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, 2022/2023 год.

Тема 34: Сравнение между UMLet и PlantUML

Есе

Автор:

Снежина Цанкова, фак. номер 0MI3400204

ноември, 2022

София

Съдържание

1. Въведение 3
2. Характеристики и използване на UMLet и PlantUML 4
   2.1. Дефиниции 4
   2.2. Основни характеристики 4
      2.2.1. Графичен интерфейс на UMLet 4
      2.2.2. Създаване на персонализиран елемент в UMLet 6
      2.2.3 Създаване на клас в UMLet 6
      2.2.3. Уеб сайтът UMLetino и UMLet plugin 8
      2.2.4. Диаграми на последователностите в PlantUML 8
      2.2.5. Диаграми на класовете в PlantUML 9
   2.4. Ограничения при използването на UMLet и PlantUML 11
3. Сравнителен анализ 11
   3.1. Критерии за сравнение 11
   3.2. Сравнение на UMLet и PlantUML 12
   3.3. Сравнение на UMLet и Visual Paradigm 13
4. Примери на използване 15
   4.1. Use Case диаграма изчертана на UMLet 15
   4.2. Object state диаграма изчертана на UMLet 15
   4.3. Клас диаграма в PlantUML 16
   4.4. Диаграма на дейностите в PlantUML 17
5. Добри практики и методи за използване 18
6. Заключение и очаквано бъдещо развитие 19
7. Използвани литературни източници 19

1. Въведение

Унифицираният език за моделиране (UML) е език за моделиране с общо предназначение в областта на софтуерното инженерство, който е предназначен да осигури стандартен начин за визуализиране на дизайна на система. UML е набор от визуални стандарти, които помагат на разработчиците на софтуер и инженерите да „говорят на един и същ език“, без да се ровят в действителния код на техния продукт. Изготвянето на диаграми с UML е чудесен начин да помогнете на другите бързо да разберат сложна идея или структура. Част от най-популярните UML инструмент са именно Adobe Spark, Edraw Max, Moqups, Visio, Lucidchart, Visual Paradigm, UMLet, PlantUML и други. [3]

UMLet е безплатен инструмент за чертаене на UML диаграми. Съществуват три вариации на UMLet - stand-alone Java приложение, добавка (plugin) към различни Integrated Development Environment (IDE) редактори, приложение постъпвано чрез уеб браузър. Основната цел на UMLet е да даде възможност за бързо създаване на UML диаграми. За да бъде възможно това, инструментът притежава опростен потребителски интерфейс. Редактирането на UML елементите в инструмента се осъществява чрез текстово поле. Наличието на такова поле премахва необходимостта от много контекстни менюта, бутони и икони, но и ограничава броя на потребителите на UMLet, тъй като е необходимо допълнително познание относно специфичния език, които UMLet използва. От друга страна, потребителският интерфейс на инструмента Visual Paradigm се основава на “drag and drop”, което улеснява потребителите при използване. Друга съществена разлика между UMLet и инструменти като Visual Paradigm, LucidChart, MagicDraw, SmartDraw и др. е че те са платени - безплатна е само част от тяхната функционалност или имат безплатен тестови период, докато UMLet е напълно безплатен за използване и точно това е причина да той е предпочитан в някои случаи пред другите инструменти.

Голям плюс на UMLet, е че чрез него можем да създаваме персонализирани UML елементи чрез Java код или да модифицираме наличните вече елементи и да ги запазваме. Тази функционалност позволява на потребителите на UMLet да персонализират работа си с инструмента, така че той да е съобразен с техните нужди.

UMLet инструментът позволява експортирането на диаграми в различни файлови формати - eps, pdf, jpg, svg или тяхното директно копиране. [1]

Подобно на UMLet, PlantUML е напълно безплатен инструмент за чертаене на UML диаграми. PlantUML е инструмент с отворен код, който позволява на потребителите да генерират диаграми, използвайки само текст. PlantUML поддържа разнообразие от различни формати, както и визуализация на JSON и YAML файлове. трябва да бъдем внимателни, защото инструментът ни позволява да чертаем и непоследователни диаграми (като например два класа, наследяващи се един от друг). Така че това е по-скоро инструмент за рисуване, отколкото инструмент за моделиране. PlantUML ограничава ширината и височината на изображението до 4096. Съответно конфигурацията може да бъде променена, чрез промяна на стойността на променлива. Инструментът съществува под формата на Java приложение, добавка за Integrated Development Environment (IDE), добавка за Confluence (продукт на Atlassian) и приложение, което може да бъде достъпвано чрез уеб браузър. Диаграмите изготвени с PlantUML могат да бъдат експортирани във форматите png, svg, ascii art, LaTeX. [2]

Това есе представлява сравнение между двата инстумент - UMLet и PlantUML и съответното им сравнение с инструментът Visual Paradigm, както и напътствия за използването на инструментите и описание на основните им функционалности.

2. Характеристики и използване на UMLet и PlantUML

2.1. Дефиниции

Разглежданата версия на UMLet е 15.0. Тази версия на UMLet може да бъде достъпвана чрез уеб сайт (UMLetino) с известно ограничение на функционалностите, Java приложение или добавка за IDE - Eclipse или Visual Studio Code. [1]

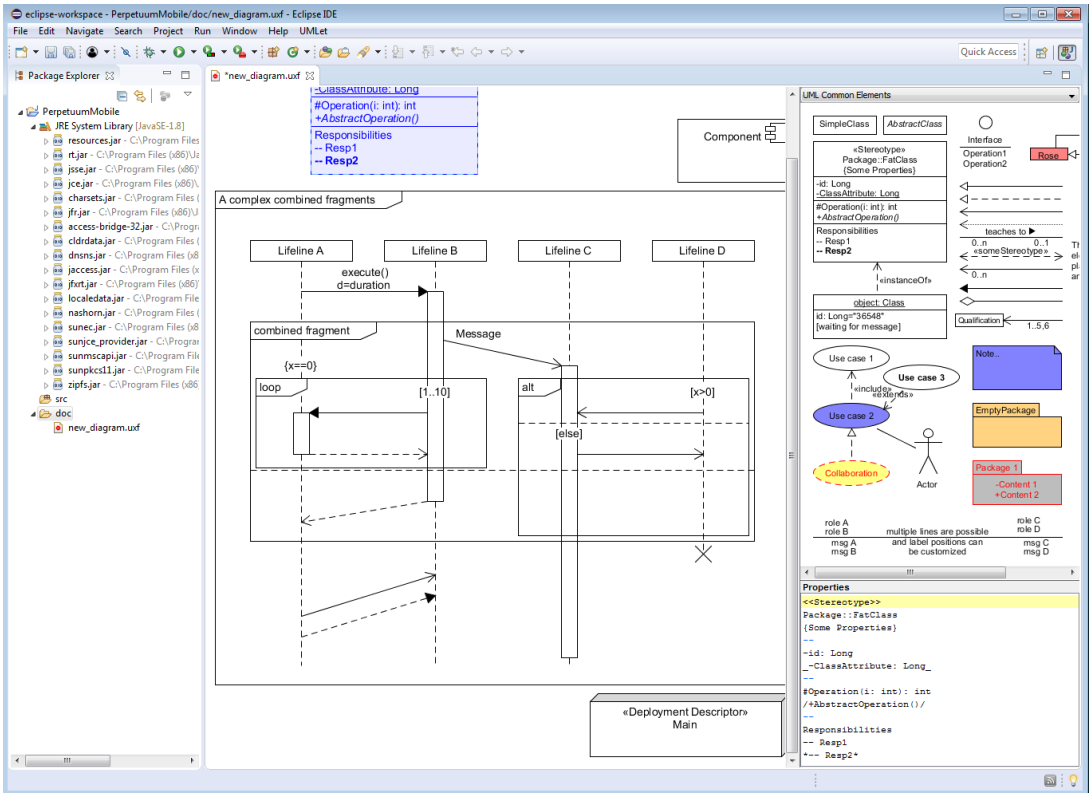
Разглежданата версия на PlantUML е V1.2022.13 (19 ноември, 2022). Тази версия на PlantLet може да бъде достъпвана чрез уеб сайт (plantuml.com), Java приложение или добавка за IDE - Eclipse, Visual Studio Code, Intellij IDEA, добавка за Confluence, инструмент на Atlassian. [2]

2.2. Основни характеристики

UMLet притежава опростен графичен интерфейс. UMLet се поддържа на различни платформи и IDE редактори като Eclipse и Visual Studio Code. Поддържа различни видове диаграми – Use Case, Interaction, Activity, State Machine, Class, Deployment, Composite Structure, Package, Collaboration. Използването на текст за форматиране прави употребата му по-интуитивна, след като дадения потребител научи маркиращият език. Предоставя голяма палета от UML елементи, които могат да бъдат персонализирани от потребителите чрез Java код. UMLet е с отворен код, безплатен за използване. Има наличие на уеб версия, която позволява на потребителите по-лесен достъп. [1]

Съответствията с PlantUML са доста. PlantUML също притежава опростен графичен интерфейс, поддържа се на различни платформи (добавка за Confluence) и IDE редактори като Eclipse и Visual Studio Code, Intellij IDEA . Поддържа различни видове диаграми – Sequence, Use Case, Class, Object, Activity, Component, Deployment, State, Timing, както и не - UML диаграми, като JSON, YAML, Gantt, Entity Relationship, MindMap. Използването на текст за форматиране прави употребата му по-интуитивна, след като дадения потребител научи маркиращият език. PlantUML е с отворен код, безплатен за използване. Има наличие на уеб версия, която позволява на потребителите по-лесен достъп. [2]

2.2.1. Графичен интерфейс на UMLet



Фиг. 1 Потребителски интерфейс на UMLet [1]

На фигура 1 е представен графичният потребителски интерфейс на UMLet. Той се състой от 3 основни елемента - текущата диаграмата, често използвани UML елементи, който могат да бъдат “drag and drop”-вани за изчертаването на диаграмата и свойства на до момента изчертаната диаграма, като например имената на използваните елементи.

1. Регион на редактора: Това е мястото, където може да се намери вашата активна диаграма. Командите за редактор са стандартни клавишни комбинации за Windows (т.е. CTRLs за запазване, CTRLc за копиране). Ако е отворен повече от един файл, те ще се появят под формата на няколко раздела в горната част на екран на редактора за лесен достъп и преглед. Компонентите в този прозорец могат да се плъзгат наоколо, уголемяват, завъртат и манипулират по друг начин с мишката. [3]

2. Регион на компонентите: Това е мястото, където се намират всички компоненти на вашите диаграми. За да се добави компонент към региона на редактора, щракнете два пъти върху компонента. [3]

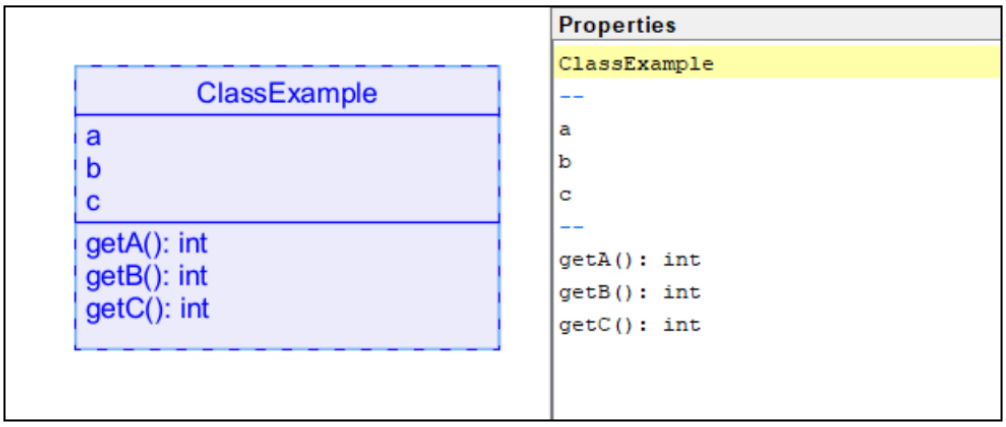
3. Помощ/Регион за редактиране на текст: Когато UMLet се отвори за първи път, това поле съдържа всички основни елементи на UMLet. Може също така да се променя размера на текста. Когато щракнете върху регион, тази област се превръща в област за редактиране на текст за елемента (например за промяна на името на класа). Когато щракнете върху празното платно, тази област ще се превърне отново в зона за помощ. [3]

2.2.2. Създаване на персонализиран елемент в UMLet

Фиг. 2 Създаване на персонален UML елемент в UMLet [1]

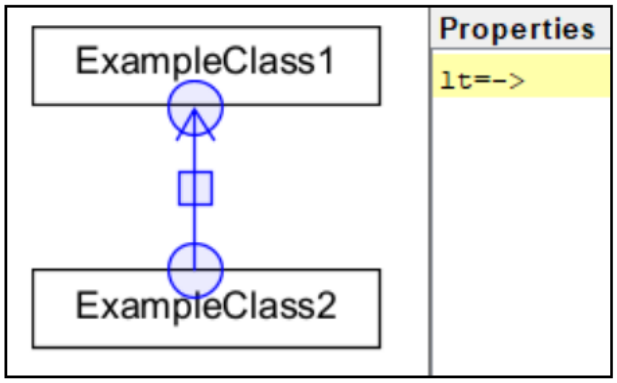
Създаването на собствен UML елемент е показано на фигура 2. Прозорецът отново съдържа 3 панела - свойства на новосъздаденият UML елемент, панел с изходният код на елемента и неговото графично представяне. Потребителят може да модифицира персонализираните елементи на две нива: той може да промени текста в панела със свойства и да види ефектите веднага в панела за визуализация. Но освен това в панела на изходния код той може да промени изходния код, който отговаря за интерпретирането на текста на свойствата и за изчертаването на елемента Кодът се компилира непрекъснато във фонов режим и незабавна графична обратна връзка се предоставя на крайния потребител в панела за визуализация. Ако кодът съдържа грешки, засегнатите редове на изходния код се маркират.

2.2.3 Създаване на клас в UMLet



Фигура 3. Създаване на клас в UMLet [1]

На Фигура 3, виждаме създаване на клас от предварително създадени UML елементи. За да редактираме класа и да добавим атрибути и методи, го избираме и в полето за свойства на класа задаваме името на нашия клас - "ClassExample" в случая. Разделител между атрибутите и методите на класа е символната комбинация "--". По този начин се създават отделни секции в класа.

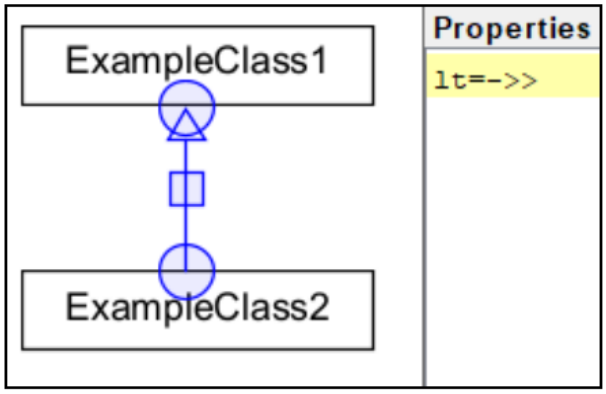


Фигура 4. Връзка между класове [1]

Фигура 5. Обратна връзка между класове [1]

На фигура 4 виждаме пример за връзка тип асоциация, сочеща от ExampleClass1 към ExampleClass2. UMLet ни позволява лесно да сменим посоката на връзката, като в панела за Свойства на 'променливата' "lt=", зададем стойност "->". По този начин вече връзката сочи от ExampleClass2 към ExampleClass1. (Фигура 5). Промяната на посоката в

UMLet става много по-лесно от други UML инструменти, които използват принципа “drag and drop”.



Фигура 6. Връзка от тип Наследяване [1]

Освен смяна на посоката, в панела “Свойства” можем да сменим директно и типа на връзката. Задавайки “lt=->>”, връзката от връзка от тип асоциация се променя и става връзка от тип наследяване (визуална репрезентация може да се види на Фигура 6).

2.2.3. Уеб сайтът UMLetino и UMLet plugin

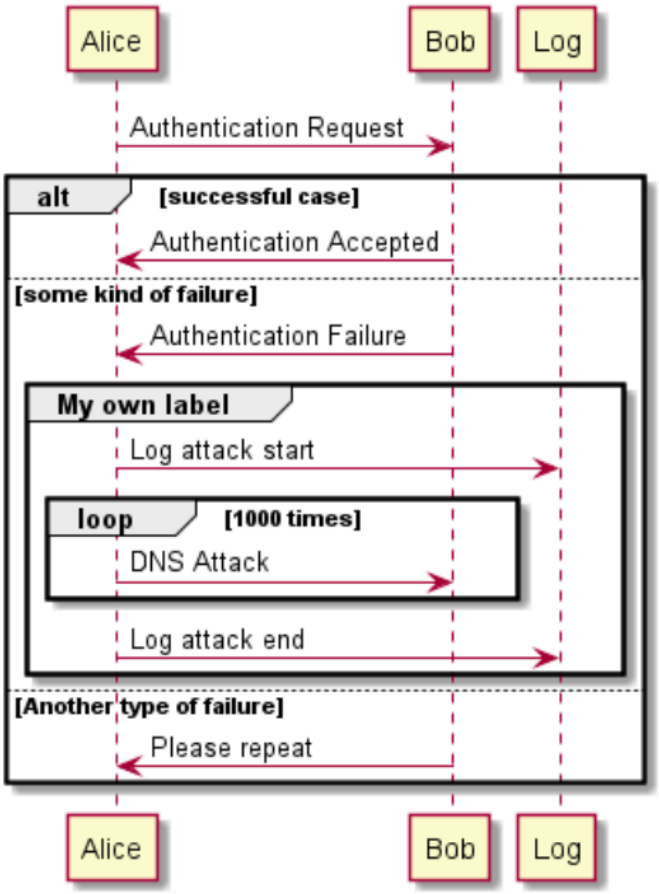
Графичният потребителски интерфейс и общите функционалности на уеб редактора UMLetino са същите като java приложението UMLet. Разликите идват от липсата на някои разширени функции за моделиране, който UMLet притежава като например генерирането на елементи от потребителите. Както и командата „save“ работи по различен начин, тя използва локално хранилище на браузъра (функция на HTML5). В UMLetino експортирането на файлове става само в формат ufx5 . След това диаграма запазена в този формат може да бъде отворена с UMLet и експортирана в някои от наличните формати (eps, pdf, jpg, svg). Въпреки това UMLet е постоянно присъстващ UML инструмент, който е полезен за изготвянето на бързи чернови на диаграми. Фактът, че е достъпен без теглене и инсталация го прави по-удобно за използване от различни устройства. Друг проблем, който са адресирали създателите е наличието на много контекстни менюта и диалогови прозорци при други инструменти за UML. Те казват, че други инструменти разчитаха на твърде много изскачащи диалогови прозорци, които са разсейващи и изискват с течение на времето значителен умствен фокус от потребителя. Затова идеята на UMLet/UMLetino е да позволи на потребителите редакция на елементите с минимални проблеми. [1]

Съществуват и две версии на UMLet като добавки за IDE. UMLet Eclipse, който може да бъде инсталиран от Eclipse Marketplace и версията за visual Studio Code налична в Visual Studio Marketplace. [1]

2.2.4. Диаграми на последователностите в PlantUML

```
@startuml
Alice -> Bob: Authentication Request

alt successful case

    Bob -> Alice: Authentication Accepted

else some kind of failure

    Bob -> Alice: Authentication Failure
    group My own label
    Alice -> Log : Log attack start
        loop 1000 times
            Alice -> Bob: DNS Attack
        end
    Alice -> Log : Log attack end
    end

else Another type of failure

   Bob -> Alice: Please repeat

end
@enduml
```

Фигура 7. Пример за Sequence диаграма на PlantUML [2]



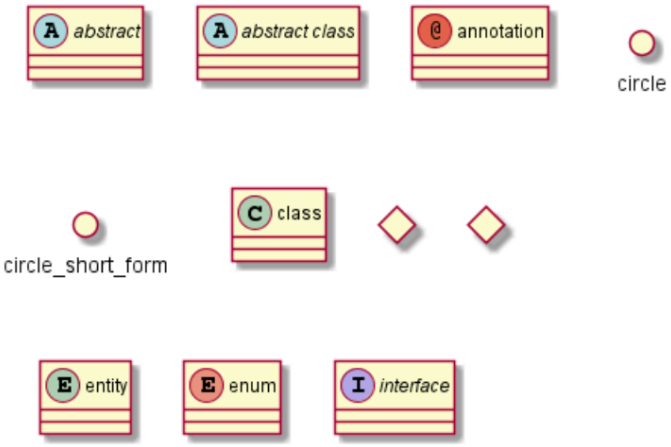
Фигура 8. Визуална репрезентация на Фигура 7 [2]

Диаграмите на последователността са единият от типовете диаграми, в които може много лесно да се види, колко лесен е за използване PlantUML - единственото нещо, което ни е необходимо е да знаем езика. Както се вижда на Фигура 7, всеки PlantUML файл започва с @startuml, и съответно завършва с @enduml. Между тези две команди се дефинират участниците (по желание, тъй като първоначалното им дефиниране може да бъде пропуснато, но в такъв случай всички участници ще бъдат изобразявани като правоъгълници, ако желаем да създадем участник Иван, който да бъде изобразен като човек следва да декларираме “Ivan : actor” в началото). След това чрез стрелки и съответните им етикети, описани след ‘:' можем да изчертаем бързо и лесно дори доста сложни и обемни на пръв поглед диаграми, както може да се види на Фигура 8.

2.2.5. Диаграми на класовете в PlantUML

```
@startuml
abstract        abstract
abstract class  "abstract class"
annotation      annotation
circle          circle
()              circle_short_form
class           class
diamond         diamond
<>              diamond_short_form
entity          entity
enum            enum
interface       interface
@enduml
```

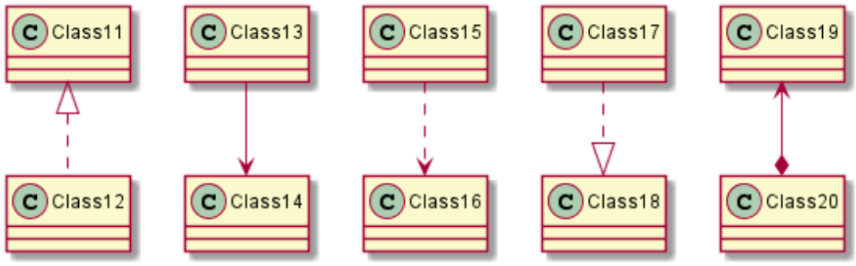
Фигура 9. Елементи на клас диаграми [2]



Фигура 10. Визуализация на елементите на клас диаграма[2]

В PlanUML могат да се чертаят и диаграми на класовете. На Фигура 9 и Фигура 10 могат да се видят различните елементи, които могат да бъдат използвани за изчертаването на този тип диаграми. А съответно на Фигура 11 и Фигура 12 могат да се видят употреба на различните връзки между класовете. Връзките между класовете са три вида - разширение - '<|—', композиция - '*—' и агрегация - 'o—'. Съответно връзките могат да бъдат изчертани с прекъснати линии ако символната комбинация '—' бъде заменена с '..'.

```
@startuml
Class11 <|.. Class12
Class13 --> Class14
Class15 ..> Class16
Class17 ..|> Class18
Class19 <--* Class20
@enduml
```



Фигура 11. Връзки между класове в PlantUML [2]

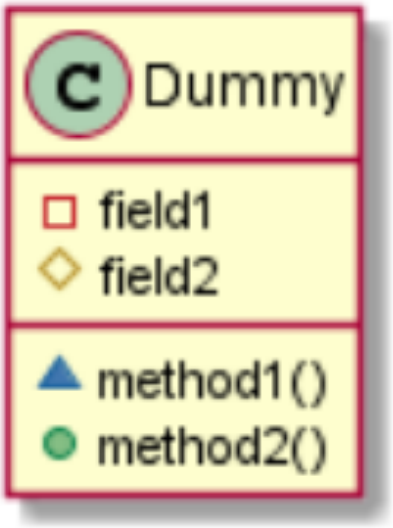
Фигура 12. Визуализация на връзки между класове в PlantUML [2]

На Фигура 13 и Фигура 14 се вижда употребата и съответната визуализация на различните идентификатори за видимост и достъп на атрибутите и методите на класа.

Character	Icon for field	Icon for method	Visibility
-	□	■	private
#	◇	◆	protected
~	△	▲	package private
+	○	●	public

```
@startuml

class Dummy {
 -field1
 #field2
 ~method1()
 +method2()
}

@enduml
```

Фигура 13. Пример за употреба за видимост на атрибути и методи [2]



Фигура 14. Визуализация на Фигура 13 [2]

2.4. Ограничения при използването на UMLet и PlantUML

- UMLet не поддържа генериране на код като други UML инструменти.
- Реверсивно проектиране – UMLet не може да генерира клас диаграма от наличен код
- PlantUML е Domain Specific Language - необходимо е известно време работа с него, за да бъде научен и съответно използван ефективно.

3. Сравнителен анализ

Ще разгледаме сравнение между UMLet и PlantUML и между UMLet и Visual Paradigm.

Критериите, по които ще сравняваме двата инструмента са – дали са с отворен код или не, интерфейс, последната версия на инструментите, могат ли да се интегрират с IDE редактори, кои UML диаграми включват, използват ли актуална версия на езика UML (UML 2), поддържат ли реверсивно проектиране, какъв е принципа на използване.

3.1. Критерии за сравнение

Избрани са точно тези критерии за сравнение, тъй като чрез анализ на графичния интерфейс на инструментите, можем да разберем дали те ще са достатъчно интуитивни за използване от потребителите. На база датата, на която е излязла последната версия можем да разберем дали тяхната поддръжка продължава да е актуална и дали инструментите се развиват и в последно време. IDE редакторите, с които са интегрирани инструментите говори за това колко използван е инструментът от потребителите. Коя версия на UML използват, тъй като след излизането на UML 2.0 има промяна в диаграмите.

Кои диаграми включват в техния инструментариум. Поддържат ли инструментите реверсивно проектиране. Какъв е принципа на използване при тези инструменти.

3.2. Сравнение на UMLet и PlantUML

Таблица 1: Сравнение на UMLet с PlantUML

Критерий	UMLet	PlantUML
Отворен код	Да	Да
Последна версия	15.0, 29 април 2022	V1.2022.13, 19 ноември 2022
Интерфейс	уеб сайт, приложение, добавка за IDE	уеб сайт, приложение, добавка за IDE, добавка за Confluence
Интеграция с IDE	Eclipse, Visual Studio Code	Eclipse, Visual Studio Code, Intellij IDEA, VIM
UML 2	Не	Да
UML диаграми	Use Case, Interaction, Activity, State Machine, Class, Deployment, Composite Structure, Package, Collaboration	Use Case, Sequence, Activity, State, Class, Deployment, Package, Component, Object, Timing and UI mock diagrams
Reverse Engineering	Не	Да с puml2code – C#, C++, Java, PHP, Python
Начин на употреба	Частично drag and drop	Изцяло текстово

При изчертаването на по-просто диаграми, UMLet ни предоставя по-бърз начин за изобразяване на диаграмата. Докато използването на PlantUML ни улеснява при създаването на по-сложни диаграми. Разликата идва от това, че PlantUML е Domain Specific Language - ефективното използване на инструмента започва след ползване и опит. Докато при UMLet човек може много по-бързо да се запознае и да започне да създава различни видове UML диаграми.

Двата редактора имат уеб версии, с ограничение откъм някои функционалности. При уеб версията на PlantUML се поддържа експортиране на диаграмата в png, svg формати и ASCI art, не е налично експортиране от тип ufx (както е при UMLet), което означава че после не можем да ескпортираме диаграмата и да я редактираме в наличните други среди. Това, което потребителите могат да направят е да запазят текстовото описание на диаграмата, за да могат впоследствие да я редактират в друга среда.

И двата инструмента са с отворен код, безплатни, което ги прави по-предпочитани пред други платени версии. Най-често към тях се обръщат хора, които тепърва започват да

изучават UML диаграми и желаят безплатен инструмент за изчертаването на диаграмите. Хората също така предпочитат софтуер с отворен код поради редица причини, включително контрол (кода може да бъде изследван и променян), софтуера може да бъде използван за всякакви цели, няма ограничение.

Нови версии на UMLet излизат сравнително рядко, което не е предпочитано от потребителите, тъй като инструмента не се развива и не коригира негови грешки. Въпреки, това UMLet е достигнал е едно ниво включващо всички важни функционалности за неговото използване. PlantUML се актуализира редовно като постоянно биват добавяни нови функционалности, както и non-UML диаграми, което го прави по универсален за работа.

Интегрирането с различни IDE редактори отново правят инструментите пополярни и използвани от различни потребители. Тъй като всеки би предпочел да използва инструмента в среда, която вече познава. UMLet и PlantUML могат да се добавят като плъгини на Eclipse и Visual Studio Code. PlantUML освен интеграция в тези два редактора може да бъде добавян като плъгин за VIM, IntelliJ.

Друга основна разлика в инструментите е, че UMLet не поддържа версия UML 2. Това означава, че има разлика в диаграмите, които можем да изчертаем в инструмента. UMLet не поддържа реверсивно проектиране, докато PlantUML го поддържа чрез puml2code. Езиците, които поддържа са C#, C++, Java, PHP, Python, Ruby, TypeScript.

UMLet частично се основава на drag and drop принципа, въпреки че в него са елиминирани множеството контекстни менюта и бутони. Той вместо това включва панел, които притежава е 1:1 мащабен модел на елементите. PlantUML използва само текстово описание, което е минус за потребителите при началното запознаване с инструмента. Ако целта на даден потребител е бързо и лесно създаване на диаграма, то той би се обърнал към UMLet.

3.3. Сравнение на UMLet и Visual Paradigm

Таблица 2: Сравнение на UMLet с Visual Paradigm.

Критерий	**UMLet**	**Visua lParadigm**
Отворен код	Да	Не
Последна версия	15.0, 29 април 2022	17.0, 1 август 2022
Интеграция с IDE	Eclipse, Visual Studio Code	Eclipse, Visual Studio Code, Intellij
UML 2	Не	Да

UML диаграми	Use Case, Interaction, Activity, State Machine, Class, Deployment, Composite Structure, Package, Collaboration	Class, Use case, Sequence, Communication, State machine, Activity, Component, Deployment, Package, Object, Composite structure, Profile, Timing, Interaction overview
Drag and drop	Частично	Да
Reverse engineering	Не	Да – Java, C++, .Net, Python, Ruby
Цена	Безплатен	Платен

Разликите в интерфейсите на двата инструмента са големи, тъй като те са създадени с коренно различни идеи. Интерфейсът на UML е опростен, а този на Visual Paradigm сравнително комплексен. Интерфейсът на Visual Paradigm е по-модерен от този на UMLet. От друга страна интерфейса на UMLet е много по-интуитивен в сравнение с този на Visual Paradigm. Много по-лесно може да се започне работа и времето за обучаване е сравнително малко при UMLet. Интерфейсът на Visual Paradigm, макар и модерен е доста комплексен и претрупан, потребителите се нуждаят от добро познаване на UML диаграмите и начина на работа с инструмента. Той съдържа в себе си множество от функции, като голяма част от тях се използват рядко, но те все пак не са скрити за потребителя. Затова този инструмент е предпочитан за по задълбочена работа с диаграмите, тъй като предлага повече опции и функционалности. Използва се повече за финални изображения на диаграмите, докато UMLet е по-предпочитан за изготвяне на чернови на диаграми (заради бързото им създаване).

Visual Paradigm също притежава уеб базирана версия, която иска задължителна регистрация в система за разлика от другите два разгледани инструмента. Въпреки тази допълнителна стъпка, това подсигурява, че работата по нашите диаграми няма да бъде изгубена.

Докато UMLet е отворен код и е напълно безплатен, Visual Paradigm не е. При Visual Paradigm е безплатна е само Community edition версията за некомерсиални цели. Останалите версии на Visual Paradigm са с месечен абонамент, въпреки това безплатната версия притежава голям брой функционалности, които са напълно достатъчни за повечето потребители. Също така те предлагат безплатен 30 дневен тестови период.

В крайна сметка UMLet ни предоставя по-голяма свобода, тъй като няма ограничение в целите, за които може да бъде използван.

Visual Paradigm също предлага повече редактори в които може да се интегрира – Eclipse, Visual Studio Code, NetBeans, IntelliJ, което е предимство за достигането до повече потребители. Visual Paradigm за разлика от UMLet поддържа най-актуалната версия на UML – UML 2.5. В него са включените всички 14 UML диаграми.

Visual Paradigm поддържа реверсивно проектиране на следните езици – Java, C++, .Net, Python, Ruby. Основава се изцяло на drag and drop принципа.

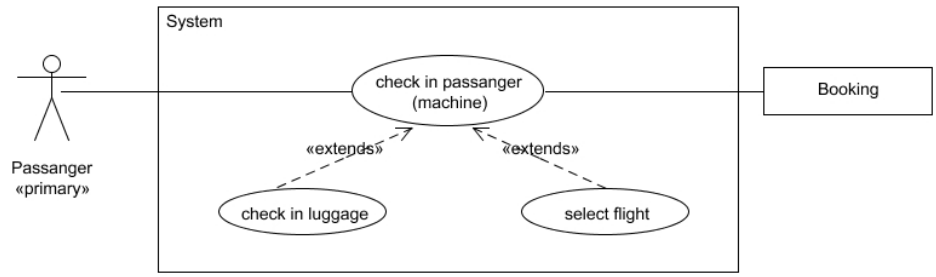
Има много предимства за използването на Visual Paradigm. Той предлагат по-добра графична среда за работа и е придобил по-голяма зрялост и популярност с времето.

4. Примери на използване

UMLet и PlantUML могат да бъдат използван от програмисти, дизайнери, архитекти, използващи UML в тяхната работа. Те могат да използват UML за създаване или редактиране на вече съществуващи диаграми. Наличието на уеб версии на инструментите ги прави наистина подходящ за бързото изчертаването на диаграми без необходимостта от теглене и инсталация. UMLet няма някои важни функционалности, които биха намерили приложени в работна среда – генериране на код или генериране на диаграми от съществуваш код, това прави тяхната употреба в професионалната сфера по-скоро за създаване диаграми с цел визуализация на връзките между различни елементи.

Двата инструмента са по-подходящ за потребители, които тепърва се запознават и обучават в използването на езика UML или пък хора, на които по-рядко им се налага използването му. Инструментите са лесни и достъпени за хората със своя опростения UI.

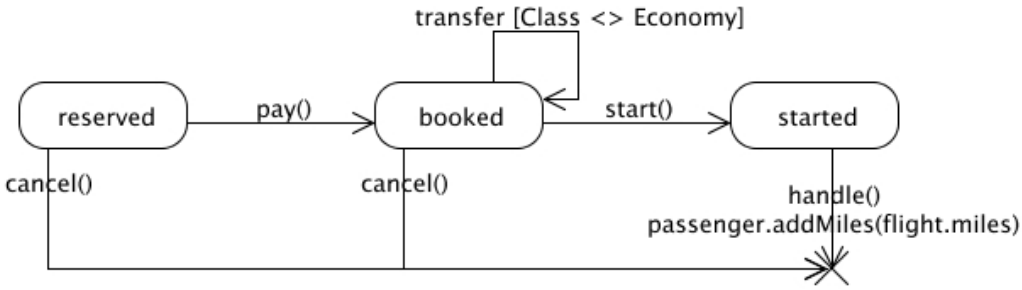
4.1. Use Case диаграма изчертана на UMLet



Фигура 15. Use case диаграма [8]

На тази фигура можем да видим Use Case диаграма изчертани с UMLet. Диаграмата е доста проста, както и извозваният инструмент, но крайната идея да се види какви са случаите на употреба на системата е успешно изпълнена.

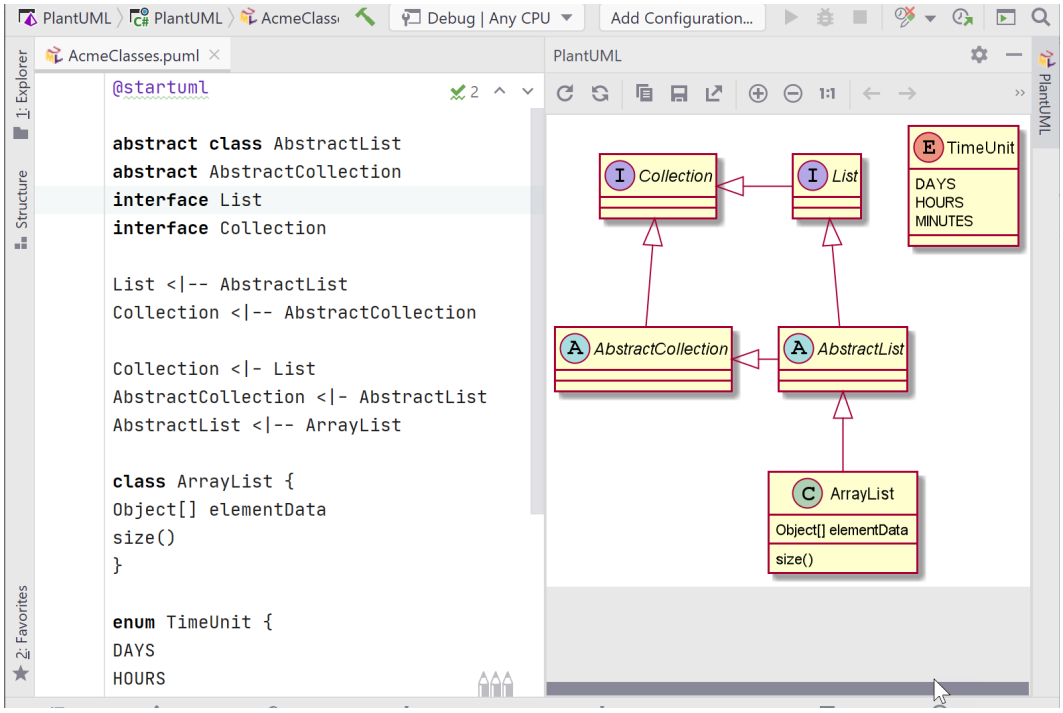
4.2. Object state диаграма изчертана на UMLet



Фигура 16. Диаграма на състоянията[8]

На тази фигура пък виждаме една диаграма на състоянията. Отново диаграмата е много изчистена от стилистична гледна точка - не са използвани различни цветове, форми, и т.н., но изготвянето на една такава диаграма в инструментът UMLet става изключително бързо, което е основният плюс на инструмента.

4.3. Клас диаграма в PlantUML

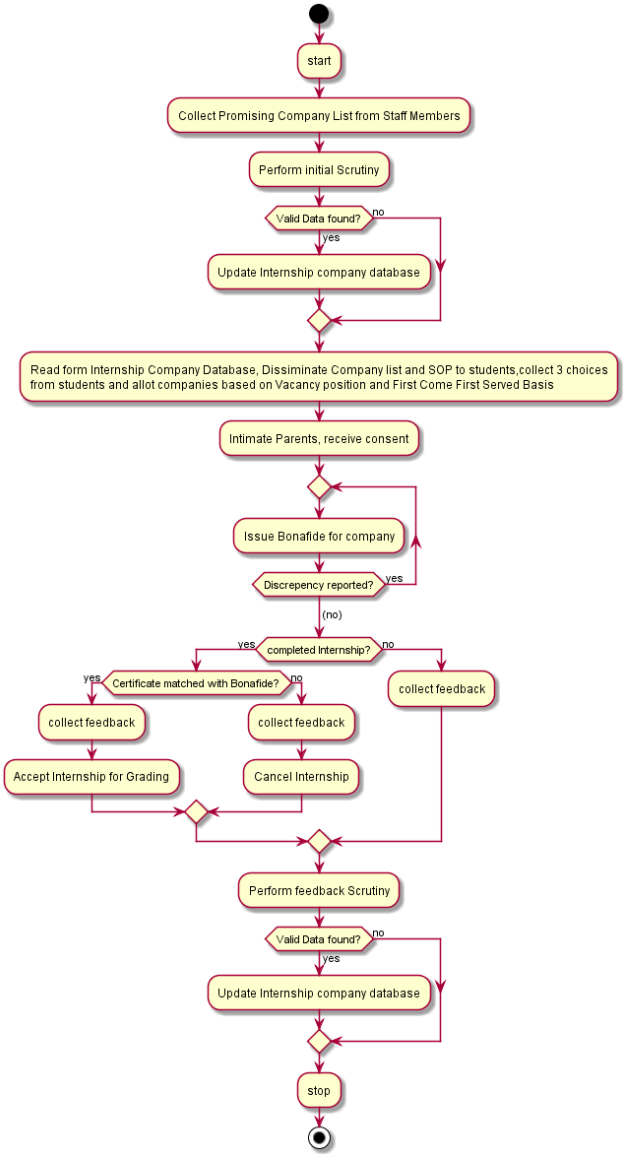


Фигура 17. Клас диаграма [9]

Силата на PlantUML е в това, че щом езикът използвана за изчертаване на диаграми бъде научен за няколко реда ‘код' могат да будят изчертавани големи с сложни диаграми за кратко време. Друго предимство на инструмента е, че във всеки един момент можем да отворим и да променим кода изключително лесно и например да превърнем всички

връзки от тип агрегация във връзки от тип композиция, докато тази същата промяна би била доста по-трудна в инструменти, които използват drag and drop.

4.4. Диаграма на дейностите в PlantUML



Фигура 18. Диаграма на дейностите[2]

Фактът, че инструментът не използва drag and drop също така може да бъде използван като плюс при разработването на детайлни диаграми на системи, диаграми, в които има множество връзки между различните елементи, тъй като при drag and drop внимателно трябва да се поставят елементите, така че да не стават нечетими диаграмите или съответните елементи да не излизат от зоните си на обхват - щом зададем, е даден

елемент е част от някакъв обхват PlantUML при нужда ще увеличи визуалната репрезентация на обхвата за нас и няма да се налага ръчно да пренаредим елементите.

5. Добри практики и методи за използване

• Големите диаграми, съдържащи купища елементи, всъщност предават по-малко информация от малките фокусирани диаграми. Когато чете толкова голяма диаграма, аудиторията няма да знае върху какво да се съсредоточи и тъй като там има твърде много, за да се обърне внимание на всички елементи.

• Опитайте се да избягвате пресичането на две линии във вашата диаграма. По някаква причина добре проектираните модели нямат проблем с пресичащите се линии.

• Ако направите линиите в диаграмата само хоризонтални или вертикални и всичките са само прави ъгли, прави диаграмата незабавно да изглежда по-добре.

• Когато чертаете йерархии на обобщение или реализация върху диаграма, винаги се уверете, че родителските елементи са по-високи от дъщерните, така че стрелките винаги да сочат нагоре.

• Ако искате да предадете съобщението, че вашият анализ е добре обмислена конструкция, която ще реши проблемите на заинтересованите страни, тогава по-добре се покажете с хубава и чиста диаграма.

• Създавайте само по една диаграма в един файл.

• Влагайте структури в PlantUML, това ви позволява да ги групирате по-лесно в една сплотена структура.

• Използвайте псевдоними, които са уникални във PlantUML файла.

• В PlantUML дефинирайте връзките в рамките на обхвата/контекста, към който принадлежат.

• Понякога искаме да има множество потоци в рамките на една диаграма. В този случай и в зависимост от случая използвайте цветно кодиране с всяка стрелка (напр. A -> B #Green : текст) и добавете легенда накрая.

• Използвайте автономериране на събития в диаграма на последователност

• Подравнявайте текста в диаграма на последователност

• Полупрозрачни групови фонове в диаграми на последователности

• Групирайте участници в полета, защото когато последователността е дълга, можете бързо да идентифицирате обхвата на вертикала, който гледате, дори когато участниците са извън полезрението.

6. Заключение и очаквано бъдещо развитие

Развитието на UMLet е бавно, нови версии излизат рядко и не съдържат в себе си нови функционалности, за сметка на това пък PlantUML често излизат с нови версии. И двата инструмента са достигнали ниво на развитие, в което се включват всички базови, а и някои небазови функционалности за създаване на различни видове UML диаграми. Те са имплементирани и работят добре. Позволяват ни бързо и лесно създаване на диаграми, без много голяма допълнителна подготовка за изучаване на самата технология или съответно изучаване на един не много сложен език за употребата на PlantUML.

Наличността на инструментите като добавки за различни IDE редактори, както и наличието на уеб версии, ги прави много достъпни за различни потребители. Има много други алтернативни инструменти, които могат да бъдат използвани вместо тях, като например Visual Paradigm, StarUML, MagicDraw, SmartDraw. Те ни предлагат по-добра графична среда за работа и са придобили по-голяма зрялост и популярност във времето. Въпреки това има хора, които предпочитат и използват UMLet и PlantUML, заради много от техните положителни страни. Това, че са изцяло безплатни, че са достъпени и през уеб браузър както и факта, че са много интуитивни за нови потребители и предлагат възможност за бързо изчертаване на диаграми.

7. Използвани литературни източници

1. UMLet - https://www.umlet.com/
2. PlantUML Guide - https://plantuml.com/
3. Introduction to UMLet - https://engage-csedu.org/sites/default/files/Lab2UMLDesignDebuggingpart1.pdf
4. PlantUML Tips and Tricks - https://www.codit.eu/blog/plantuml-tips-and-tricks/?country_sel=be
5. UML Best Practices - https://bellekens.com/2012/02/21/uml-best-practice-5-rules-for-better-uml-diagrams/
6. Visual Paradigm - https://www.visual-paradigm.com/
7. UMLetino - https://modeling-languages.com/umletino-free-online-uml-tool-fast-uml-diagrams/?fbclid=IwAR3JSlUosGRx85q5UPL6s7mdi4k5lstFvPLSr5DW8ItZbMR9p6RYQaMTY3s
8. UMLet Diagrams - http://www.itmeyer.at/umlet/uml2/
9. PlantUML Class Diagram - https://blog.jetbrains.com/dotnet/tag/uml/