|  |  |  |
| --- | --- | --- |
|  | **Софийски университет „Св. Кл. Охридски”**Факултет по математика и информатика*Магистърска програма„Софтуерни технологии”* |  |

**Предмет: Обектно-ориентиран анализ и проектиране
на софтуерни системи**

***Зимен семестър, 2020/2021 год.***

**Тема 6: Сравнение между UML и Domain-Specific Language (DSL)**

**Есе**

*Автор:*

*Лъчезара Георгиева Георгиева*

*фак. номер* 26338

декември, 2020

София

Съдържание

[1 Въведение 3](#_Toc58867677)

[2 Характеристики и използване на UML (Unified Modeling Language). 3](#_Toc58867678)

[2.1 Дефиниции 3](#_Toc58867679)

[2.2 Основни характеристики 4](#_Toc58867680)

[2.2.1 Видове инструменти за работа с UML диаграми. 5](#_Toc58867681)

[2.2.2 Генериране на код от UML диаграма 5](#_Toc58867682)

[2.3 Въведение в използването на UML чрез диаграми. 6](#_Toc58867683)

[2.4 Недостатъци при използване на UML 7](#_Toc58867684)

[3 Характеристики и използване на DSL (Domain Specific Language). 7](#_Toc58867685)

[3.1 Какво е DSL? 7](#_Toc58867686)

[3.2 Основни характеристики 7](#_Toc58867687)

[3.2.1 Пример за DSL езици: 8](#_Toc58867688)

[3.2.2 Примерна DSL диаграма: 8](#_Toc58867689)

[3.3 Въведение в използването на DSL 9](#_Toc58867690)

[3.4 Ограничения при използването на DSL 10](#_Toc58867691)

[4 Сравнителен анализ 10](#_Toc58867692)

[5 Сравнение на DSL и UML 10](#_Toc58867693)

[6 Добри практики при работа с DSL 11](#_Toc58867694)

[7 Заключение и очаквано бъдещо развитие 11](#_Toc58867695)

[8 Използвани литературни източници 11](#_Toc58867696)

# Въведение

В съответната тема ще бъдат разгледани разликите между UML (Unified Modeling Language) и DSL (Domain Specific Language). Ще се опишат различни варианти за използване на двата вида езика, за да може да се разграничи начина им на употреба и ще се опишат най-добрите практики.

# Характеристики и използване на UML (Unified Modeling Language).

UML, съкратено от Unified Modeling Language, е стандартизиран език за моделиране, състоящ се от интегриран набор от диаграми, създаден в помощ на разработчиците на системи и софтуер за определяне, визуализиране, конструиране и документиране на артефактите на софтуерни системи, както и за бизнес моделиране и други не-софтуерни системи. UML представлява колекция от най-добри инженерни практики, които са се оказали успешни при моделирането на големи и сложни системи. UML използва предимно графични обозначения, за да изрази дизайна на софтуерни проекти. Използването на UML помага на екипите на проекти да комуникират, да изследват потенциални проекти и да валидират архитектурния дизайн на софтуера.

## Дефиниции

Както е обяснено по-горе, UML се състои от интегриран набор от диаграми. Тези диаграми се делят на два вида:

* **Структурни диаграми –**  описва се статичната структура на системата или на части от нея.

Структурните диаграми включват:

Диаграма на компоненти, **Диаграма на обекти**, Диаграма на класове и **​Диаграма на разгръщане, Диаграма на пакета,Композитна диаграма**.

* **Поведенчески диаграми –** описва се динамичната структура на системата или на нейното поведение.

Поведенческите диаграми включват:

Диаграма на случаи на употреба, Диаграма на състоянията, Диаграма на дейност, Диаграма на взаимодействие.



Фигура 1. Структура на UML диаграма[[1]](https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/) .

## Основни характеристики

* UML означава унифициран език за моделиране (Unified Modeling Language).
* UML е визуализиращ език, използван за създаване на софтуерни чертежи/схеми.
* UML може да се опише като език за визуализиране на модели с общо предназначение за визуализиране, спесифициране, изграждане и документиране на софтуерна система.
* Въпреки че UML обикновено се използва за моделиране на софтуерни системи, то не значи,че не може да бъде използван и за конструиране на модели в различни сфери. Пример за това е създаването на диаграма на случаите на употреба в производствения процес в даден завод.
* UML има утвърден „стандарт“.
* Високо ниво на четимост и преизползваемост на системата.
* UML не е език за програмиране, но могат да се използват инструменти за генериране на код на различни езици, използвайки UML диаграми.



Фигура 2. Примерна клас диаграма и нейните връзки [[2]](https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/) .

### Видове инструменти за работа с UML диаграми.

За работа с UML диаграми са нужни инструменти (софтуер) за създаването на модели. Описани са няколко вида софтуер:

* Visual Paradigm
* Lucidchart
* Gliffy
* SmartDraw
* Creately

### Генериране на код от UML диаграма

Visual Paradigm поддържа генериране на изходен код на Java от UML модел на клас и поддържа синхронизиране на изходния код и UML модел чрез добавената функционалност “Java Round-Trip“. Такава функционалност има и за C++.

Може да се използва „Instant Code Generation / Reversal” (Незабавно генериране / обръщане на код).

Java, C #, C ++, Python, PHP, Hibernate, Ruby, VB.NET, .NET dll или exe, ODL, ActionScript, IDL, Delphi, Perl, XML, XML схема, Objective-C 2.0, Ada95



Фигура 3. Round-Trip Java Engineering [[3]](https://www.visual-paradigm.com/features/code-engineering-tools/) .

## Въведение в използването на UML чрез диаграми.



 Фигура 4. Диаграма на дейността [[4]](https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/) .

UML модел на софтуер може да се създаде чрез даден инструмент като Visual Paradigm.

В случая, дадената диаграма на дейността визуализира система за следене на процеса на поръчка. Когато поръчката е получена, дейностите се разделят на две. Едната обработва и изпраща поръчката, а другата създава фактурата. В този случаи, ясно се следи какво се случва с поръчката и какви проблеми могат да изникнат.

## Недостатъци при използване на UML

* **Формалната нотация и комплексната структура на диаграмите**
Изграждането на софтуер чрез UML може да бъде много сложен и пренаситен за разработчиците. Тъй като, те работят с код, а не с диаграми, то UML моделите могат повече да ги объркат, отколкото да им помогнат.
* **Диаграмите могат да бъдат пренаситени**

Тъй като UML съдържа много диаграми, които се използват за различни визуализации, с различни атрибути и връзки, разработчиците биха се объркали значително заради цялостната информация, която показват диаграмите.

# Характеристики и използване на DSL (Domain Specific Language).

## Какво е DSL?

Специфичен за домейн език (DSL) е изпълняващ спецификации език, който чрез подходящи нотации и абстракции да се справи с валидирането на конкретен порблем в дадена област. Генерират се модели чрез DSL, но не са стандартизирани като UML. DSL е предназначен за използване в контекста на определен домейн.

## Основни характеристики

* Използване на по-изразителен език вместо генерално-насочените езици.
* Споделяне на общо разбиране между всички потребители на системата.
* Използва се в конкретна област при създаване на приложението.
* Избягва се boilerplating (преизползван код) на техническия код и ще се надгражда.
* Могат да се създават уникални нотации, абстракции и означение, за да може системата да бъде разбираема за всички и да се създаде легенда.
* Може да се каже,че DSL обединява бизнес и технически екипи.

###  Пример за DSL езици:

* HTML
* CSS
* SQL
* XML
* DOT

### Примерна DSL диаграма:

Дадената DSL диаграма е чрез език за дефиниране на графи (DOT):





Фигура 5. Диаграма на граф .

## Въведение в използването на DSL



Фигура 6. Диаграма на дейността  [[6]](https://www.researchgate.net/publication/2389477_Domain-Specific_Language_Design_Requires_Feature_Descriptions) .

Идеята на DSL е да се създадат случаи на употреба,които да бъдат ясни за всеки потребител.
В тази система имаме кола, на която се описани всички особености. От съответната схема диаграмата може да се генерира като Java file по следния начин:



В Visual Studio има Modeling SDK, който изгражда графични модели за дадената цел.



Фигура 7. Диаграма на дейността  [[7]](https://docs.microsoft.com/en-us/visualstudio/modeling/getting-started-with-domain-specific-languages?view=vs-2019) .

## Ограничения при използването на DSL

Недостатъкът е, че DSL се използва тясно в рамките на предвидения домейн и цел. Освен това, въпреки че може да има предимства при използването на инструменти за DSL улавяне, те не са от съществено значение и разработването или конфигурирането на такива инструменти е допълнително усилие. Създателите на DSL се нуждаят от знания за домейни, както и знания за развитие на езика, а хората рядко имат и двете.

# Сравнителен анализ

* UML има огромна област, върху която трябва да се работи и разучи.
* UML да се използва и върху не софтуерни продукти, докато DSL е строго дефиниран като е в рамките на дадения проект, но ако е добре описан може да се преизползва.
* UML не се надгражда и вече е с изграден стандарт, докато DSL е лимитран и е създаден от инженери в съответната област и еволюцията му е бърза.

# Сравнение на DSL и UML

|  |  |  |
| --- | --- | --- |
| **Критерии** | **UML** | **DSL** |
| Област | Голяма и комплексна | Малка и добре дефинирана |
| Размер на езика | Голям | Малък |
| Нива на абстракция | Задоволително | Лимитирано |
| Разработен от | Специален екип | Инженери в областта |
| Живот | Продължителен | До няколко месеца |
| Еволюция | Бавна | Бърза |

# Добри практики при работа с DSL

* DSL не са GPL. Добра практика е да се ограничава диапазона от проблеми в окончателния домейн.
* Не е необходимо да се дефинира собствен DSL. Добра практика е да се намери съществуващ DSL, който решава нуждите на потребителя.Пример за това са - сайтове като DSLFIN, който изброява DSL за финансовия домейн.
* По-добре е DSL да се правят конкретно, а не прекалено технически.

# Заключение и очаквано бъдещо развитие

DSL e предпочитан, тъй като може да се дефинира езика, който ще се използва и да се използва за конкретна инстанция. DSL могат да се пишат с текстови процесори. Използването на DSL редактор обаче улеснява проверките на синтаксиса и семантиката.

# Използвани литературни източници

1. [What is Unified Modeling Language (UML)?](https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/)
2. [What is Unified Modeling Language (UML)?](https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/)
3. [UML/ Code Generation Tool](https://www.visual-paradigm.com/features/code-engineering-tools/)
4. [Activity DIagram](https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/)
5. [Domain-Specific Language Design Requires Feature Descriptions](https://www.researchgate.net/publication/2389477_Domain-Specific_Language_Design_Requires_Feature_Descriptions)
6. [What can you do with Domain-Specific Language?](https://docs.microsoft.com/en-us/visualstudio/modeling/getting-started-with-domain-specific-languages?view=vs-2019)