептуализују стварност у експлицитном облику, кроз текстуалну или графичку представу. Две класе језика представљају језици опште намене (енгл. *General-Purpose Languages*) и наменски језици (енгл. *Domain-Specific Languages*). Језици опште намене применљиви су у било ком домену рада. Наменски језици формирани су за подршку специфичног домена, како би крајњи корисници користили концепте из домена приликом креирања модела. Наменски језици најчешће се користе у оквиру парадигме *MDSD* [4].

Сваки наменски језик сачињен је из три компоненте: *апстрактне синтаксе,* где је приказана структура језика и начин комбиновања основних елемената; *конкретне синтаксе*, која пружа специфичну репрезентацију језика, а може бити текстуална или графичка; и *семантике*, којом је описано значење елемената или њихових комбинација, дефинисаних у оквиру језика.

3. ПРЕГЛЕД РАДА ВЕКТОРСКИХ БАЗА ПОДАТАКА

Векторске базе података служе као складишта за векторе, односно, излазне резултате алгоритама машинског учења. Такође, чувају мета-податке у вези са ускладиштеним векторима, омогућавајући ефикасну манипулацију подацима. Чувањем вектора, омогућен је поступак претраге сличности, усклађујући корисничке упите са одговарајућим векторима. Претраживање вектора подразумева израчунавање удаљености између вектора, користећи метрике попут косинусне сличности. Систем за управљање векторском базом података (енгл. *Vector DataBase Management* System – *VDBMS*) [2] израчунава удаљености између вектора корисничког упита и вектора складиштених у бази података, те као одговор на упит враћа најсличније векторе, у складу са метрикама сличности. Рад са векторским базама података може се посматрати кроз неколико целина – дефинисање структура података, манипулацију подацима, векторизацију, те векторску претрагу и филтрирање података.

У овом поглављу, представљен је преглед начина рада векторских база података, где је поглавље 3.1 усмерено ка разјашњењу структуре векторских база података, док су у поглављу 3.2 анализиране подржане операције над подацима у векторској бази података. Сви концепти описани у овом поглављу коришћени су приликом формирања апстрактне синтаксе наменског језика *uniVEC*, истакнуте у поглављу 4.

3.1 Структура векторске базе података

Пре уноса података, крајњи корисници треба да припреме податке, у циљу складиштења у формату релевантном за векторске базе података. Као подршка при форматирању података, јавља се концепт *шеме*. Концептом шеме формирана је структурна корелација између података и објеката унутар базе података.

Концепт *колекције* служи као складиште за организацију и управљање ентитетима. Приликом креирања колекције, у одређеним векторским базама података, потребно је креирати шему колекције, која обезбеђује структуру колекције.

Концепт *поља* односи се на специфичан атрибут ентитета у колекцији. Поље може бити векторизовано или невекторизовано. Шеме се такође могу користити за опис структуре поља.

Ентитет је представљен скупом поља унутар колекције. Сваки ентитет колекције јединствено је идентификован примарним кључем.

Концепт *кластера* подразумева логичко груписање сличних података унутар векторске базе података. Овај механизам груписања организује тачке података на основу њихове близине или сличности у простору обележја, олакшавајући тако анализу података.

Након складиштења података, у циљу ефикасније обраде упита, врши се *индексирање* података. У векторским базама података, разликују се скаларни индекси, формирани за невекторизована поља, са циљем да убрзају упите са потпуним подударањем и векторски индекси, формирани на основу векторизованих поља, ради убрзања претраге сличности.

Код индекса векторског типа, потребно је навести која *метрика* ће бити коришћена за израчунавање сличности између тачака у векторском простору, које су складиштене у векторској бази података.

3.2 Векторизација и манипулација подацима

Основни концепт векторских база података је *векторска уградња* (енгл. *vector embedding*), помоћу ког су подаци различитог типа структуираности

приказани кроз *n*-димензионалне векторе. Векторска уградња, у општем смислу, подразумева укорењен концепт у машинском учењу, где представља податке мапиране у високодимензионални простор, при чему семантички слични подаци егзистирају као физички блиске тачке података у векторском простору. Уобичајен приступ формирања векторских уградњи ослоњен је на дубоке неуронске мреже, где се користе предефинисани модели машинског учења, попут модела *BERT* (енгл. *Bidirectional Encoder Representations from Transformers*).

Поред векторске претраге сличности, могуће је вршити и претрагу на основу вредности обележја датих у оквиру постављеног упита. Додатно, *VDBMS* нуди уобичајене операције уноса, измене и брисања података.

Како не постоји стандардизовани језик за интеракцију са векторским базама података, корисници им углавном приступају помоћу програмског кôда, што захтева стручност у интерпретацији циљних операција у оквиру одабраног програмског језика. Програмски језици који подржавају рад са векторским базама података су *Python, Java, JavaScript/TypeScript, Node.js* и *Go*, а приступ бази података могуће је остварити путем *REST API* принципа.

4. НАМЕНСКИ ЈЕЗИК *UNIVEC*

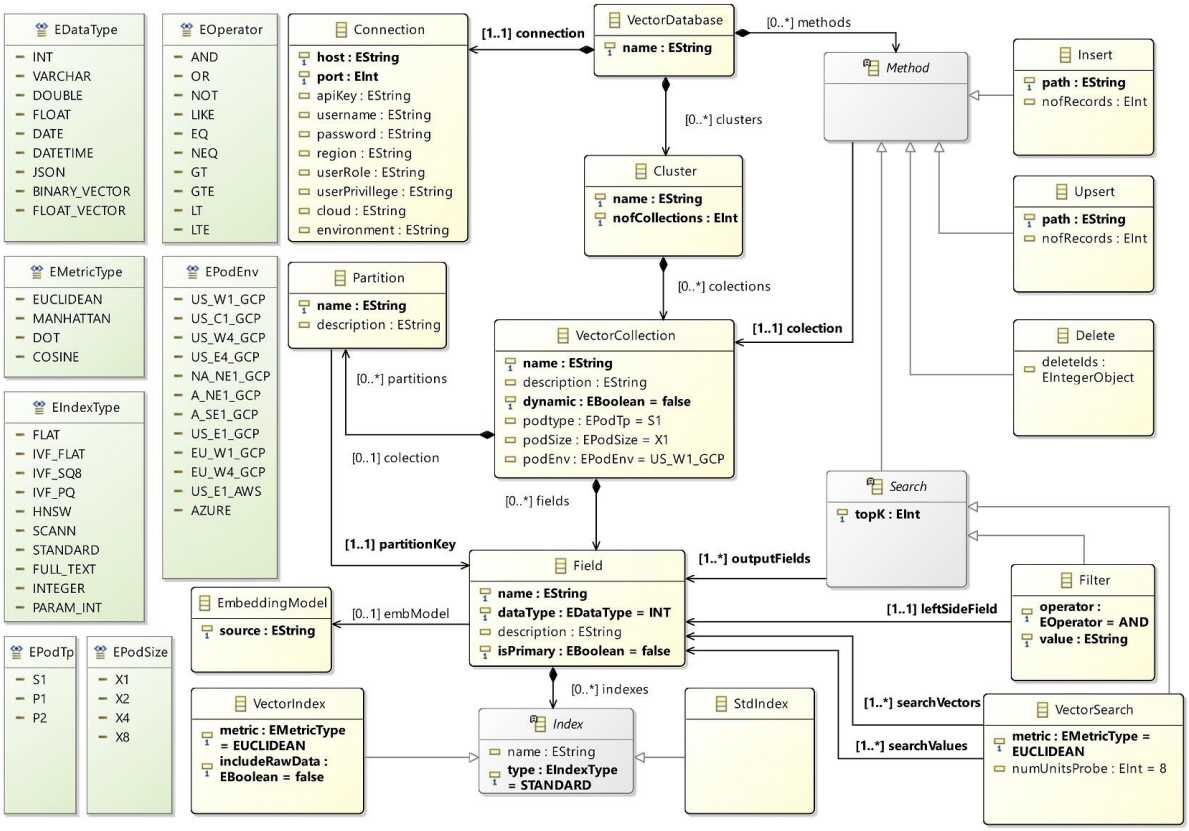
Наменски језик *uniVEC* развијен је помоћу језика *Ecore* у окружењу *Eclipse Modeling Framework* (*EMF*) [5]. За потребе синтаксне и семантичке провере модела насталих помоћу језика *uniVEC*, формиране су софтверске компоненте за проверу ограничења дефинисаних на нивоу мета-модела. Наменски језик подржава конкретну текстуалну синтаксу креирану у оквиру радног оквира *Xtext* [6]. Апстрактна синтакса језика *uniVEC* представљена је у поглављу 4.1, а у поглављу 4.2 приказан је пример модела насталог употребом дела конкретне синтаксе.

4.1 Апстрактна синтакса језика *uniVEC*

Коренски концепт апстрактне синтаксе приказане на *Слици 1* је векторска база података (*VectorDatabase*), која мора да има једну конекцију (*Connection*), како би веза ка циљној векторској бази података била успостављена. У векторској бази података може се наћи више кластера (*Cluster*), где сваки кластер може садржати више колекција (*VectorCollection*). Колекцију чини скуп поља (*Field*), која могу бити векторизована и невекторизована. За свако векторизовано поље неопходно је навести модел за генерисање векторских уградњи (*EmbeddingModel*). За поље је могуће дефинисати индекс (*Index*), векторизовани или невекторизовани, у складу са наведеним типом поља. За једну колекцију могуће је формирати онолико партиција (*Partition*) колико има поља, где је кључ партиције поље на основу ког је иста формирана. На нивоу векторске базе података, могуће је извршити следеће методе (*Method*): унос (*Insert*), измену (*Upsert*) и брисање (*Delete*) података, те филтрирање података (*Filter*) и векторску претрагу сличности (*VectorSearch*). Методи се извршавају на нивоу одређене колекције.

4.2 Конкретна синтакса језика *uniVEC*

За потребе коришћења језика *uniVEC*, формирана је текстуална конкретна синтакса језика, која је настала



Слика 1. Апстрактна синтакса наменског језика *uniVEC*

на основу мета-модела. У примеру представљеном у *Листингу 1*, приказано је формирање векторске базе података, са једним кластером, где је садржана колекција у вези са филмовима. Колекција има два поља, идентификатор филма – невекторизовано поље и примарни кључ, као и наслов филма – векторизовано поље. Од метода, примарно је илустрован унос података из датотеке типа *CSV* (енгл. *Comma-Separated Values*), те је извршена векторска претрага сличности, на основу поља наслова филма, где треба добавити пет најсличнијих филмова који у себи садрже реч „*Greek*“.

```
CREATE VDB domijvd
WITH CONNECTION
 host = 127.0.0.1
 port = 19530
WITH CLUSTERS
CREATE CLUSTER domijcluster
 number of collections = 5
WITH COLLECTIONS
 CREATE COLLECTION MOVIES
  dynamic
  description = descr01
WITH FIELDS [
 CREATE FIELD id
  PRIMARY
  type = INT,
 CREATE FIELD movietitle
  type = VARCHAR ]
WITH METHODS
 INSERT INTO COLLECTION "domijcluster.MOVIES"
  data file path = "C:\Users\userXY\Desktop\data.csv"
  number of records = 10000;
 Vector Search ON COLLECTION "domijcluster.MOVIES"
  topK = 5
  metric = EUCLIDEAN
  search fields = [movietitle]
  value = "Greek"
  output fields = ["domijcluster.MOVIES.id",
"domijcluster.MOVIES.movietitle"];
```

Листинг 1. Пример конкретне синтаксе језика *uniVEC*

5. ГЕНЕРАТОРИ КÔДА

Генератори кôда генеришу извршив програмски кôд из модела, насталих помоћу језика *uniVEC*, у одговарајуће *Python* скрипте за подршку рада са одабраном векторском базом података, као што су: *Pinecone*, *Milvus*, *Chroma*, *Weaviate* или *Qdrant*. Генератори кôда написани су у програмском језику *Xtend* [6]. У оквиру *Листинга 2*, илустрован је шаблон за генерисање индекса. Поред наведеног шаблона, имплементирани су и шаблони за генерисање концепата описаних у поглављу 3.

```
def generateIndex(Index index) '''
    «IF index instanceof StdIndex»
        indexType = «index.type.getName()»
        Index(name="«index.name»",
type=IndexType.valueOf(indexType))
    «ELSEIF index instanceof VectorIndex»
        indexType = «index.type.getName()»
        metricType = «index.metric.getName()»
        indexParams = newHashMap[String, String]
        «IF index.includeRawData»
            indexParams.put("includeRawData", "true")
        «ENDIF»
        «FOR limit: index.indexLimits»
            indexParams.put("«limit.type.getName()»",
"«limit.value»")
        «ENDFOR»
        VectorIndex(
            name="«index.name»",
            type=IndexType.valueOf(indexType),
            metric=EMetricType.valueOf(metricType),
            index_params=indexParams)
    «ENDIF» '''
```

Листинг 2. Део шаблона за генерисање концепта индекса

6. ЗАКЉУЧАК

Наменски језик *uniVEC* обједињује концепте векторских база података и омогућава униформни приступ различитим векторским базама података. Представљени језик и генератори кôда имају циљ да буде: поједностављен рад са циљном векторском базом података; редуковано време неопходно за овладавање концептима у различитим векторским базама података; олакшано тестирање различитих векторских база података; и поједностављена миграција између одабране изворне и одредишне векторске базе података.

Будући правци истраживања и развоја представљеног окружења обухватају подршку рада са додатним векторским базама података, што би подразумевало имплементацију нових генератора кôда и потенцијално проширење синтаксе наменског језика *uniVEC*. Додатно, могуће је интегрисати језик *uniVEC* у форми библиотеке у одређени програмски језик, како би било могуће користити *uniVEC* у окружењу програмског језика, што би инжењерима софтвера омогућило програмирање софтверског решења и приступ векторској бази података из једног окружења.

ЛИТЕРАТУРА

[1] Z. Altan, Ed., *Applications and Approaches to Object-Oriented Software Design: Emerging Research and Opportunities*. in Advances in Systems Analysis, Software Engineering, and High Performance Computing. IGI Global, 2020. doi: 10.4018/978-1-7998-2142-7.

[2] T. Taipalus, “Vector database management systems: Fundamental concepts, use-cases, and current challenges,” *Cogn. Syst. Res.*, vol. 85, p. 101216, Jun. 2024, doi: 10.1016/j.cogsys.2024.101216.

[3] M. Brambilla, J. Cabot, and M. Wimmer, *Model-Driven Software Engineering in Practice*, 2nd ed., San Rafael, USA: Morgan & Claypool Publishers, 2017.

[4] M. Mernik, J. Heering, and A. M. Sloane, “When and how to develop domain-specific languages,” *ACM Comput. Surv.*, vol. 37, no. 4, pp. 316–344, Dec. 2005, doi: 10.1145/1118890.1118892.

[5] D. Steinberg, F. Budinsky, M. Paternostro, E. Merks, *EMF: Eclipse Modeling Framework*, 2nd ed., Upper Saddle River, NJ: Addison-Wesley, 2009.

[6] L. Bettini and S. Efftinge, *Implementing domain-specific languages with Xtext and Xtend*, 2nd ed., Birmingham Mumbai: Packt Publishing, 2016.

КРАТКА БИОГРАФИЈА

Елена Акик рођена је 2000. године у Руми у Републици Србији. Школске 2019/2020. уписује се на Факултет техничких наука у Новом Саду, на студијски програм Информациони инжењеринг. Основне академске студије завршила је 2023. године и исте године се уписује на мастер академске студије, на студијском програму Рачунарство и аутоматика, модул – Инжењеринг информационих система.