**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 15: Ревю на MagicDraw UML

# Есе

*Автор:*

*Мануел Демирджиян, фак. номер 26339*

Декември, 2020

София

# Съдържание

# 1 Въведение

Настоящата работа представлява ревю и дълбок поглед над MagicDraw. MagicDraw е един от най-разпространените визуални инструменти за моделиране на UML (Unified Modiling Language), SysML (Systems Modeling Language), BPMN (Business Process Model and Notation) и UPDM (Unified Profile for DoDAF/MODAF). Създаден за бизнес анализатори, софтуерни анализатори, програмисти и QA инженери, този универсален инструмент улеснява анализа и проектирането на обектно-ориентирани системи и бази от данни. В следващите секции бъдат изложени различните характеристики на MagicDraw, начина на използване и ще бъде направен сравнителен анализ с продукт, представляващ пряка конкуренция на инструмента.

# 2 История

Софтуерната компания No Magic е основана през 1995г., като тяхното мото е, че няма „магия“ за разработване на добър софтуер. По това време на дузината служители на компанията им се е наложило да използват най-популярния тогава инструмент, а именно Rational Rose, за който No Magic са взели решение да придобият лицензи за целия си екип. Пол Дикинсън, основателят на компанията, се обажда на представителите на Rational с молба за ценова оферта. Отговорът от тяхна страна е бил следния – “Цената за едногодишен лиценз е 3800 долара”, което умножено по 12 прави 46 000 долара. Като резултат от това, Пол Дикинсън решава, че ще бъде по рентабилно за компанията му, те самите да създадат инструмента, който да използват, като се знае, че с времето необходимостта от лицензи ще нараства. Java и UML са нововъзникващи технологии, така че е взето решение да се разработи собственият инструмент за чертане на UML на компанията No Magic, който се нарича Magic Diagrams. В последствие името на проекта е променено на MagicDraw. В момента MagicDraw е най-популярният инструмент за моделиране на UML диаграми, модели и други, през 2002 продуктът получава престижна награда от списанието “Java Developer’s Journal” за най-добър инструмент за моделиране на Java и най-добрият инструмент за разработване в екип. Днес No Magic продължава да инвестира в инструменти за разработчици. Компанията продължава да разширява продуктовата линия MagicDraw, така че да включва инструменти не само за дизайнери, но и за ИТ бизнес специалисти, софтуер и бизнес архитекти.

## 2.1 Основни характеристики

Основните характеристики на MagicDraw се състоят в специфичните инструменти, които предлага софтуера за визуално моделиране на обектно-ориентирани системи, механизми за кодово инженерство (с пълна поддръжка за двупосочни връзки за

програмните езици J2EE, C#, C++, CORBA IDL, .NET, XML Schema, WSDL), както и за моделиране на схеми на бази от данни и reverse engineering.

MagicDraw поддържа UML 2 мета модел, най-актуалния XML стандарт за съхранение на данни и най-популярните програмни езици за внедряване. За разлика от други среди за моделиране и архитектура на UML MagicDraw улеснява потребителите при внедряването на среда на разработка на софтуер, която намира приложение в голям процент от бизнеса. Това става благодарение на Open API технологията.

- Интуитивен интерфейс – MagicDraw предоставя интуитивни инструменти в рамките на много добре проектиран GUI, който позволява на потребителите да моделират, без да се налага да отделят време за учене за контролите. Това води до много бързо подобряване на производителността на разработчиците.

- Извличане на модели от съществуващ изходен код - Обратното проектиране на MagicDraw е най-бързият начин за получаване на UML модели от изходния код на Java, C #, C ++, CORBA IDL, EJB 2.0, aDDL, CIL (MSIL), WSDL и XML Schema. Автоматичното генериране на диаграми на последователности от изходния код на Java добавя по-подробен изглед на системата.

- Визуализиране на модели - Автоматичното генериране на статична структура, зависимост от пакети и йерархични диаграми на MagicDraw позволява множество изгледи на един и същ модел. Автоматичното генериране на една йерархична диаграма изисква само няколко секунди в сравнение с часовете, необходими за извършване на същата работа ръчно.

- Генериране на сорс код от UML модел - MagicDraw генерира код за Java, EJB, C#, C++, CORBA IDL, DDL, WSDL, XML схема. Интеграциите с най-популярните IDE (Eclipse, IBM WSAD и RAD, Borland JBuilder, IntelliJ IDEA, NetBeans, Sun Java Studio) премахват необходимостта от вградена IDE на MagicDraw.

- Domain specific language (DSL) – позволява адаптация на MagicDraw към определен профил и домейн за моделиране.

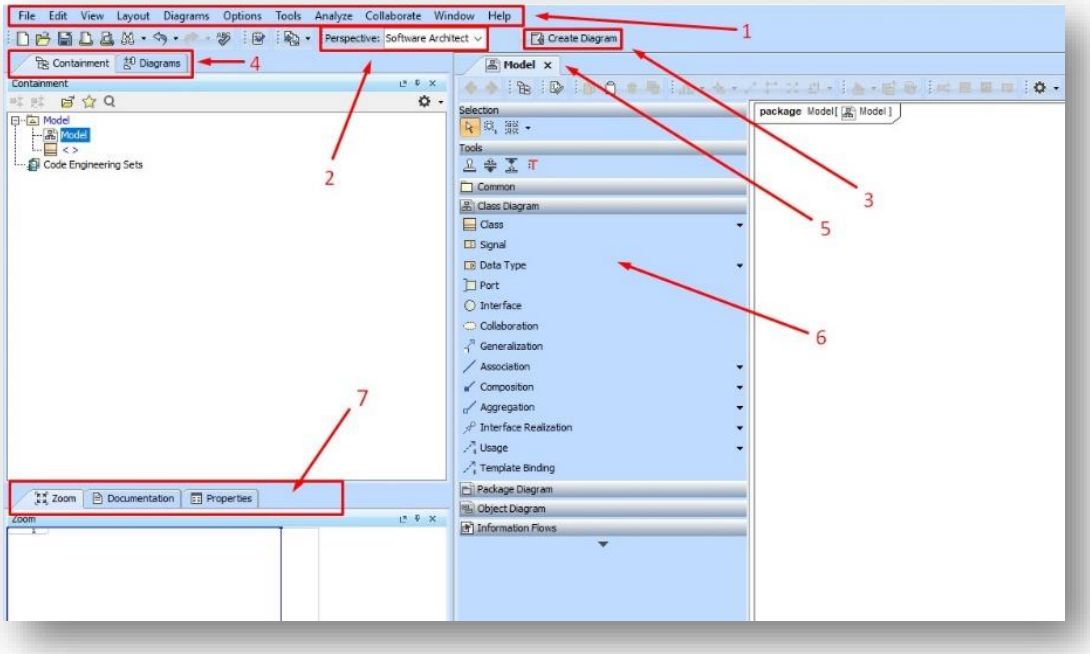
- Сътрудничество и развитие на екип - Използвайки екипния сървър на MagicDraw, множество разработчици могат да работят едновременно върху един и същ модел. Това прави сътрудничеството на екипа значително по-бързо и осигурява просто управление на конфигурацията, контролиран достъп до всички общи артефакти, отдалечен достъп до модела и оптимален начин за управлението му и избягване на конфликти.

- Документация - С адаптивния механизъм за генериране на отчети MagicDraw потребителите могат да адаптират документацията така, че да отговаря на вътрешния процес на разработка.

- Разширимост – С UML профилите и персонализираните диаграми разработчиците могат да разширят стартния UML 2, така че да отговаря на кокретния домейн на проблема. С помощта на Open API може да се разшири функционалността, за да се включат нови дизайнерски модели, показатели и трансформации, както и други плъгини.

### 2.1.1 GUI / Интерфейс

Интерфейсът на MagicDraw е специално проектиран за да предостави на потребителите възможно най-бързия начин да започнат работа по проектите си. Лесният достъп до най-често срещаните операции е опорна точка на потребителския интерфейс на MagicDraw. Тъй като всички основни команди са достъпни само с едно щракване, разработчиците могат да се съсредоточите върху моделирането. Предоставя се голям избор от методи, достъпни с едно щракване: от стандартни менюта, контекстни менюта, преки пътища или ленти с инструменти. Проучванията показват, че с MagicDraw можете да завършите задачите си с половината стъпки, изисквани от други инструменти.

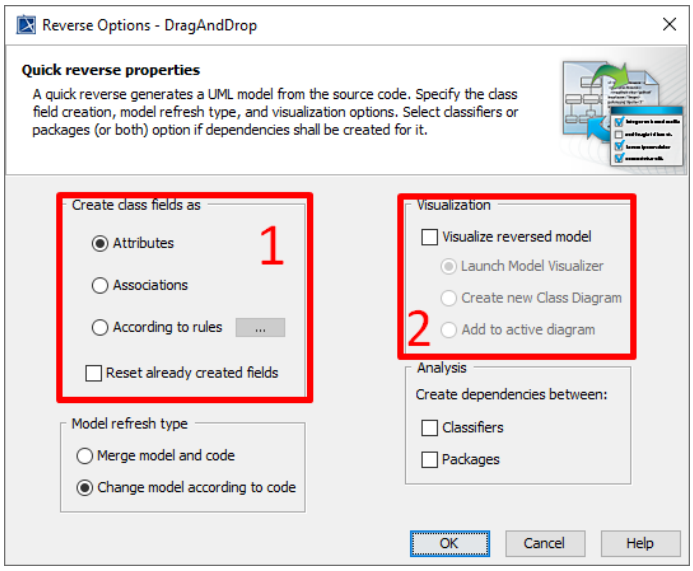



Фиг.1 MagicDraw GUI / интерфейс

1) Стандартно header меню – съдържа в себе си всички необходими настройки, функционалност и инструменти
2) Перспектива (DSL) – Специализирани профили за различен тип проекти. Позволява промяната на целия GUI спрямо типа на проекта и специалиста, който го разработва.
3) Бутон за създаване на нова диаграма, която се доква към (5).
4) Напълно персонализираща се док система за различни инструменти. Могат да бъдат добавени или премахнати набор от инструменти с drag & drop система.
5) Тук се позиционират всички новосъздадени диаграми.
6) Списък с налични инструменти за моделиране на диаграми.
7) Функционалност за увеличаване на конкретен участък от диаграма, генериране на проектна документация и настройки. Има възможност за докване на допълнителни инструменти.

### 2.1.2 Генериране на модели и генериране на код

Специалният инструмент Java Reverse, позволява на MagicDraw разработчиците да използват входен сорс код с цел да извлекат UML модел. Тази методика се нарича Code Reverse. За тестването на тази функционалност е използван C# скрипт, който е взет от Unity проект. Функционалността, която изпълнява скрипта се състои в възможността на потребителя да задържа и провлачва обекти по екрана. Резултатите са изложени по-долу.

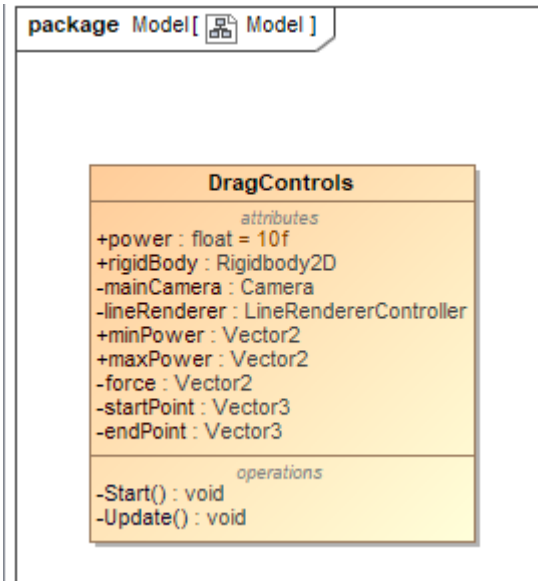



Фиг.2 Настройки за Code Reverse операция в MagicDraw

1) Избират се правилата, по които да се генерира модела
2) Избира се начина на визуализация – този оператор дава възможността да се визуализира даден модел след Code Reverse по 3 различни начина. Резултатът може да се постави в нова диаграма или вече съществуваща такава.

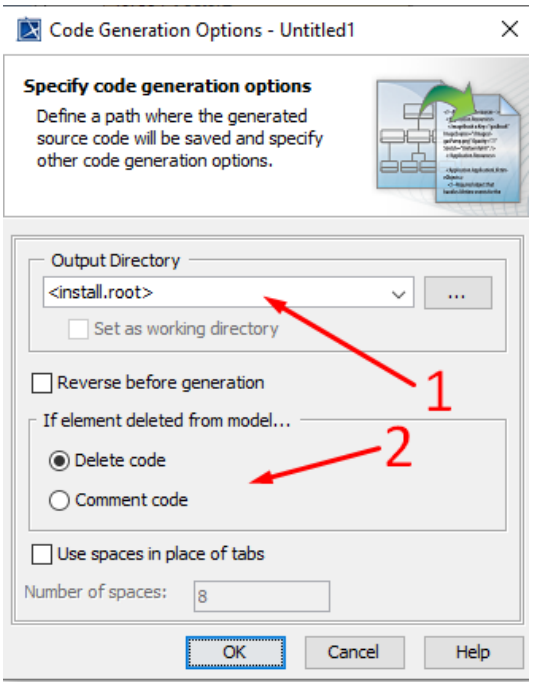
```
public class DragControls : MonoBehaviour
{
    public float power = 10f;
    public Rigidbody2D rigidBody;

    Camera mainCamera;
    LineRendererController lineRenderer;

    public Vector2 minPower;
    public Vector2 maxPower;
    Vector2 force;
    Vector3 startPoint;
    Vector3 endPoint;



    private void Start()
    {
        mainCamera = Camera.main;
        lineRenderer = GetComponent<LineRendererController>();
    }

    private void Update()
    {
        if (Input.GetMouseButtonDown(0))
        {
            startPoint = mainCamera.ScreenToWorldPoint(Input.mousePosition);
            startPoint.z = 15;
        }

        if(Input.GetMouseButton(0))
        {
            Vector3 currentPoint = mainCamera.ScreenToWorldPoint(Input.mousePosition);
            startPoint.z = 15;
            lineRenderer.RenderLine(startPoint, currentPoint);
        }

        if (Input.GetMouseButtonUp(0))
        {
            endPoint = mainCamera.ScreenToWorldPoint(Input.mousePosition);
            endPoint.z = 15;

            force = new Vector2(Mathf.Clamp(startPoint.x - endPoint.x, minPower.x, maxPower.x),
                                Mathf.Clamp(startPoint.y - endPoint.y, minPower.y, maxPower.y));

            rigidBody.AddForce(force * power, ForceMode2D.Impulse);
            lineRenderer.EndLine();
        }
    }
}
```

Фиг.3 Сорс код ( генериране на UML диаграма от готов сорс код)




Фиг.4 Генерирана диаграма от сорс код

При генерирането на код от диаграма се извършват почти същите действия. Резултатът е създаване на C# клас, съдържащ в себе си всички дефиниции, като преди това се проверява за синтактични грешки в кода. Също така Magic Draw гарантира за всички направени промени по диаграмата и обновява файловете без значение дали те са коригирани, изтрити или са направени допълнителни връзки между тях. Тази функционалност на Magic Draw го прави изключително силен и ефикасен в сферата си, понеже значително намалява времето за разработка и шансовете за грешки породени от човешки фактор.




Фиг. 5 Настройки за генериране на код от готова диаграма

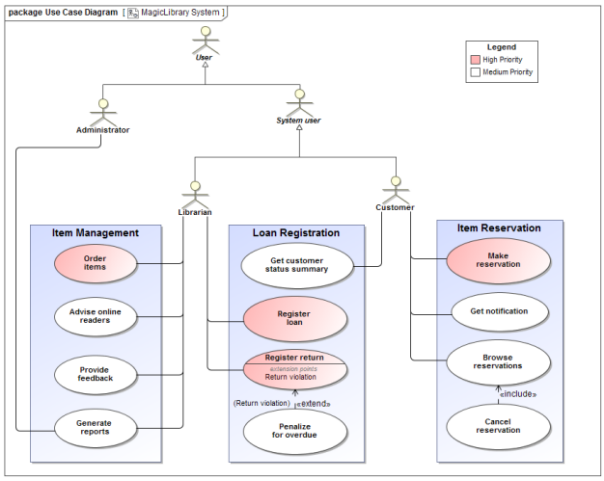
1) Избира се директория в която да бъде експортиран файла
2) Избира се какво да се случи с елемент от код, който е променен

### 2.1.3 Визуализиране на модели

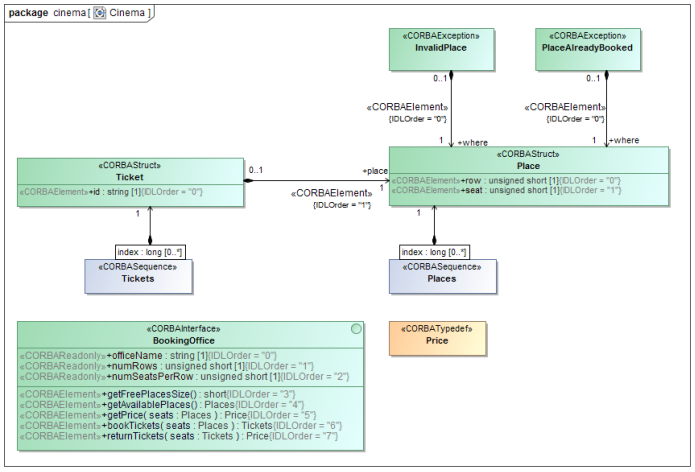
MagicDraw дава възможност да бъдат визуализирани едни и същи модели по различни начини в зависимост от това какъв тип специалисти ги обработват.

| | | |
|---|---|---|
| Programmer | QA Engineer | Project Manager |
| Designer and Analyst | Technical Writer | Corporate Executive |

Фиг.4 Таблица с типове профили




Фиг.5 Use Case диаграма




Фиг.6 Class диаграма

### 2.1.4 Domain Specific Language (DSL)

Domain Specific Language Engine позволява адаптация към определен профил и домейн за моделиране, като по този начин се позволява персонализиране на множество графични интерфейси, инициализация на модел, добавяне на семантични правила и създаване на собствени диалогови прозорци за спецификация и интелигентни манипулатори. Възможността да се използват множество специфични персонализации помага да се направи MagicDraw по-добре ориентиран към различни платформи и технологии. Активното валидиране позволява проверка на конкретни модели на домейни в реално време и също така предлага помощ при появата на конкретни проблеми. DSL елементите могат да бъдат преобразувани във всеки подтип или по-общ тип, използвайки функцията ”Конвертиране в”. DSL позволява да се създават производни свойства, които позволяват разширяване на UML метамодел или неговия профил.

### 2.1.5 Сътрудничество и работа в екип

Колаборацията между разработчици е един от най-силните инструменти на MagicDraw. Членовете на екипа могат индивидуално да разработят отделни части от един модел и след това да ги обединят, за да образуват цялостен модел. В сравнение с индивидуалното моделиране, съвместоното моделиране е по-ефективно и води до по-добри резултати, тъй като всеки член използва своите най-добри умения и опит. Това е оптималният начин за управление на вашия модел и избягване на конфликти на версии. Сътрудничеството в MagicDraw се реализира благодарение на сървърните услуги, които предлага като решение.

1) Teamwork Cloud
2) MagicDraw Teamwork Server

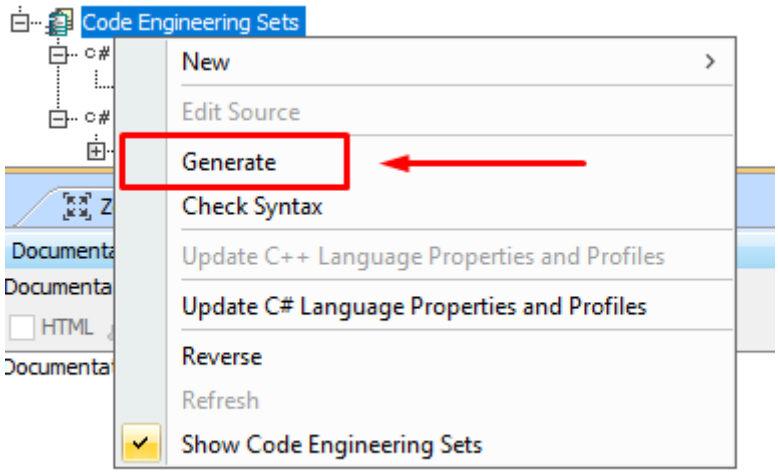
И двата сървъра предоставят хранилище за съхранение на проекти и потребители. Те също така поддържат управлението на разширенията на потребителите, както и проекти за споделяне и управление на версии.

Проектите, съхранявани в cloud системата, могат да бъдат достъпни чрез мрежата от множество клиенти, които използват MagicDraw. Различните потребители, в зависимост от тяхната роля в екип или предприятие, могат да имат различни решения за проектите.

Един и същи модел или дори една и съща диаграма могат да бъдат достъпни и модифицирани паралелно. Всеки потребител може незабавно да получи най-новата версия на модела, както и да извърши свои собствени промени.

### 2.1.6 Документация

MagicDraw притежава свойството да издава документация по проекта във формата на репорти. Използва се командата „Generate” за извличане на информация.




Фиг.7 Извличане на репорт от проект

Командата „Generate“ създава отчет на документ с получен набор от информация от аргументи и параметри. След това информацията се генерира като изходен файл към посочена от разработчика директория. По подразбиране аргумент са посочените данни за параметрите, които се извикват. Ако е посочена опцията -properties, аргументът е името на файл със свойства. Ето и някой от най-използваните команди:

- -project projectFullName
- -output outputFileName
- -template templateName
- -package packageNameList
- -element elemenNameList

```
generate -properties "C:\\output\\prop.properties"
```

```
generate -project "C:\\MagicDraw\\samples\\diagrams\\class
diagram.mdzip" -output "C:\\output\\output.rtf" -template "Class
Specification Report" -package "Model" -report "Built-in" -
autoImage 1 -imageFormat png -recursive false -
outputOnBlankField "#NA"
```

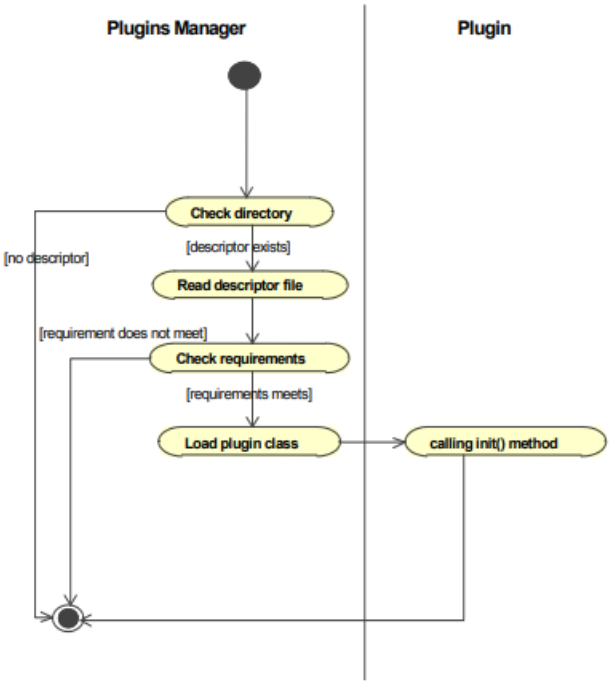
```
./generate.sh -project "/home/project/project.mdzip" -output
"/home/output/output.rtf" -template "Class Specification Report"
-package "Model" -report "Built-in" -autoImage 1 -imageFormat png
-recursive false -outputOnBlankField "#NA"
```

Фиг.8 Пример за команда за извличане на отчет в Windows и съответно в Linux

### 2.1.7 Разширимост

MagicDraw дава на разработчиците на софтуер възможността да разширят възможностите на инструмента като осигуряват Open API. Open API представлява персонализиран интерфейс, който определя взаимодействията между множество софтуерни посредници и библиотеки. Той определя видовете заявки, извикване на функции и как трябва да бъдат направени те. Осигурява механизми за разширение, така че програмистите да могат да обогатят съществуващата функционалност, без да променят ядрото на софтуера.

В контекста на MagicDraw тази функционалност се използва най-често за създаването на плъгини, които да свържат MagicDraw с други програми. Пример за това е Eclipse (IDE за Java).




Фиг.9 Схема на плъгин система в MagicDraw

## 2.2 Ограничения при използването на средата

Като всеки друг софтуерен инструмент и MagicDraw си има ограничения. Някой от тези ограничения се състоят в:

- Не е разработена функционалност за разглеждане на бизнес процеси в реално време (без нуждата от друг външен инструмент), което прави потребителите зависими от друг софтуер и неговата функционалност.
- Документацията е изключително обемна и наслагвана с годините, което прави намирането на информация в нея сравнително трудно. Преплитат се документации за различни аспекти от инструмента
- Трудности при работа на Mac
- Фокусиран е върху по-модерните езици, и не набляга много на нестандартните езици за софтуерно инженерство.
- Няма много възможности за промяна на визуални елементи.

# 3 Сравнителен анализ

Сравнителния анализ ще бъде базиран върху MagicDraw и един от най-големите му конкуренти, а именно Visual Paradigm. Ще се разгледат плюсове и минуси в двата инструмента, приликите и разликите между тях и посоките на развитие, към които трябва да се стремят двата продукта.

## 3.1 Критерии за сравнение

- Интерфейс (GUI)
- Цена за абонамент
- Функционалности
- Подобряване на работния процес
- Многослойност
- Разширяване на функционалността

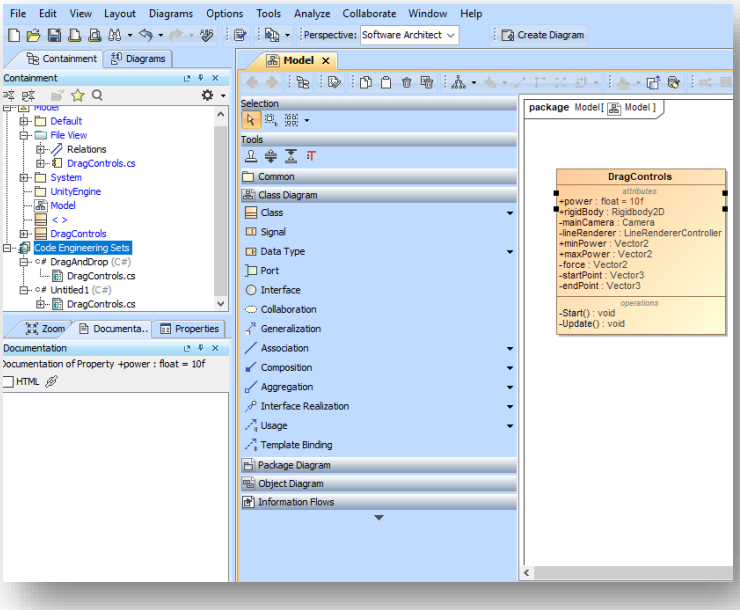
## 3.2 Сравнение в табличен вид

| **Критерий** | **MagicDraw** | **Visual Paradigm** |
|---|---|---|
| Цена за абонамент | $69 | $35 |
| API | Да | Да |
| Многослойност | Да | Да |
| Бизнес модели (реално време) | Не | Да |

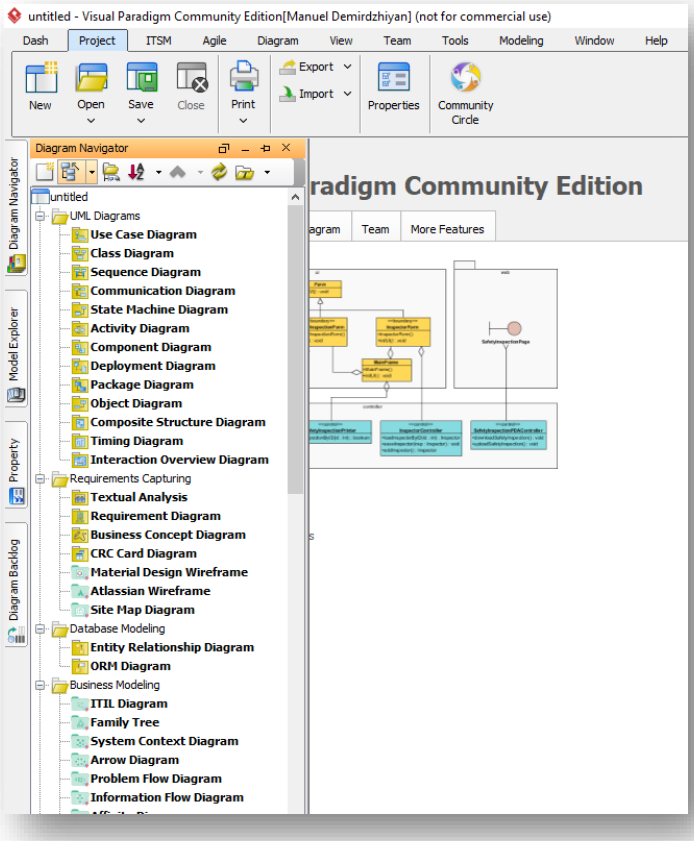
## 3.3 Сравнение на интерфейс (GUI)

Интерфейсите на двата софтуера с сравнително комплексни. Този на MagicDraw е доста по-старомоден в сравнение на този на Visual Paradigm. От друга страна обаче, интерфейса на MagicDraw е много по-интуитивен и лесен за научаване. Много по-лесно може да се започне работа, време за обучаване е сравнително по-малко. Опростяването на интерфейса на MagicDraw се дължи главно на факта, че е разработена функционалност, която отделя типа на разработчици на различни профили.

Интерфейсът на Visual Paradigm, макар и модерен е доста комплексен и претрупан. Съдържа в себе си множество от инструменти, като голяма част от тях се използват рядко, но те все пак не се скриват от потребителя.




Фиг.10 Визуализация на потребителски интерфейс на MagicDraw




Фиг.11 Визуализация на потребителски интерфейс на Visual Paradigm

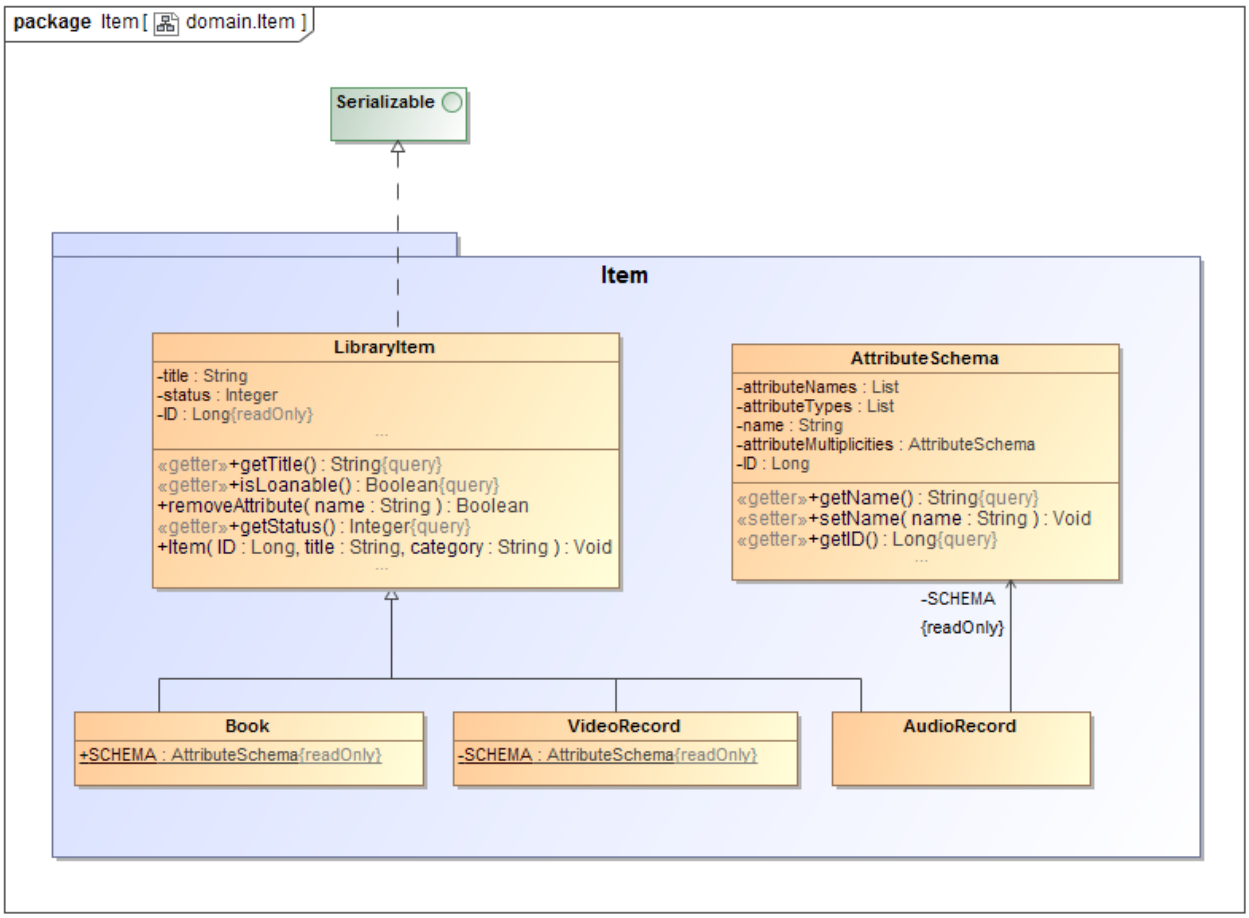
# 4 Примери на използване

Примерите за използване ще бъдат разделени на 6 групи.

- Използване от програмисти
- Използване от Анализатори и Дизайнери
- Използване от QA инженери
- Използване от Технически писател
- Използване от Ръководителя на проекта

## 4.1 Използване от програмисти

Едно от основните предимства при MagicDraw, което касае програмистите, е свойството за проверка на синтаксиса на специфичен за езика UML преди генерирането на кода. Това помага на програмиста при неволни пропуски в диаграмите, също така намалява повторимите елементи. MagicDraw поддържа езици като Java, C#, C++ и CORBA IDL, това превръща генерирането на кодови рамки в ефективен рационализиран процес. Също така, благодарение на тази система, MagicDraw позволява създаването на скриптове за всички основни бази данни и тяхното структурно проектиране.




Фиг. 11 Схема на клас диаграма ( Java )[2]

## 4.2 Използване от Анализатори и Дизайнери

MagicDraw съдържа пълна поддръжка за UML 2.5, включително Клас, Use Case, комуникация, последователност, състояние, активност, внедряване, пакет, компонент, съставна структура, диаграми за внедряване и UML метамодел. MagicDraw улеснява моделирането на всички аспекти на дадена система: изисквания, статична структура, дейности, състояния и много други.

## 4.3 Използване от QA инженери

Инструменти за моделиране на тестови казуси и автоматизация. Инженерът за осигуряване на качеството обикновено използва автоматизирани инструменти за тестване, но въпреки това той трябва да определи стратегия за структурни и функционални тестове. Use Case диаграмата е инструмент за тестово моделиране, значително подобряващ производителността на такъв вид инженер.

## 4.4 Използване от технически писател

MagicDraw елиминира досадната подготовка на документи с автоматичното генериране на отчет. Използвайки механизма за автоматично генериране на отчети на MagicDraw, техническият писател може бързо да извлече изчерпателна документация по професионални изисквания за софтуерно или друго проектиране. MagicDraw генерира стандартни артефакти, които съответстват на работния процес. С помощта Jython може лесно да се пресонализират отчетите според вътрешните фирмени стандарти. MagicDraw също така позволява да се генерират растерни изображения в JPEG и PNG формат, както и мащабируеми EOSm SVG, WMF и EMF. Като допълнение, MagicDraw позволява персонализирани шрифтове и цветове, персонализиране на външния вид на елемента на модела.

## 4.5 Използване от ръководителя на проекта

По-бързо сътрудничество на екипа. Всеки разработчик може незабавно да получи най-новата версия на модела. Всеки има право да работи паралелно от своя страна. Няма нужда да бъдат обединявани отделни XMI файлове на ръка, опростено е управлението на конфигурацията. Всички данни се съхраняват на едно място, няма голямо количество файлове, които са разпръснати из мрежата, промените могат да бъдат направени без конфликти. MagicDraw осигурява на ръководителя на проекта лесен достъп до всички артефакти. Различните разработчици могат да имат различни нива на достъп до проекта, качен на cloud системата. Взимайки под внимание ситуацията с развиващата се COVID-19 пандемия, е нужно да се допълни, че MagicDraw осигурява отдалечен достъп до проектите, нужно е само достъп до интернет и проектите са готови за работа.

# 5 Заключение и очаквано бъдещо развитие

MagicDraw е най-добрият инструмент в света на архитектурата, управлявана от модели. Той дава възможност на разработчиците, които го използват да персонализират средата си така, че да бъде удобна и опростена, без да губят време и да забавят работния процес. Интуитивния интерфейс дава възможност на новите потребители да навлязат сравнително бързо в работния процес и да започнат да градят проектите си. Благодарение на Open API системата, се дава възможност на всеки, който желае, да подобри, да разшири или да обнови функционалността на инструмента, което прави бъдещото му развитие неограничено.

Използвани литературни източници


1. https://en.wikipedia.org/wiki/MagicDraw
2. https://www.nomagic.com/products/magicdraw
3. https://www.itcentralstation.com/products/comparisons/no-magic-magicdraw_vs_visual-paradigm
4. https://www.visual-paradigm.com/
5. www.quora.com
6. “System Analysis and Design with UML” [4th Edition] – David Tegarden / Alan Denis / Barbara Haley Wixom