



Въведение в UML 2.5. Use case диаграма

Упражнение 1

Какво представлява UML

► При разработване на един софтуер, обикновено се преминава през следните няколко фази:

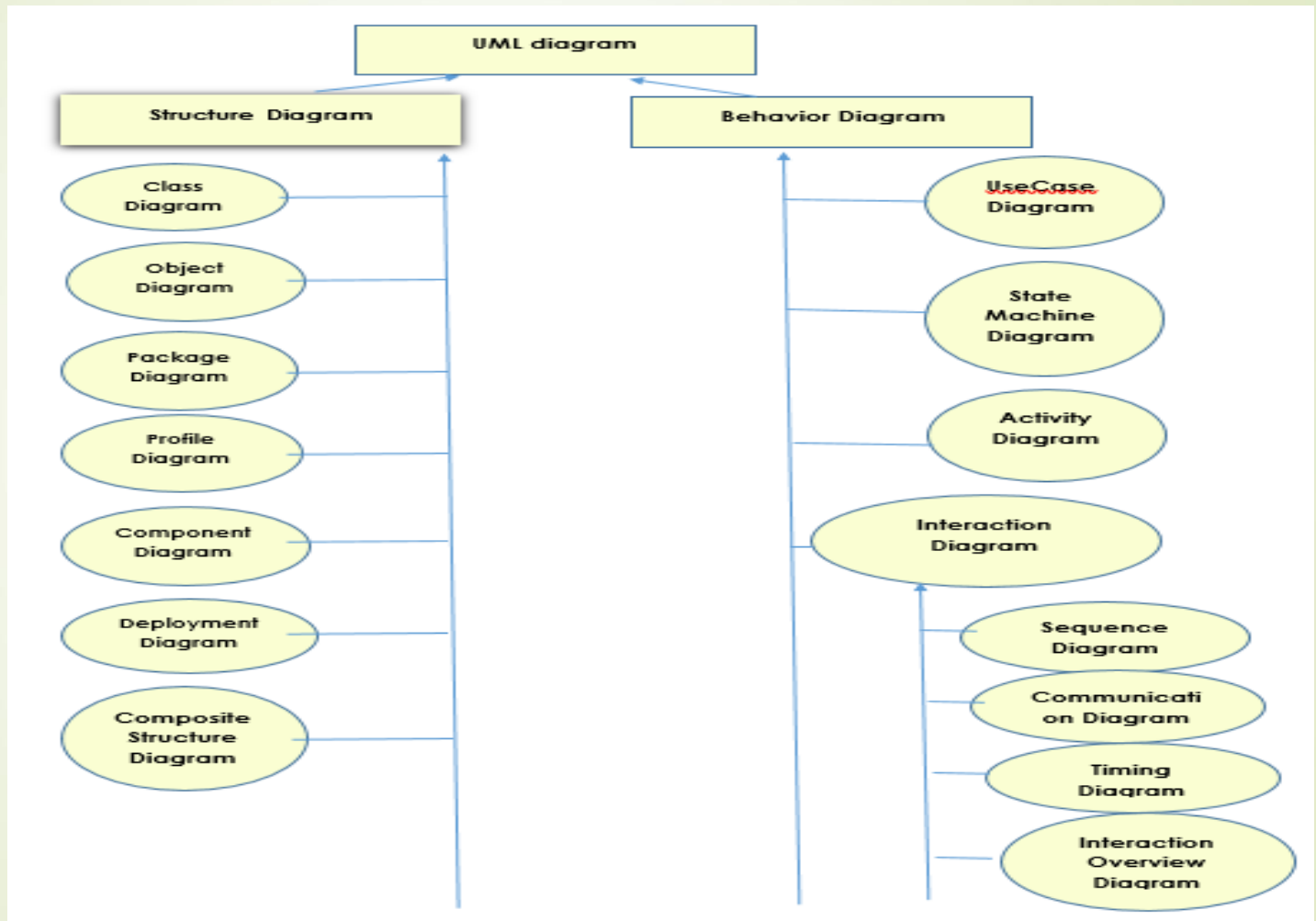
- Анализ и дефиниция
- Проектиране
- Разработване
- Тестване
- Внедряване

► UML е език за специфициране, визуализиране, конструиране и документиране на процеса за разработване на обектно-ориентирани софтуерни системи. Стандартизиран е през 1997г. от Object Management Group, като описва поведението и структурата на обектно-ориентирана софтуерна система, чрез диаграми.

UML 2.5

- **UML** спецификацията определя два основни вида UML диаграми:
 - Структурни диаграми ([structure diagrams](#))
 - Диаграми на поведението ([behavior diagrams](#))
- **Структурните диаграми** показват **статичната структура** на системата. Нейните части показват различни нива на абстракция, тяхното изпълнение и как са свързани помежду си. Елементите в структурна диаграма представляват смислените понятия на една система.
- **Диаграмите на поведение** показват **динамичното поведение** на обектите в дадена система, което може да бъде описано като поредица от промени в системата последователно във времето.

Структура в UML 2.5



<https://www.uml-diagrams.org/>



UML модел

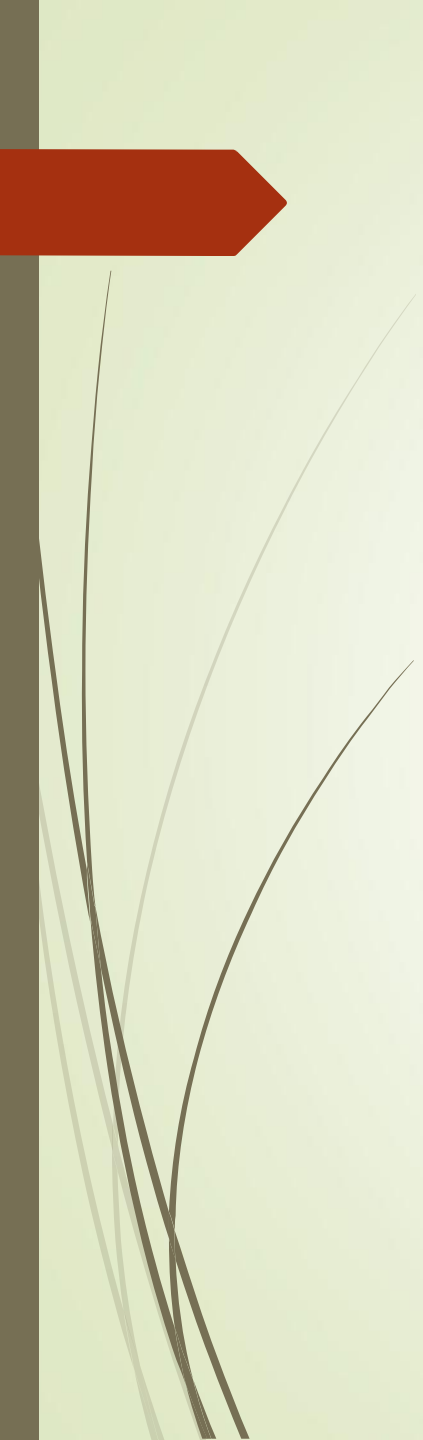
Един UML модел, представлява изглед на системата, която предстои да се разработи. Включва всички UML диаграми, описва в най-големи детайли какво включва системата и как тя ще работи.

Кой може да използва UML модела!

- **Клиентите и ръководителите на проекти** ще използват Use case диаграмите, за да получат изглед върху системата и обхвата на проекта.
- **Ръководителят на проекта** ще използва Use case диаграмите и документацията, за да раздели проекта на управляеми части.
- **Анализаторите и клиентите** ще използват Use case документацията, за да видят каква функционалност ще предоставя системата.
- **Техническият екип** ще използва Use case документацията, за да напише ръководство за потребителя.
- **Анализаторите и разработчиците** ще използват Sequence и Communication диаграмите, за да видят как ще протича логиката на системата, обектите в нея и съобщенията между тях.

Кой може да използва UML модела!

- **QA екипът** ще използва Use Case документацията, Sequence и Communication диаграмите, за да получат информация за тестовите случаи.
- **Разработчиците** ще използват Class и State Machine диаграмите, за да получат детайлен поглед върху частите от системата и как те се свързват.
- **Екипът за внедряване** ще използва Component и Deployment диаграмите, за да разбере кои компоненти къде ще бъдат разпределени в мрежата.
- **Целият екип** ще използва модела, за да бъде сигурен, че изискванията проследяват кода и че от кода могат да се възстановят изискванията.



Use case диаграма

1. Use case диаграми

Use case диаграмите се използват при извличане и спецификацията на изискванията на възможните действия в системата. Описва софтуерния продукт, който ще се разработва, на високо ниво и показва някои от use cases (случаи на употреба), някои от актьорите и връзките между тях.

Клиентът може да види каква функционалност ще предоставя системата, преди още да е започнало разработването ѝ.

Работа с use cases

Use case (Случай на употреба) е парче от функционалността, която системата ще предоставя. Има уникално име! Може да има входни и изходни условия като най-често съдържа поток от действия (процес).

В UML, use case се представят със следната нотация:

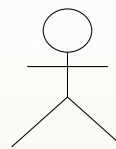


Use case моделът предоставя пълно описание на функционалността на системата, която ще се разработва.

Предимството да видите системата в use cases е, че можете да не се замисляте върху изпълнението ѝ, на този етап.

Работа с актьори

- **Актьорът** е всяко нещо, което взаимодейства със системата. Той може да бъде потребител, друга софтуерна система, външен хардуер или интервал от време.
- Актьорът е външен обект за системата, който произвежда или консумира данни.

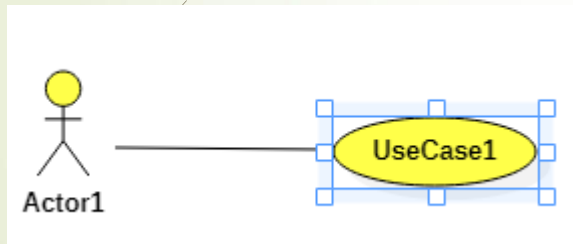


Actor1

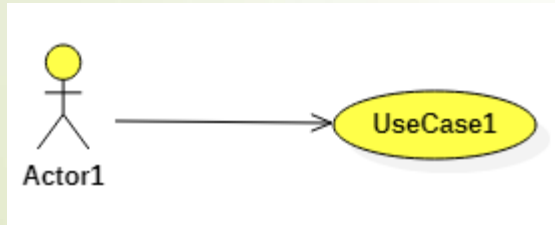
Нотация на UML за актьор

Връзки между елементите на use case диаграмите

- Връзки между актьор и use case



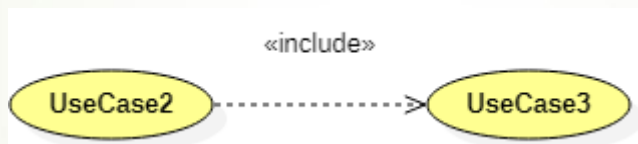
Association (Двупосочна)



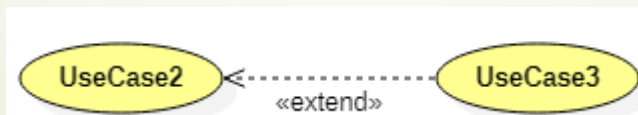
Directed association (Еднопосочна)

Връзки между елементите на use case диаграмите

- Връзки между два use cases
 - Include (Включва) – функционалността на един use case се включва във функционалност на друг use case;

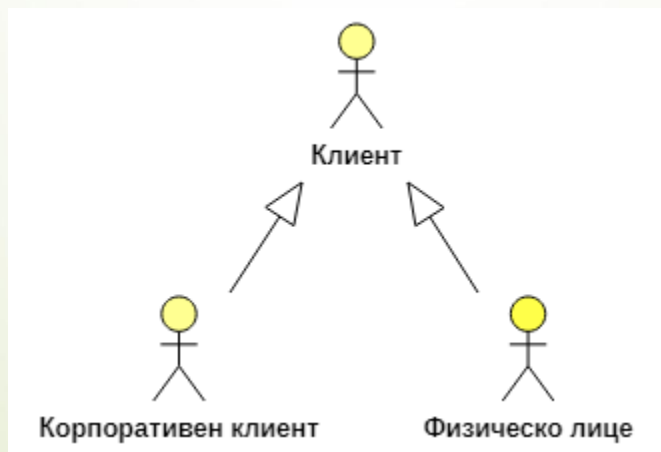


- Extends (Разширява) – функционалността на един use case се разширява с функционалност на друг use case, само при нужда.



Връзки между елементите на use case диаграмите

- **Generalization (наследяване)** – това е връзка, която по-често се използва между актьори, за да покаже, че имат връзка помежду тях.



Документация на use case

- Документацията на всеки use case включва документиране на потока от събития, което описание от своя страна е съставено от следните части:
 - Кратко описание
 - Предусловия
 - Първоначален поток от събития
 - Алтернативен поток от събития
 - Следусловия

Документация за всеки use case

Документацията на всеки use case включва документиране на потока от събития, което описание е съставено от :

- **Кратко описание** – какво ще прави даденият use case;
- **Предусловия** – условия, които трябва да се изпълнят, за да се стартира use case;
- **Първоначалния и алтернативен поток от събития включва:**
 - Как се стартира use case.
 - Различните пътища през use case.
 - Нормалния или първоначален поток през use case.
 - Някакви отклонения от първоначалния поток, познати като алтернативен поток, през use case.
 - Някакви възникнали грешки
 - Как свършва use case
- **Следусловия** – условия, които трябва да са изпълнени след завършване на use case.

Описание на use case „Теглене на пари“ от ATM

- **Първоначален поток от събития:**

1. Use case започва, когато потребителя вкара картата си в ATM.
2. ATM представя съобщение за добре дошъл и подканя потребителя да си въведе PIN.
3. Потребителя си въвежда PIN.
4. ATM потвърждава, че PIN е валиден. Ако не е валиден се изпълнява АП А1
5. ATM предоставя възможните опции:
 - Депозирание на средства
 - Теглене на пари
 - Трансфер на средства
6. Потребителят избира опцията „Теглене на пари“
7. ATM подканя потребителя да избере сумата, която иска да изтегли.
8. Потребителя избира сумата
9. ATM пита дали иска разписка
10. Потребителя прави избор.
11. ATM проверява дали в сметката има избраната сума. Ако тя е недостатъчна се стартира АП А2. Ако възникне грешка се стартира поток за грешка E1.
12. ATM приспада изтеглената сума от сметката.
13. ATM предоставя на потребителя изисканите пари.

Описание на use case „Теглене на пари“ от ATM

- ▶ 14. Ако потребителя иска бележка се стартира АП А3.
- ▶ 15. ATM връща картата на потребителя
- ▶ 16. Use Case приключва

- **Алтернативни потоци:**

- A1: Въвеждане на грешен PIN**

- 1. ATM уведомява потребителя, че въведения PIN е грешен
 - 2. ATM връща картата на потребителя
 - 3. Use case приключва

- A2: Недостатъчна наличност по сметка**

- 1. ATM уведомява потребителя, че сумата в сметката е недостатъчна.
 - 2. Връща картата
 - 3. Use case приключва

- A3: Искане на бележка**

- 1. ATM предоставя бележка на потребителя за извършената операция
 - 2. ATM връща картата
 - 3. Use case приключва

Описание на use case „Теглене на пари“ от АТМ

➤ **Поток на грешка E1:** Грешка при проверка на сумата в сметката на потребителя:

1. АТМ уведомява потребителя, че е възникнала грешка при проверка на сумата и предоставя на потребителя телефонен номер на поддържащия екип на АТМ
2. АТМ отбелязва кода на грешката в log за грешки, както и датата, времето, името на потребителя и номера на сметката
3. АТМ връща картата на потребителя
4. Use Case приключва