Обектно-ориентиран подход

за анализ и проектиране на информационни системи

ОО срещу структурен подход

- Структурен – основан на функциите на системата
- Обектно-ориентиран – основан на данните, които реализират функциите
- Предимства на ОО
  - по-точно дефиниране на изисквания
  - по-лесно разбиране на системата
  - изчистено проектиране
  - други ?

Обектно-ориентиран подход

- Обекти и отношения
- Приложения
  - ОО анализ
  - ОО проектиране
- ОО моделиране на данни и информация
  - моделиране на структурата на обектите
    - пакети, класове, атрибути
  - моделиране на структурата на поведението на обектите
    - операции и методи
  - моделиране на отношенията
  - представяне на правилата на бизнеса

Основни понятия

- Атрибути
  - “Наименована позиция в класификатор, която описва интервал от стойности, които екземплярите на класификатора могат да приемат”
  - класификатор
    - специфициране на обекти според тяхните структура и поведение
    - класове, типове, интерфейси, подсистемаи, компоненти или
    - база от данни
- Операции
  - “Услуга, която може да бъде поискана от обект като ефект на поведение”
  - подпис (име и параметри)
- Методи
  - “Приложение на операциите. Специфицира алгоритъмът или процедурата, които определят резултата от операцията”

Основни понятия

- Отношение
  - семантична връзка между елементите на модела
  - примери: асоциация и генерализация
    - Асоциация
      - семантична връзка между два или повече класификатора, включваща връзките между техните екземпляри
    - Генерализация
      - таксономично отношение между по-общ и по-специфичен елемент
      - по-специфичният елемент е изцяло съвместим с по-общия и съдържа допълнителна информация
- Интерфейс
  - декларация на колекция от операции, които могат да се използват за дефиниране на услуга, предлагана от екземпляр

ОО анализ и проектиране

- **OOA**
  - дефиниране на проблем
  - идентификация и описание на обектите, които участват в проблемната област
- **OOD**
  - дефиниране на решение
  - дефиниране на обектите за програмиране
- **OOAD**

WHAT, WHO, WHEN, HOW

Обектно-ориентиран подход

- Въпроси, които търсят отговор:
  - Какви процеси е необходимо да се поддържат от ИС?
  - Кои елементи на ИС участват в тези процеси?
  - Каква е ролята на всеки елемент на ИС?
  - Как елементите на ИС взаимодействат при реализиране на процесите?
  - ...

Обектно-ориентирано моделиране

- Специфициране
  - структурата и поведението на ИС
- Визуализация
  - как ще изглежда и работи ИС
- Подготовка
  - прототипи на бъдещата ИС
- Документиране
  - на проектни решения

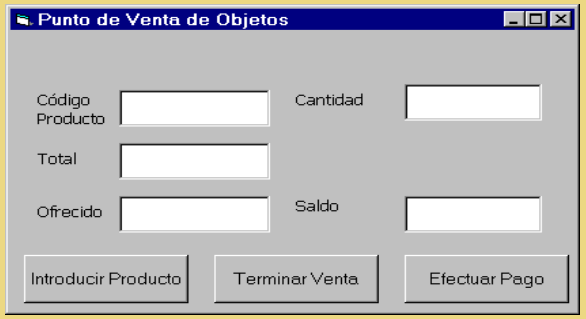
Техники за ОО моделиране

- Стъпка по стъпка
- Итерации
  - последователно приближение към планираното
  - оценка на всяка итерация
- Use Case
  - описание на приложението на системата в конкретна среда

Обекти на ОО моделиране

ГПИ

Обекти на интерфейса



Логически модел

Обекти от домейна

Обекти от дейността

Sale

Payment

Database Server

Security Administrator

Физически модел

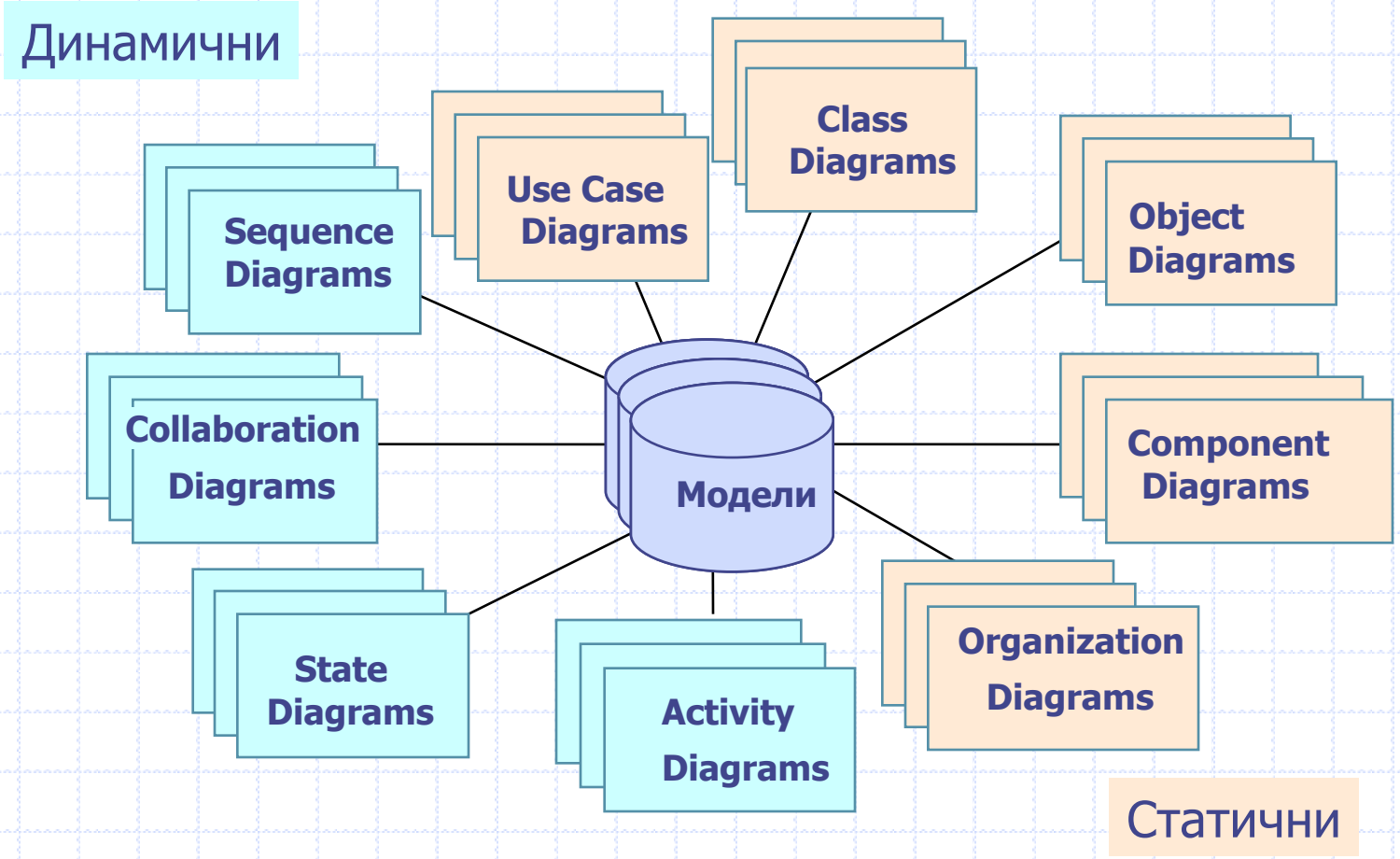
Обекти от паметта

DB

Средства за ОО моделиране

- Методологии
  - OMT
    - (Rumbauth, 1992)
  - Booch OO Analysis and Design
    - (Booch, 1994)
  - Jacobson's Objectory
    - (Jacobson, 1992)
- UML (Unified Modeling Language)
  - OMG (Object Modeling Group)
    - приема UML като стандартна нотация за ОО методи

Видове ОО модели



Основни сведения за UML

- UML
  - език за моделиране
  - използва графични изразни средства
- Дефинира нотация за описание на модели и мета-модел
- Видове диаграми
  - use case
  - activity
  - sequence
  - class
  - моля, допишете ...

Основни концепции

- Използвани конструкции
  - елементи
  - отношения – свързват елементите
  - диаграми - свързват елементите в групи със специално значение
- Категории елементи
  - Basic
    - Основни: Use Cases, Classes
  - Behaviour
    - За моделиране на поведение: Interaction
  - Grouping
    - За групиране на елементи: Packages
  - Anotation
    - За описание: Narratives, Notes

Основни елементи

- Use Cases
  - описват поведението на системата
  - описват множество действия на системата, които водят до получаване на видим и важен, за потребителя, резултат
  - създават се в сътрудничество с потребителите
- Показват
  - транзакциите в системата
  - използване на данните, прилагане на ограниченията
  - ролята и значението на БД в цялата система
- Използват се като
  - основа за създаване на модели на данни и операции
  - средство за контрол и оценка на софтуер

Елементи на Use Case

- Основни данни
  - име, номер, кратко описание
  - тригери – събития, които го предизвикват
    - външни
    - времеви, периодични
- Вход и изход
  - източници и консуматори
  - цели
- Детайли
  - стъпки
  - данни

Създаване на Use Cases

- идентифициране на use case
- идентифициране на основните стъпки за реализиране на всеки use case
- идентифициране на елементите / данните за всеки use case
- потвърждаване
- следваща итерация

1: Идентифициране на use case

Дейности	Типични въпроси
Попълване на формуляр Ако има повече от 9 use case – групират се в пакети	Ask **who**, **what**, and **where** about the tasks and their inputs and outputs: What are the major tasks performed? What triggers this task? What tells you to perform this task? What information/forms/reports do you need to perform this task? Who gives you these information/forms/reports? What information/forms/reports does this produce and where do they go?

2: Идентифициране на основните стъпки за всеки use case

Дейности	Типични въпроси
Алгоритъм за преобразуване на входните данни в изходни	Ask **how** about each use case: How do you produce this report? How do you change the information on the report? How do you process forms? What tools do you use to do this step (e.g., on paper, by email, by phone)?

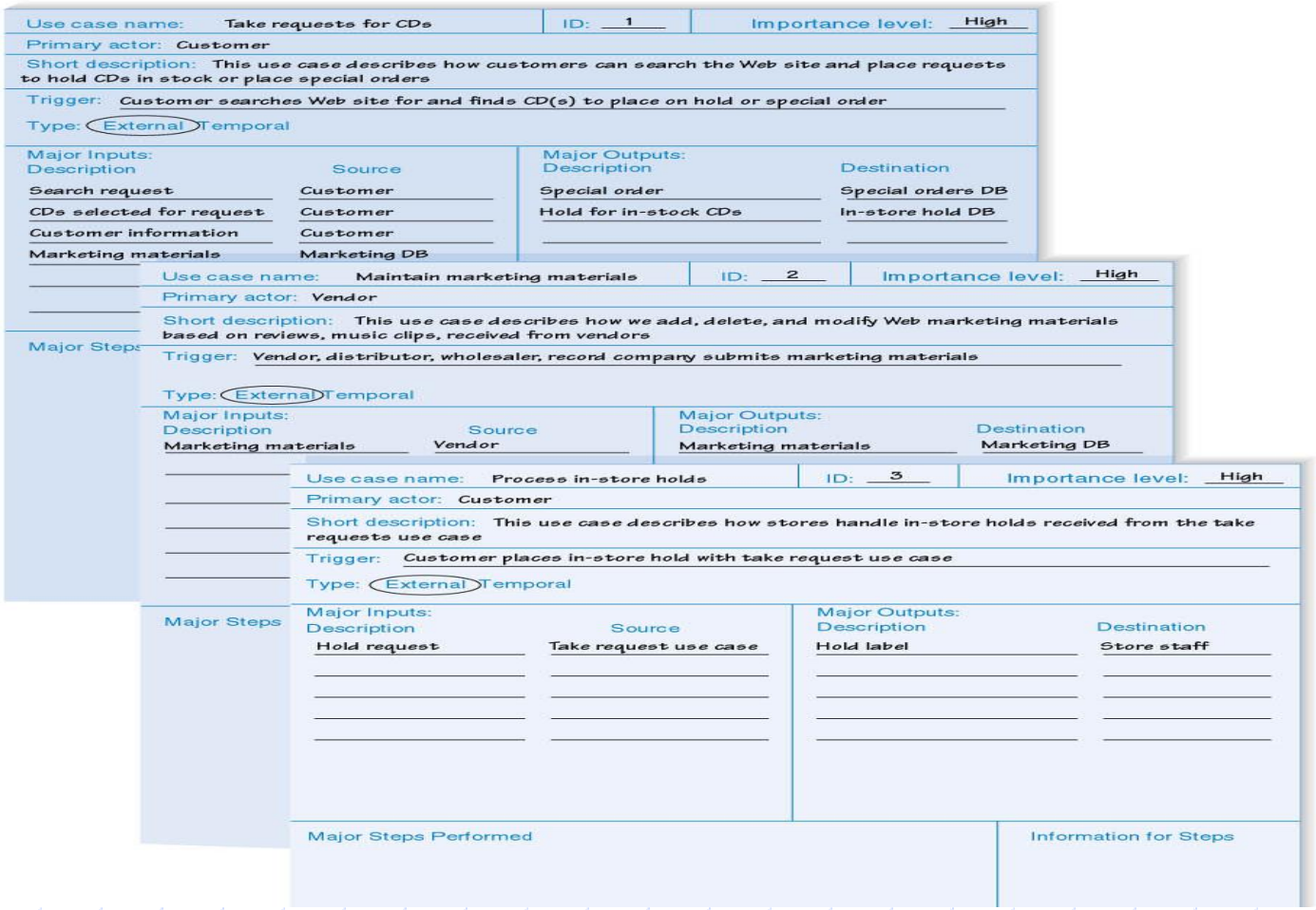
3: Идентифициране на елементите на стъпките

Дейности	Типични въпроси
тригери, вход и изход	Ask **how** about each step How does the person know when to perform this step? What forms/reports/data does this step produce? What forms/reports/data does this step need? What happens when this form/report/data is not available?

4: Потвърждение

Дейности	Типични въпроси
•Валидиране	Ask the user to execute the process using the written steps in the use case – that is, have the user role-play the use case

Примери за формуляри



Use case name: Take requests for CDs | ID: 1 | Importance level: High
Primary actor: Customer
Short description: This use case describes how customers can search the Web site and place requests to hold CDs in stock or place special orders
Trigger: Customer searches Web site for and finds CD(s) to place on hold or special order
Type: (External) Temporal

Major Inputs: Description	Source	Major Outputs: Description	Destination
Search request	Customer	Special order	Special orders DB
CDs selected for request	Customer	Hold for in-stock CDs	In-store hold DB
Customer information	Customer		
Marketing materials	Marketing DB		

Use case name: Maintain marketing materials | ID: 2 | Importance level: High
Primary actor: Vendor
Short description: This use case describes how we add, delete, and modify Web marketing materials based on reviews, music clips, received from vendors
Trigger: Vendor, distributor, wholesaler, record company submits marketing materials
Type: (External) Temporal

Major Inputs: Description	Source	Major Outputs: Description	Destination
Marketing materials	Vendor	Marketing materials	Marketing DB

Use case name: Process in-store holds | ID: 3 | Importance level: High
Primary actor: Customer
Short description: This use case describes how stores handle in-store holds received from the take requests use case
Trigger: Customer places in-store hold with take request use case
Type: (External) Temporal

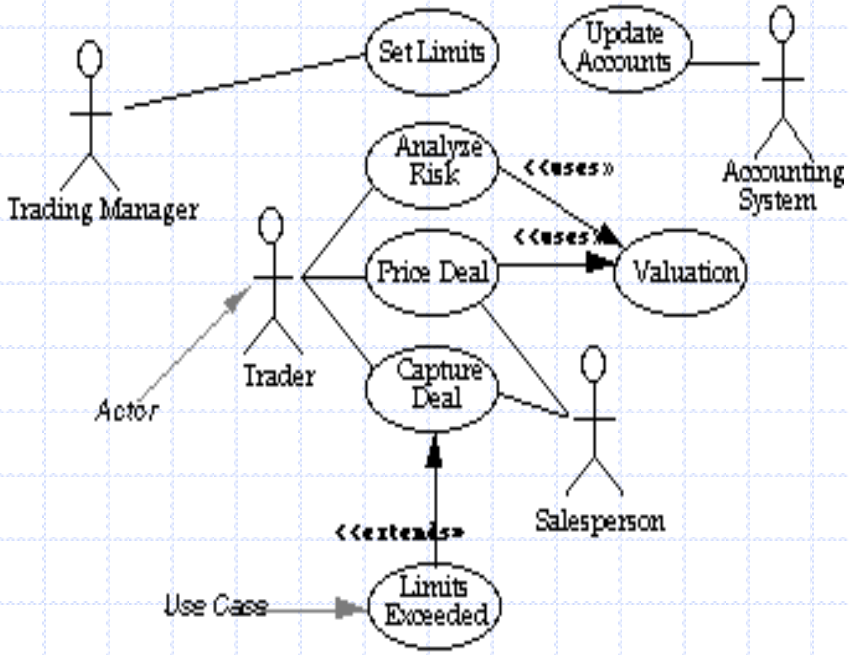
Major Inputs: Description	Source	Major Outputs: Description	Destination
Hold request	Take request use case	Hold label	Store staff

Major Steps Performed | Information for Steps

Use Case диаграма

- Елементи
  - Actor
    - Роля на външен за системата обект, който взаимодейства с нея
    - Един физически обект може да играе няколко роли – всяка роля се моделира отделно
  - Use Case
    - Транзакция – логическа единица работа на актьор
    - За всеки актьор се определя кои транзакции инициира
- Подобно на сценарий, Use Case диаграмите описват какво трябва да се прави в системата

Use Case диаграма



Actor

- Кой или какво ще използва коя част от системата
  - определени актьори имат нужда от достъп до определени данни по определен начин
- Технология
  - Определят се границите на системата
  - Прави се списък на актьорите
    - Може да бъде всеки
      - потребител
      - програма
      - устройство
    - Може да бъде с йерархична структура
  - Определя се итеративно

Use Case

- Use cases – в границите на системата
  - Транзакции от високо ниво
- При определянето им се прилага се методът top down

- Асоциация
  - връзката между актьор и use case

Асоциации

- extends
  - условна част от транзакция, не цяла транзакция
  - ако условието е изпълнено – разширение на транзакцията
  - напр. избор от меню
- uses
  - use case, използван от два или повече други use cases
  - включва се в друг use case
  - относително самостоятелна част от системата

Използване на Use Case диаграма

- Разширения на значението, посредством
  - Summaries – кратко текстово описание, коментар относно предназначението
  - Narratives- подробно текстово описание на детайлните стъпки в use case: 1…., 2 …., …
  - Activity Diagram – описание на алгоритъма
  - Data Elements –данни, необходими за този use case
  - Business Rules –ограничения върху данните и операциите в този use case

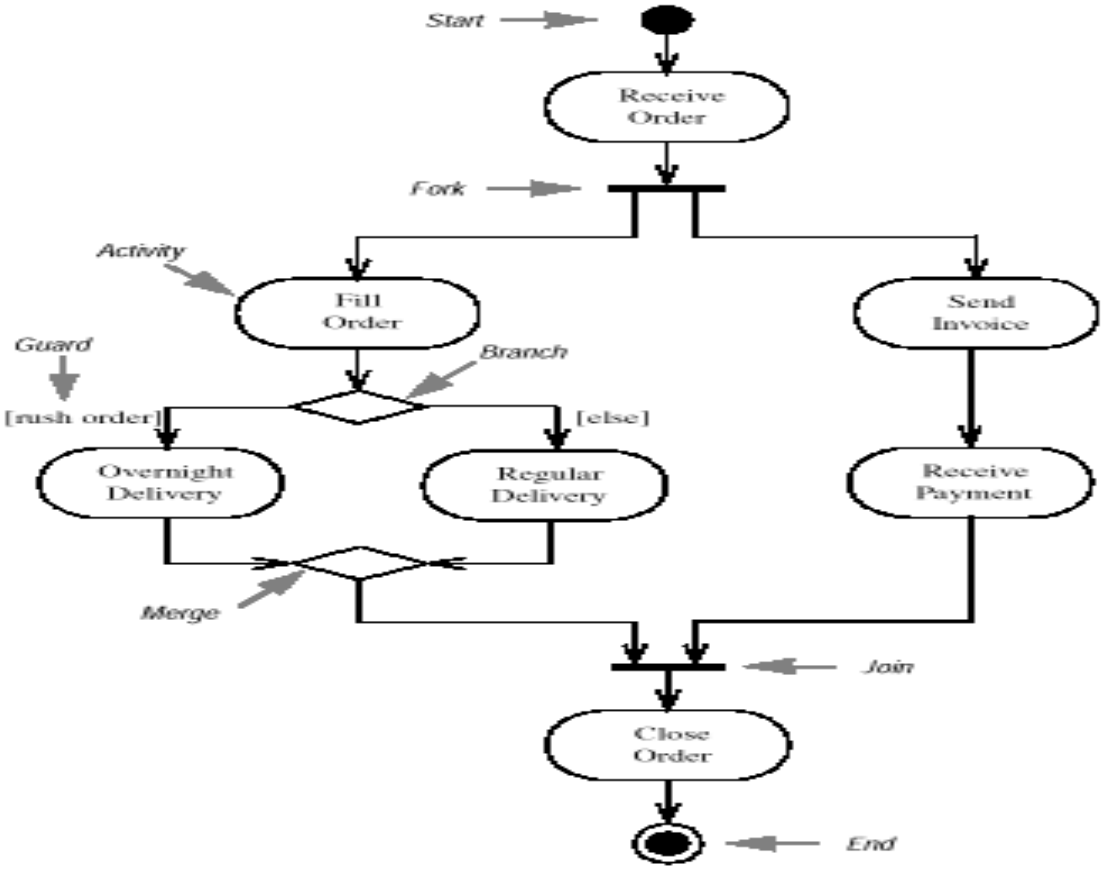
Кой използва Use Case

- Потребителят
  - за по-добро разбиране и анализ на средата
- Работният колектив
  - за взаимно разбиране
  - начин на комуникация
- Клиентът
  - проверка на очакванията
  - взаимно разбиране между клиента и автора
- Организацията
  - който тества и оценява системата

UML Activity диаграма

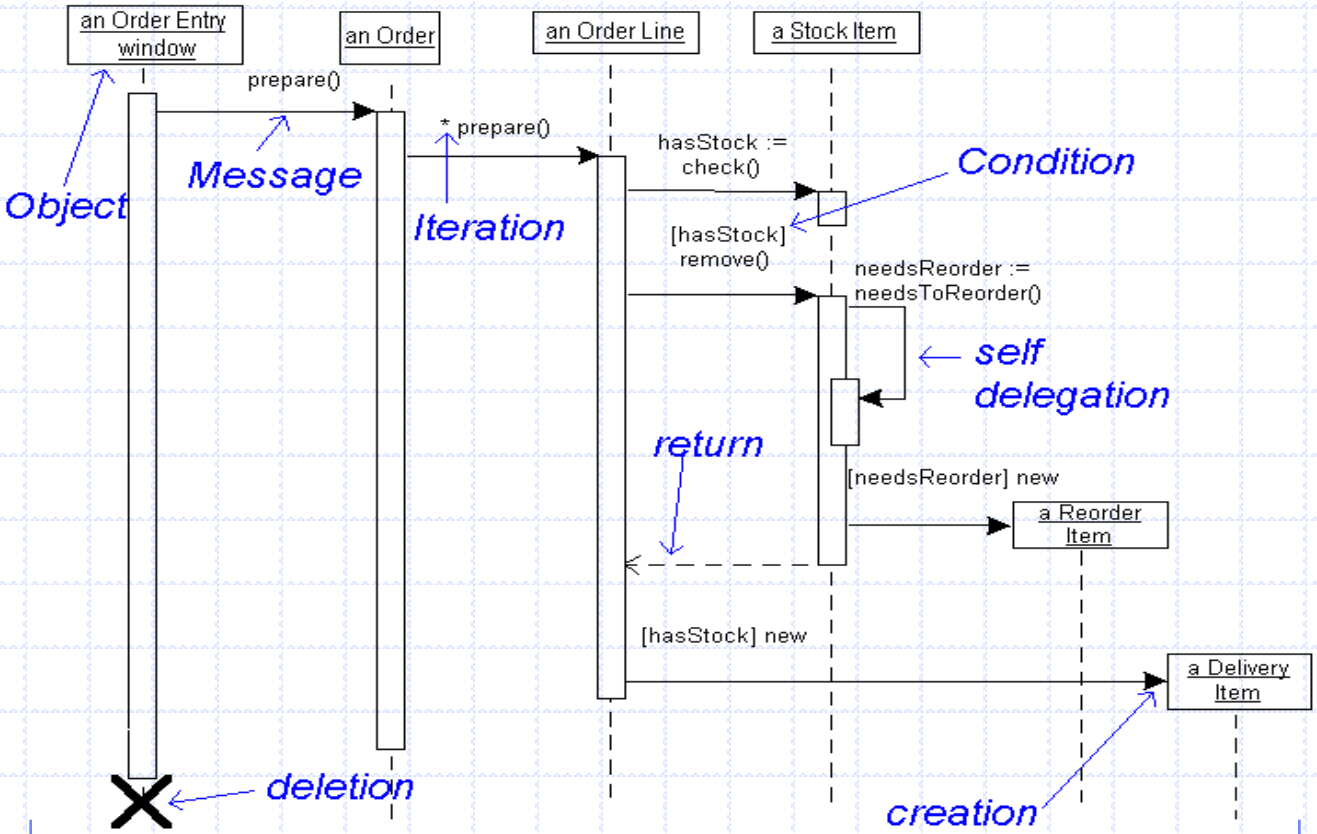
- Управляваща логика на Use Case
- Управляваща структура
  - последователна
  - условна (алтернативна)
- Елементи
  - Action State (стъпки)
  - Guard Condition (условия)

UML Activity диаграма



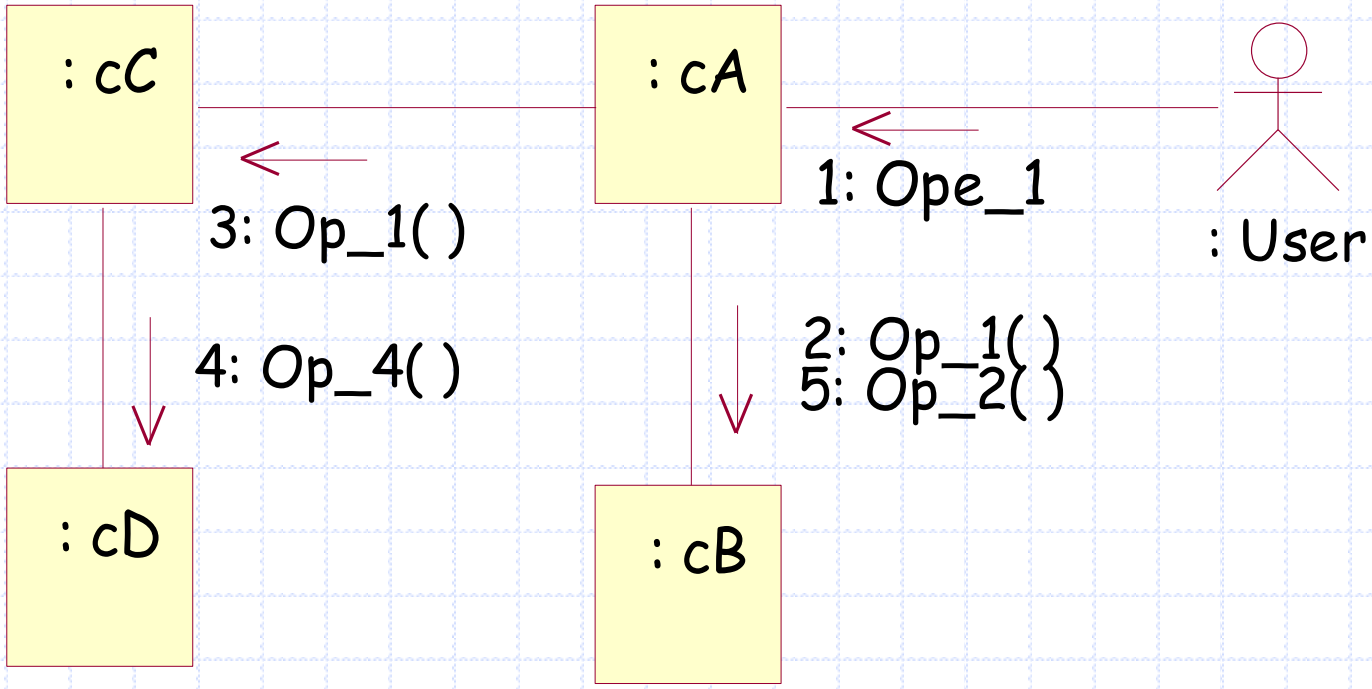
UML Sequence диаграма

- Дефинира връзките между обектите и тяхната последователност във времето



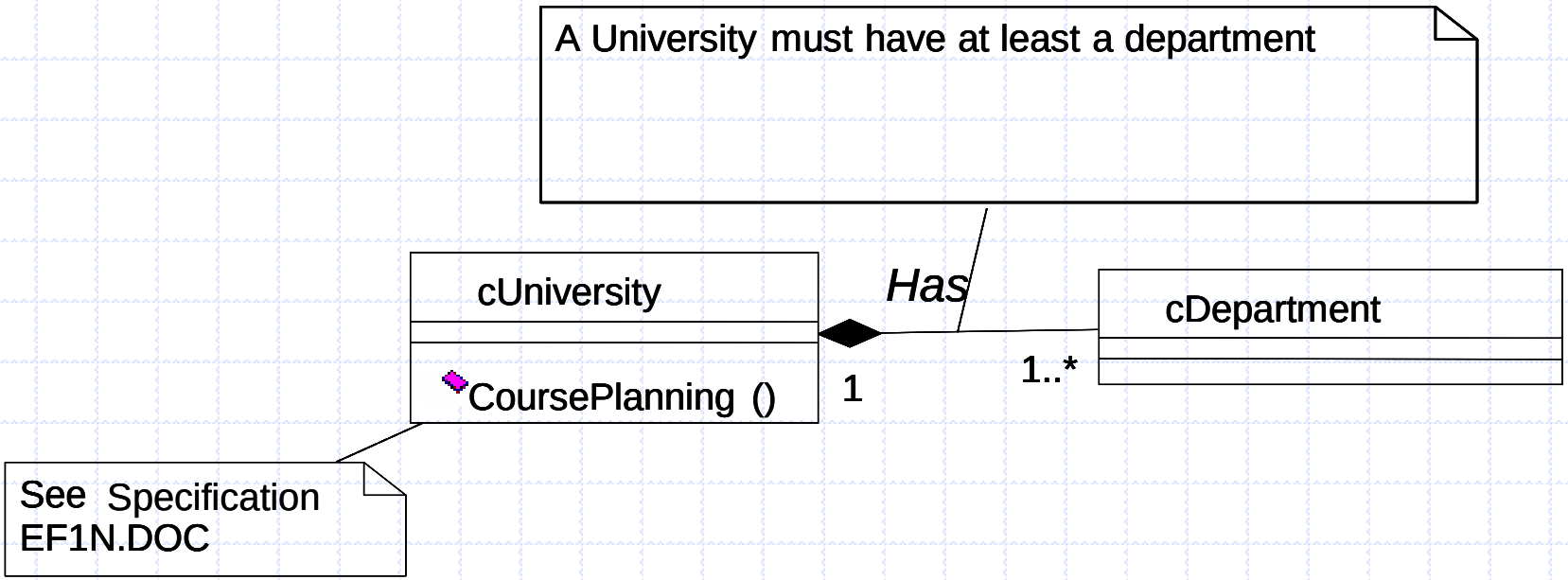
UML Collaboration

- Дефинира взаимодействието между обектите за реализиране на дадена операция



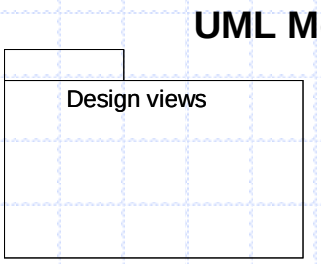
UML Notes

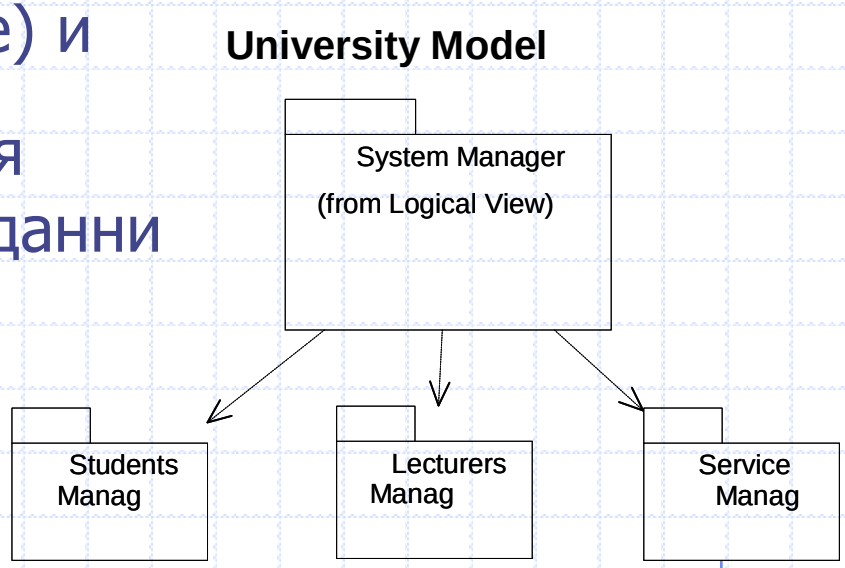
- Анотиране на обекти или отношения



UML Packets

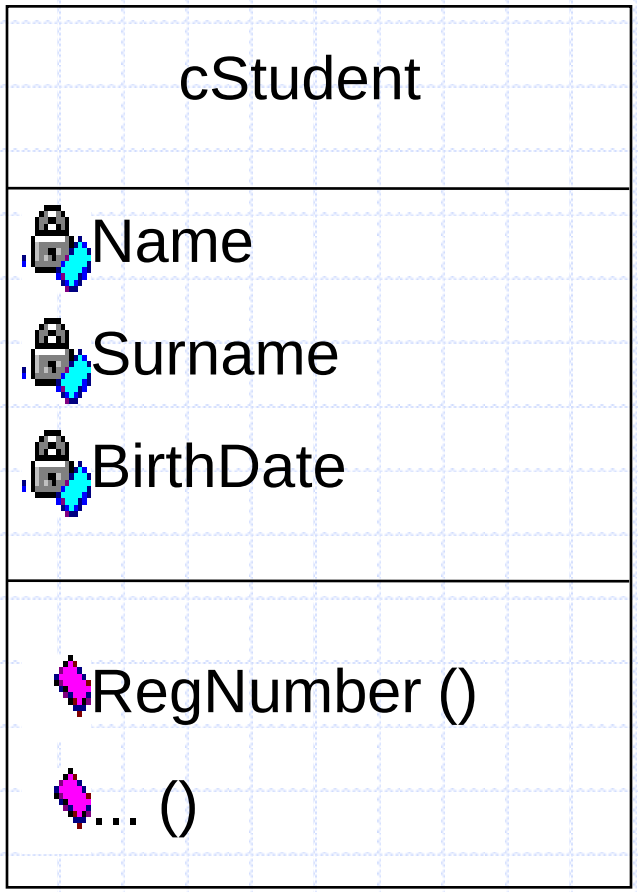
- Групиране – дефиниране на модули на модела
- Група от елементи със сходно значение, които често се променят едновременно
- Подсистеми, reusable components
- Спецификация (use case) и реализация (class) на множество от поведения
- Различни схеми и бази данни
- Name spaces
- Колаборация на подсистемите



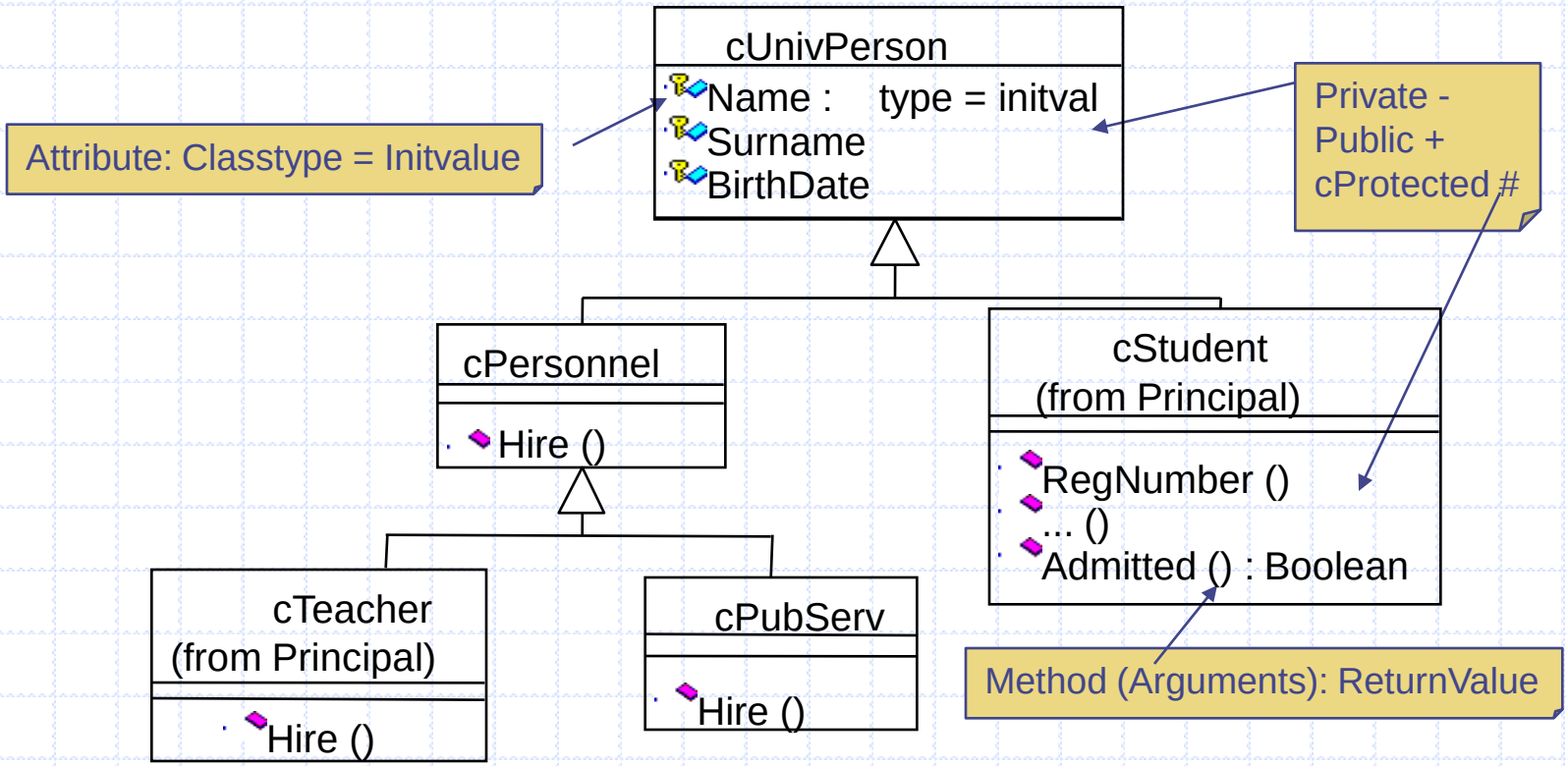


UML Class диаграма

- Аналог на Entity в ER
- Моделиране на структурата на системата
- Описание на множество от обекти, които споделят общи
  - атрибути
  - операции
  - методи
  - отношения
  - семантика
- Може да използва множество от интерфейси за специфициране на групи операции



UML Class диаграма



UML Class диаграма

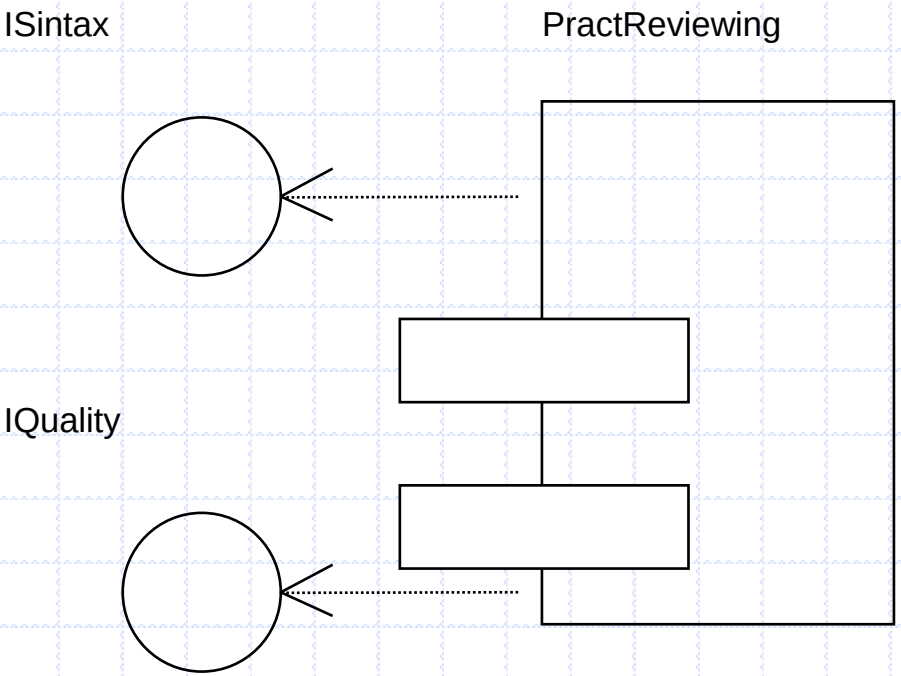
- Може да има класове без атрибути, но не без операции
- Атрибути
  - на екземплярите
    - всеки екземпляр има собствени стойности на атрибута
  - на класа
    - една обща стойност, споделяна от всички екземпляри на класа
- Видимост на атрибутите
  - public + всеки друг клас може да оперира с тях
  - private – само методите на класа
  - protected # само методите на този клас, или на под-клас

Абстрактни класове

- Абстрактни класове
  - обща абстракция, споделена от няколко под-класа
  - класове в отношение генерализация-специализация
  - нямат екземпляри

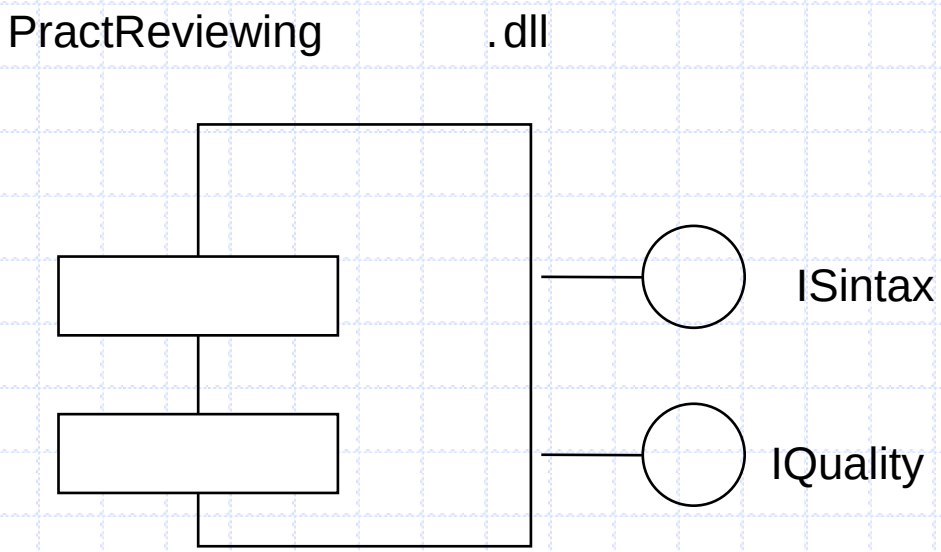
UML Interface

- Абстрактно множество от операциите на клас или компонент
- Колекция от операции, използвани за специфициране на ролята на клас или компонент, но и
- независим обект, който може да има и собствени атрибути



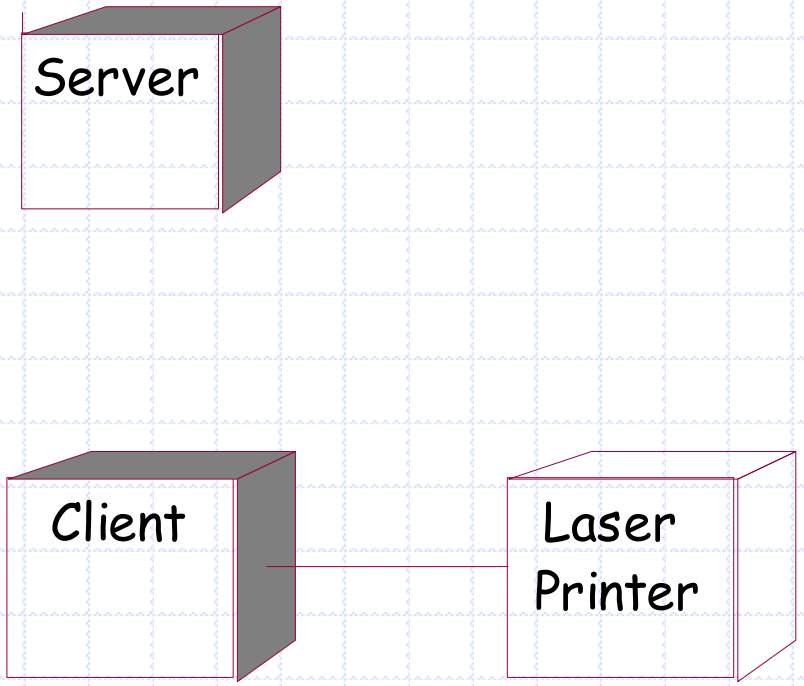
UML Component

- Физическо групиране на класовете, интерфейсите и взаимодействията в компоненти с ясно и завършено значение

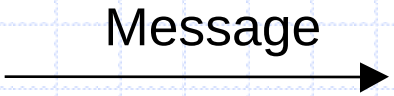


UML Nodes

- Модел на физически елементи
- Представляват ресурси



Messages



- Елементи, които дефинират взаимодействието между обектите
- Предават информация от един обект (sender) към друг (receiver)
- Предаваните съобщения имат определени контекст и цел
- 5 типа съобщения
  - create
  - destroy
  - synchronous call (изпращане на съобщение; чакане на отговор)
  - asynchronous call (изпращане на съобщение без блокиране за чакане на отговор)
  - asynchronous signal (изпращане на информационно съобщение; не се предполага отговор, напр. за грешка)

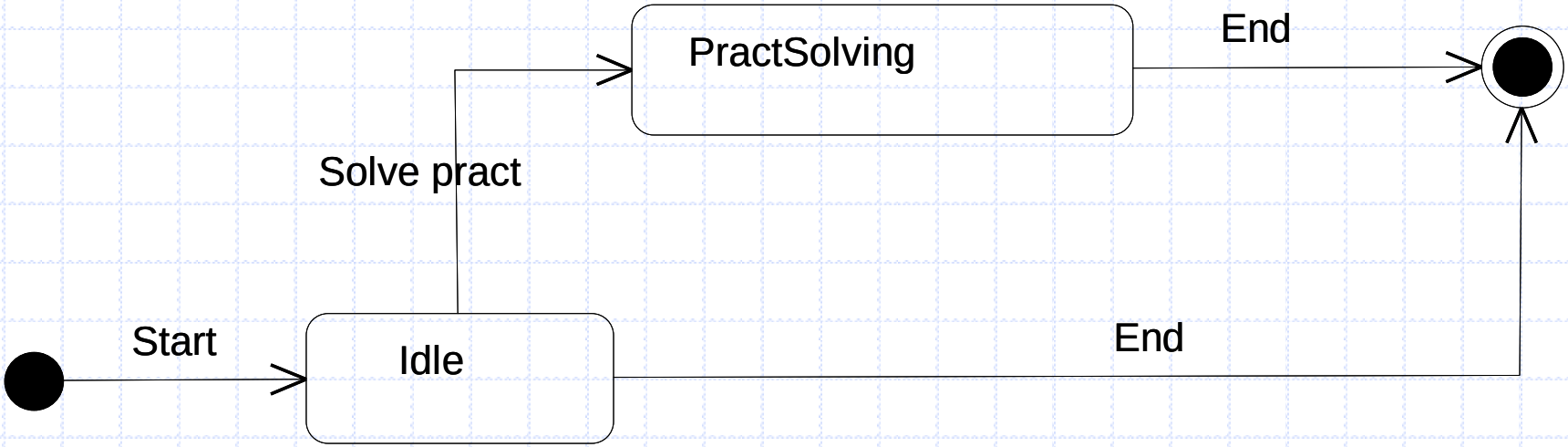
UML Interactions

- Последователност от съобщения между обектите с определен контекст и цел
- Interaction

  обекти + съобщения + определена последователност

State Machines

- Описание на състоянията, до които обектът може да достигне

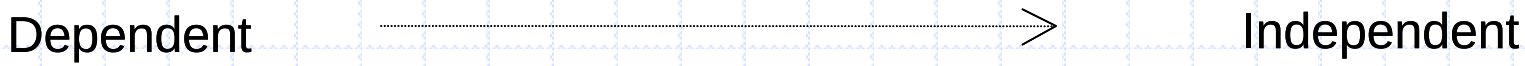


UML отношения

- Зависимост
- Асоциация
- Генерализация
- Реализация

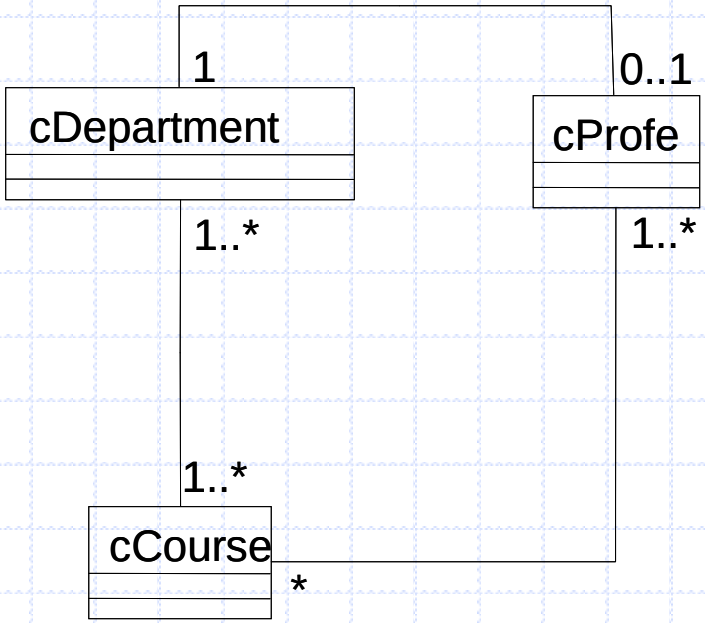
Зависимост

- Промяна в независимия елемент се отразява на зависимия
- Представя отношение Uses



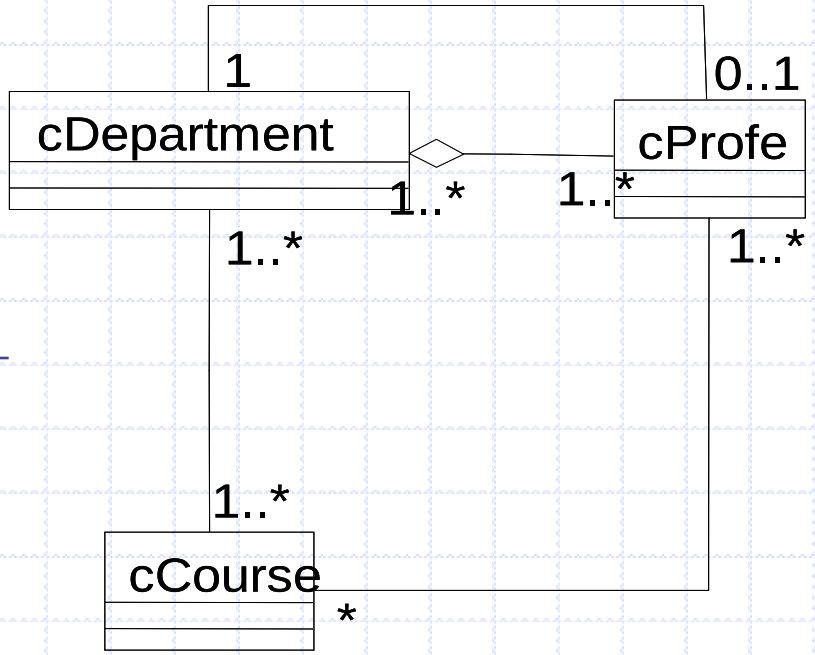
Асоциация

- Множество от връзки между обекти с еднакво ниво на абстракция
- Връзка между елементи
- Множественост
  - 0..* : нула или повече
  - 0..1 : не повече от един
  - 1..* : поне един обект
  - 1 : точно един
  - * : нула или повече
  - 2..6 : поне 2, но не повече от 6
  - 1,3,5-7 :



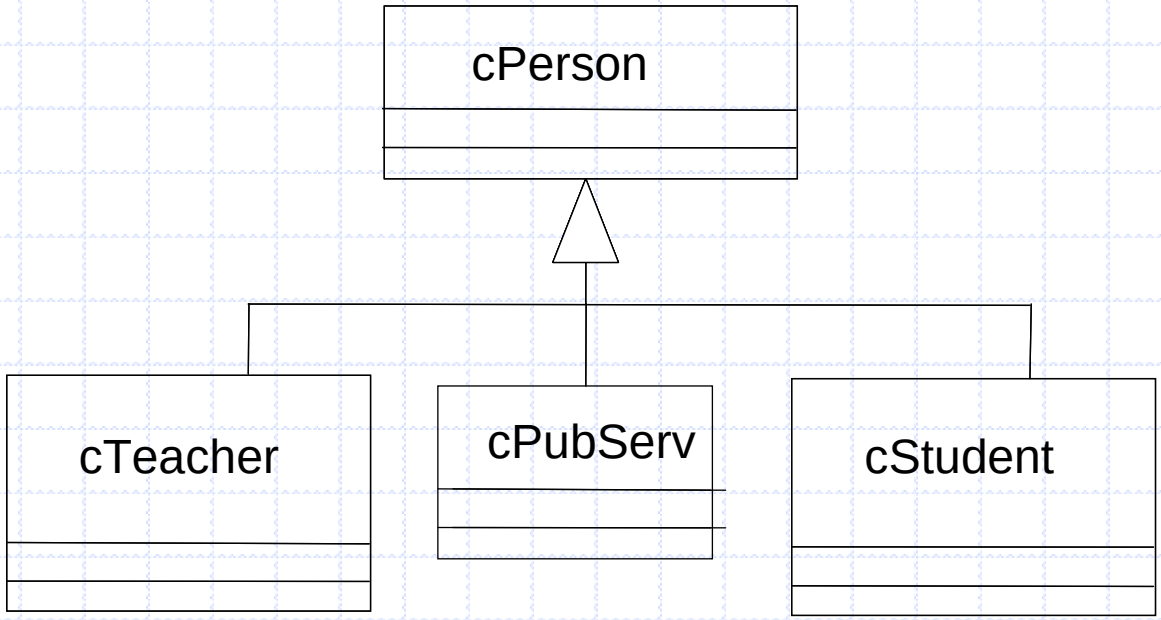
Асоциация

- Aggregation
  - вид асоциация
  - представя структурно отношение между цяло и частите му
- Видове
  - слаба (споделена) -
  - силна (композиция) - единият обект изцяло притежава другия
  - насочени / ненасочени
  - наименовани
  - множествени
- Роли



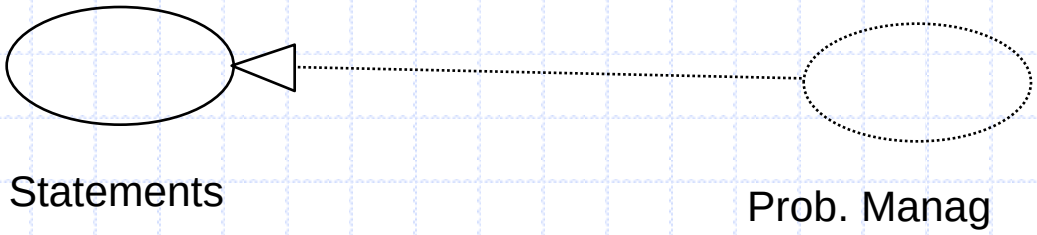
Генерализация

- Представя наследственост
- Супер-клас и под-клас
- Is-a



Реализация

- Семантично отношение
  - специализация
- Използва се за
  - Интерфейси
  - Use cases и Collaborations



UML Constrains

- Ограничения и бизнес правила
- Асоциации – referential integrity
- Нотация: { }
- PK {OID}
  - идентичност и уникалност на обектите
- Множество на допустимите стойности (domains) `[set], <enum>`
- Комплексни ограничения {OCL}
  - Object Constrain Language

Недостатъци на ООМ и UML

- Слаба връзка между спецификация и реализация
- UML
  - труден за прилагане
  - прекалено универсален, не различава типовете приложения
- Други ?