Зборник радова Факултет техничких наука, Нови Сад

UDK: 004.9
DOI: https://doi.org/10.24867/18BE23Jovkic

АНАЛИЗА АЛАТА ЗА КРЕИРАЊЕ ГРАФИЧКЕ КОНКРЕТНЕ СИНТАКСЕ НАМЕНСКИХ ЈЕЗИКА

ANALYSIS OF TOOLS FOR CREATION OF DOMAIN SPECIFIC LANGUAGES GRAPHIC CONCRETE SYNTAX

Јован Јовкић, *Факултет техничких наука, Нови Сад*

Област – ПРИМЕЊЕНЕ РАЧУНАРСКЕ НАУКЕ И ИНФОРМАТИКА

Кратак садржај – *У овом раду описан је графички наменски језик за моделовање рецепата за припрему јела. Описано је окружење које је коришћено за израду апстракне синтаксе, а након тога приказана је апстрактна синтакса овог језика, као и детаљан опис сваког од концепата. Графичка конкретна синтакса је моделована уз помоћ три различита алата. Сваки алат је детаљно описан, односно описане су све технологије које су коришћене у изради мастер рада. Након тога приказана је примена сваког појединачног алата на изради конкретне синтаксе језика за приказивање рецепата. Овај део представља и круцијалну тачку самог рада. На крају су укратно сумирана искуства употребе сва три алата. За развој метамодела коришћено је окружење Eclipse Modeling Framework. За развој конкретне синтаксе коришћени су следећи алати: Sirius, Eugenia и VisualStudio.*

Кључне речи: *моделовање рецепата за припрему јела, наменски језици, развој софтвера заснован на моделима*

Abstract – *The paper describes a graphical domain-specific language for modeling recipes for food preparation. The environment used to create the abstract syntax is described, followed by the abstract syntax of this language, as well as detailed description of each of the concepts. Graphical concrete syntax is modeled using three different tools. Each tool is described by in detail, all the technologies used in the preparation of the master thesis are described. After that, the application of each individual tool on the creation of a specific syntax of the language for displaying recipes is shown. This part is also a crucial point of the work itself. Finally, the experiences of using all three tools are briefly summarized. Eclipse Modeling Framework was used for the creation of the meta-model. Sirius, Eugenia and VisualStudio were used to develop the concrete language syntax*

Keywords: *Recipes for food preparation, Domain-Specific Languages, Model-Driven Software Development*

1. УВОД

Софтвер постаје све присутнији у људским животима, самим тим се захтеви које софтвер треба да испуни значајно увећавају. Развој софтвера представља сложен процес који постаје све ефикаснији сваки пут када се ниво апстракције повећа.

Такав корак у развоју софтвера представља и појава наменских језика који су засновани на моделима. Развој софтвера није самостална активност, често је потребна комуникација са особама који нису из света рачунарства, а самим тим је потребно одређено посредовање у опису техничких аспеката развоја.

Наменски језици омогућују како програмерима, тако и доменским експертима, који немају искуства са програмским језицима, да креирају моделе које ће се касније одређеним трансформацијама претворити у готове апликације, извештаје или нешто треће. Посебно су корисни графички наменски језици, јер не захтевају писање кода, већ је довољно креирати графички модел у едитору, који може бити генерисан. Модел се креира уз помоћ графичких представа концепата, које би требало да буду интуитивне и да доменске експерте подсете на концепте реалног система, које они веома добро познају.

Рецепт представља кратак текст који се састоји од корака које је потребно испратити не би ли се направило жељено јело. Може се састојати од дела где су побројани сви потребни састојци и од корака за припрему.

Доменски језик омогућава професионалним куварима да на једноставан начин представе начин припреме одређеног јела тако што ће користити одређене графичке симболе и на тај начин извршити опис припреме. Осим кувара, језик могу користити и аматери, који могу своје идеје да представе уз помоћ графичких представа и на тај начин уоче однос између одређених састојака, који су неопходни за припрему јела.

Циљеви употребе оваквог доменског језика јесу повећање ефикасности, односно на једноставнији начин записивања рецепата, на тај начин ће визуелно бити приказани сви састојци и везе између њих. Поред овога, доменски језик ће омогућити више врста валидације, као што је провера да ли кухињски прибор може да се користи за састојке одређеног типа.

НАПОМЕНА:
Овај рад проистекао је из мастер рада чији ментор је био доц. др Милан Челиковић.

2. ПРЕГЛЕД ПОСТОЈЕЋЕГ СТАЊА У ОБЛАСТИ

Модел је апстракција система или неког његовог дела. Модел може пружити једноставан поглед на систем, а такође може бити и много детаљнији. Уобичајено је коришћење *Unified Modeling Language (UML)* модела, то су модели који се данас најчешће користе.

Језик за спецификацију модела система се назива метамодел, односно модел представља инстанцу метамодела у одређеном домену. Метамодел у основи представља дефиницију језика за моделирање, јер омогућава да се преко њега представи велики број модела.

Domain Specific Language (DSL), наменски језик, јесте програмски језик са вишим нивоом апстракције оптимизован за одређену класу проблема. Овакви језици се развијају у континуираној сарадњи са стручњацима из дате области за коју се развија језик. У многим случајевима овакве језике не користе програмери, већ особе које добро познају домен за који је језик креиран. Поред тога, предност оваквих језика јесте што су они блиски како програмерима, који се баве креирањем таквим језика, тако и корисницима, односно доменским стручњацима.

Језик за моделовање садржи три обавезна дела:

- Апстрактна синтакса – описује структуру језика и начин на који се различити примарни елементи могу међусобно комбиновати.
- Конкретна синтакса – представља реализацију апстрактне синтаксе, односно представља графичке симболе или текстуалне ознаке који означавају концепте апстракте синтаксе. Текстуална и графичка конкретна синтакса су два најпознатија типа конкретне синтаксе.
- Семантика – описује значење елемената који су дефинисани у језику [1].

3. АПСТРАКТНА СИНТАКСА

Апстрактна синтакса представља први корак који је неопходан у оквиру развоја модела, односно у процесу моделовања. Састоји се од дефинисања концепата и њихових атрибута тако што се дефинише метамодел, који представља основу за развој текстуалне или графичке конкретне синтаксе.

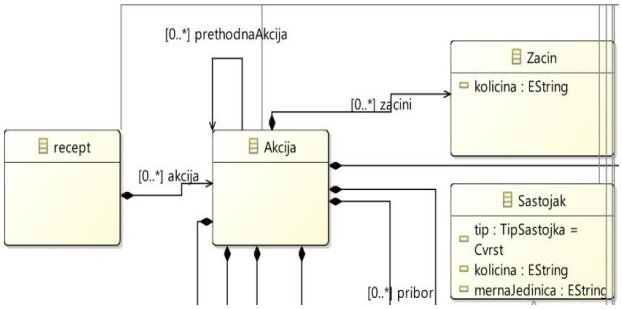
Метамодел се састоји од класа, атрибута, асоцијација. Концепти се представљају уз помоћ класа, сам назив класе представља назив концепта. Атрибути класе представљају особине концепта, док асоцијације представљају везе између концепата.

Eclipse Modeling Framework (EMF) јесте оквир за моделирање и генерисање кода за израду алата и других апликација које су засноване на моделима.

EMF [2] је састоји од три основна дела:

- EMF – основни *EMF* оквир, укључује метамодел *Ecore* за описивање модела и подршку која укључује обавештавање о променама, подршку са *XMI* серијализацијом, као и АПИ за манипулисање *EMF* објектима
- EMF.Edit – оквир који укључује генеричке класе за израду едитора за моделе. Обезбеђује класе које омогућују да се *EMF* модели прикажу коришћењем стандардних прегледача. Укључује и скуп класа за изградњу едитора која подржавају поништавање и враћање акција (*undo* и *redo*)
- EMF.Codegen – способан је да генерише све оно што је неопходно да би се развио комплетни едитор за *EMF* модел. То укључује и графички кориснички интерфејс из којег се могу навести опције генерисања и могу се позвати генератори.

На слици 1. приказан је графички приказ дела апстрактне синтаксе која је развијена уз помоћ оквира *EMF* за потребе графичког језика за представљање рецепата. Потпуни приказ метамодела није био могућ због ограничења простора.



Слика 1: Део апстрактне синтаксе језика за приказивање рецепата

У наставку су побројани концепти који се користе у оквиру наменског језика за приказивање рецепата:

- Именовани елемент – у себи обједињује заједничке особине именованих елемената.
- Рецепт – коренски концепт
- Акција – концепт који представља појединачан поступак код којег се одређени састојци трансформишу у други облик
- Састојак – концепт који се односи на одређену намирницу која је потребна за спремање датог јела. Може бити улазни и излазни састојак.
- Прибор – концепт који се односи на алатке које су неопходне, да би се успешно завршила акција
- Посуда – концепт који се односи на посуде у којима се припрема јело
- Уређај – концепт који се односи на електричне уређаје у којима се припрема јело
- Савет – порука која ће особи која касније буде користила дати рецепт помоћи у изради јела
- Тајмер – концепт који се односи на време које је потребно да се одређена акција изврши.

4. ТЕХНОЛОГИЈЕ ЗА КОНКРЕТНУ СИНТАКСУ

Након дефинисања концепата и креирања апстрактне синтаксе потребно је дефинисати конкретну синтаксу која ће крајњем кориснику омогућити креирање модела наменског језика.

Графичка синтакса може бити текстуална и графичка. Текстуална конкретна синтакса је кориснија у случају када ће језик користити програмери, јер су то људи којима је близак текстуални код и моћи ће врло лако

да се навикну на нову синтаксу. Док је графичка синтакса много ближа људима који немају искуства са програмским језицима. Графичка конкретна синтакса је лакша за учење, ово пре свега зависи од искуства особе која ће користити наменски језик. Међутим, у случају визуалних језика, корисник може одмах да крене да их користи тако што ће постављати елементе на канвас и повезивати их, док се у случају текстуалних језика добије празан фајл, тако да је много теже урадити било шта без учења елемената језика [6]. Пошто овакве језике често користе домен-ски експерти, онда је и графичка синтакса чест избор у таквим ситуацијама.

Конкретна графичка синтакса мора имати следеће елементе:

- Графички симболи – линије, слике, облике и лабеле за текстуалне податке
- Правила како се симболи постављају на дијаграм и како се комбинују
- Мапирање графичких симбола са елементима апстрактне синтаксе, на тај начин се одређује који концепт је визуализован тим симболом

4.1. Sirius

Sirius [3] је *Eclipse* пројекат који омогућава веома једноставно креирање едитора за графичко моделирање користећи се *Eclipse* технологијама. Sirius, који је изграђен на врху *GMF-a*, користи се за креирање, визуализацију и уређивање модела помоћу интерактивних едитора. Sirius подржава дефиницију софистицираних графичких едитора, који пружају многе функције, као што су филтери, слојеви, сложени стилови итд. Дијаграми подржавају неколико врста графичких конструкција: једноставне чворове, контејнере (који могу садржати и друге чворове и контејнере), листе, чворове који су закачени за ивице других чворова. Свака врста елемената подржава богат скуп могућих стилова, могуће је одабрати разне облике и боје. Структура графичких елемената приказаних на дијаграму не мора одговарати физичкој структури, односно апстрактној синтакси коју представља.

4.2. Eugenia

Eugenia [4] је алат који поједностављује развој *GMF* модела тако што аутоматски генерише *.gmfgraph*, *.gmftool* и *.gmfmap* модел који су неопходни *GMF* едитору. Ови модели се генеришу на основу једног анотираног *Ecore* модела. *Eugenia* подржава следеће анотације: *gmf*, *gmf.diagram*, *gmf.node*, *gmf.link*, *gmf.compartment*, *gmf.affixed* и *gmf.label*.

4.3. Visual Studio

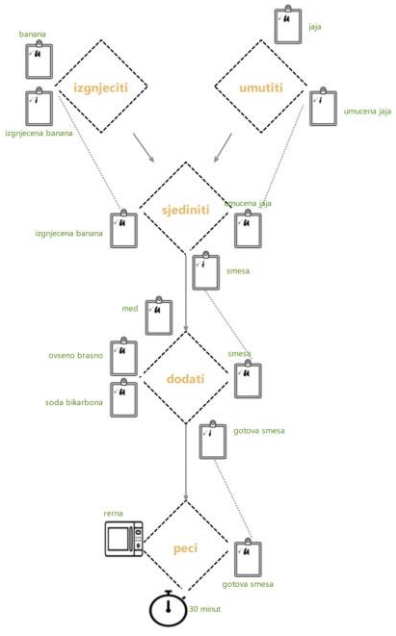
За наменског језика може се користити решење *Visual Studio* [5] које се развија из темплејта, односно шаблона. Кључни део решења јесте дијаграм за дефинисање домен специфичног језика, који се налази у *DslDefinition.dsl*. На овом дијаграму је могуће дефинисати класе и облике језика. Односно, апстрак-тна и конкретна синтакса се развијају на истом дијаграму, класе представљају апстрактну синтаксу, а облици представљају графичку синтаксу, односно како ће изгледати графички репрезенти појединих класа. Такође, могуће је додати програмски код, на тај начин је могуће додати много више детаља у развијани језик.

5. КОНКРЕТНА СИНТАКСА ЈЕЗИКА ЗА ПРИКАЗИВАЊЕ РЕЦЕПАТА

Сваки од алата је искоришћен за израду графичке конкретне синтаксе за потребе језика за приказивање рецепата.

5.1. Sirius

У оквиру *Viewpoint-a*, налази се један слој, који је подразумеван, сви елементи језика налазе се у истом слоју. Елементи које се налазе директно на дијаграму јесу графичке представе акција, порука и свих веза. Остали елементи се налазе у оквиру других елемената. Велики број елемената се нелази у оквиру чвора акција, сви ти елементи представљају ивичне чворове, јер ће се на дијаграму налазити залепљени на ивице матичног чвора, односно елемента који представља акцију. Нови чвор се креира тако што се на одабраном слоју одабере опција додати нови чвор. Након тога је потребно унети обавезне особине, без којих чвор не може бити исцртан, осим обавезних који су означени са звездицом, постоји и велики број опционих особина, које дају неке додатне инфор-мације о начину на који ће чвор бити представљен на дијаграму, али без њих сваки чвор може бити исцртан. Веза се креира одабиром Related Based Edge опције у оквиру подразумеваног чвора. Након тога је потребно унети неопходне особине које су потребне не би ли се исцртала једна веза. На слици 2. приказан је пример дијаграма који је креиран уз помоћ едитора који је генерисан уз помоћ Sirius алата.



Слика 2: Пример дијаграма који је креиран уз помоћ едитора који је генерисан у Sirius алату

5.2. Eugenia

На апстрактну синтаксу која је добијена на основу *Ecore* модела, додате су анотације, на основу којих је одређена конкретна синтакса и могуће је креирати моделе овог језика. У наставку приказан је део кода, приказана је анотирана класа Акција.

```
@gmf.node(figure="polygon",    polygon.x="0    75    150    75",
polygon.y="75   0   75   150",border.width="5",   label="naziv",
label.placement="internal")
class Akcija extends ElementNaziv
{
          @gmf.affixed(foo="bar")
          val Zacin[*] zacini;
          @gmf.affixed(foo="bar")
          val  UlazniSastojak[*] ulaznisastojak;
          @gmf.affixed(foo="bar")
          val  IzlazniSastojak[*] izlaznisastojak;
          @gmf.affixed(foo="bar")
          val Pribor[*] pribor;
          @gmf.affixed(foo="bar")
          val Posuda[*] posude;
          @gmf.affixed(foo="bar")
          val Uredjaj[*] uredjaji;
          @gmf.affixed(foo="bar")
          val Tajmer[?] tajmer;
          val PrethodnaAkcija[*] prethodnaAkcija;
          val SavetPovezivanje[*] savetPovezivanje;
}
```

Листинг 1:Део кода, анотирана класа Акција

5.3. Visual Studio

Развој језика специфичног за домен у окружењу *Visual Studio* захтева у исто време изградњу како апстрактне, тако и конкретне синтаксе. Апстрактна и конкретна синтакса се развијају на истом дијаграму. Након креирања пројекта добију се почетне класе које је потребно обрисати или преименовати.

5.4. Резиме сва три алата

Sirius је *Eclipse* пројекат који на основу претходно креиране апстрактне синтаксе омогућава дефинисање конкретне графичке синтаксе. Алат не захтева писање кода, већина основних функционалности се може креирати попуњавањем различитих особина у појединим картицама. Добре стране алата јесу што је велики спектар функционалности покривен у оквиру алатки које су у понуди самог алата. Мана овог алата јесте што је потребно на више места радити са истим концептом, на једном месту се креира графичка представа самог елемента, односно одређује се његова слика или геометријска презентација, на другом месту је потребно одредити како ће се концепт преставити у кутији са алаткама и које активности повлачи смештање самог елемента на канвас.

Eugenia омогућава да се *Ecore* метамодел који је дефинисан у *EMF*-у анотира уз помоћ одређених напомена и да се на основу тих напомена генерише едитор у којем је омогућено креирање дијаграма уз помоћ новокреиране графичке синтаксе. Овај приступ веома брзо доводи до решења и веома је добар за почетнике, јер не захтева превише напора за креирање почетне графичне синтаксе. Међутим, проблем је што уз помоћ анотација није могуће дефинисати све оно што је омогућено у претходном алату.

Visual Studio је другачији алат, самим тим што представља и другачије окружење. На истом дијаграму се развијају и апстрактна и конкретна синтакса. Рад на изради језика у овом алату је био мало комплексији у односу на претходна два. Ивични елементи не могу бити приказани у виду слике, без проширивања додатним функционалностима писањем кода, што је представљало проблем.

6. ЗАКЉУЧАК

Сва три алата која су коришћена у овом раду су показали своје предности, али и мане. *Eugenia* је алат који је веома једноставно користити, јер се састоји од писања анотација на готову апстрактну синтаксу. Због тога је овај алат веома користан за почетнике, јер веома брзо доводи до решења. *Visual Studio* је алат који захтева у исто време креирање апстрактне и конкретне синтаксе, што може бити веома корисно. Свакако, алат који се показао као најбољи јесте *Sirius*, који пружа много могућности, већина операција се изводи кроз дијалоге, али у случају проширења је свакако могуће писање јава кода.

Као проширење рада могуће је испробати још неке од алата за дефинисање конкретне синтаксе, који ће се можда показати као једноставнији и бољи за рад. Примери таквих алата су: *MetaEdit+*, *Graphiti*, *Atom3*, *Poseidon*. За сваки од ових алата је потребно одредити које су им предности и мане, неке од особина које је потребно испитати јесу: комплексност рада у алату, време које је потребно да се добија како иницијална, тако и финална верзија једног наменског језика, поред тога цена алата такође може одлучити да ли ће се тај алат користити даље у раду.

7. ЛИТЕРАТУРА

[1] Brambilla, M., Cabot, J., & Wimmer, M. (2017). Model-driven software engineering in practice

[2] Eclipse Modeling Framework(EMF). Доступно на: https://www.eclipse.org/modeling/emf/. Посећено: 11.02.2022.

[3] Sirius. Доступно на: https://www.eclipse.org/sirius/. Посећено: 11.02.2022.

[4] Eugenia. Доступно на: https://www.eclipse.org/epsilon/doc/eugenia/. Посећено: 11.02.2022.

[5] Visual Studio. Доступно на: https://visualstudio.microsoft.com/. Посећено: 11.02.2022.

[6] Milosavljević, G., Dejanović, I., Tumbas Živanov, M., Perišić B. (2010). Comparation of Textual and Visual Notation of DOMMLite Domain-Specific Language

Кратка биографија:

Јован Јовкић рођен је 1997. године у Новом Саду. Завршио је гимназију у Оџацима. Факултет техничких наука, смер Примењено софтверско инжењерство, уписао је 2016. године. Дипломирао је 2020. год. и исте године уписао мастер студије на Факултету техничких наука, смер Рачунарство и аутоматика.