# Анализ систем

## Анализ систем

### System analysis

1. Системный анализ - совокупность методов для представления некоторой системы в виде модели или схемы.
2. Анализ в свою очередь это разложение системы на такую модель.

#### Structured analysis

1. Represents the system in terms of data and the processes that act upon that data. System development is organized into phases with durables and milestones to measure progress. The waterfall model typically consists of five phases: testing and maintenance & evolution. Iteration is possible among the phases.
2. structured analysis uses DFDs to model data and processes

##### Structured analysis model

1. Конечным результатом структурного анализа является создание модели системы.
2. В структурном анализе моделью являются совокупность:

* источников данных
* data flows
* процессы

1. Эта модель выражается в DFD диаграмме, содержащей все эти элементы.

Но существуют и другие диаграммы для построения модели структурного анализа.

###### DFD

1. file:///D:/studwork/2%20%D0%BF%D0%BE%D1%81%D1%82%D1%83%D0%BF%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5/%D0%BE%D0%BB%D0%B8%D0%BC%D0%BF%D0%B8%D0%B4%D1%8B%20%D0%B8%20%D1%8D%D0%BA%D0%B7%D1%8B/%D0%98%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D0%B5%20%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B%20%D0%B8%20%D1%82%D0%B5%D1%85%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D0%B8.%20%D0%A0%D0%B0%D1%81%D0%BF%D1%80%D0%B5%D0%B4%D0%B5%D0%BB%D0%B5%D0%BD%D0%BD%D1%8B%D0%B5%20%D0%B8%D0%BD%D1%82%D0%B5%D0%BB%D0%BB%D0%B5%D0%BA%D1%82%D1%83%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5%20%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D0%B5%20%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B/%D1%80%D0%B0%D0%B7%D0%B4%D0%B5%D0%BB%209.%20%D0%90%D1%80%D1%85%D0%B8%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D1%8B%20%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D1%85%20%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC/systems-analysis-and-design\_compress.pdf
2. DFD диаграмма состоит из компонентов:

* Entity (источник данных)
* data flow
* process ( программы которые трансформируют данные)
* хранилища данных

1. Эти компоненты лишь абстракции созданные для упрощенной визуализации

НО те физические вещи, стоящими за этими компонентами описаны в **словаре данных**.

Что представляют из себя источники данных (клиенты, банк итд), какая программа осуществляет тот или иной процесс - всё это в словаре данных.

###### ER-diagram

Конструктивные элементы

В ER-диаграммах (Entity-Relationship) основные конструктивные элементы описывают структуру данных и связи между ними.

**1️⃣ Сущности (Entities)**

* Представляют объекты реального мира, о которых нужно хранить информацию.
* Обозначаются **прямоугольниками**.
* Примеры: Student, Course, Order.

**2️⃣ Атрибуты (Attributes)**

* Свойства или характеристики сущностей.
* Обозначаются **эллипсами**, соединёнными с сущностью линией.
* Типы атрибутов:
  + **Простые (Simple)** — неделимые, например name.
  + **Составные (Composite)** — могут быть разбиты на части, например FullName = FirstName + LastName.
  + **Многозначные (Multivalued)** — могут иметь несколько значений, например PhoneNumbers.
  + **Ключевые (Key/Primary Key)** — уникально идентифицируют сущность, например StudentID.

**3️⃣ Связи (Relationships)**

* Показывают, как сущности связаны между собой.
* Обозначаются **ромбами**, соединяющими прямоугольники сущностей.
* Связи бывают:
  + **Один к одному (1:1)**
  + **Один ко многим (1:N)**
  + **Многие ко многим (M:N)**

**4️⃣ Атрибуты связей**

* Свойства самой связи.
* Например, связь Enrollment между Student и Course может иметь атрибут Grade.
* Обозначаются **эллипсом**, соединённым с ромбом связи.

**5️⃣ Общие элементы**

* **Подтип / Супертип (Generalization / Specialization)** — отношения наследования между сущностями.
* **Объединение (Aggregation)** — когда связь рассматривается как сущность для других связей.

#### Object-oriented analysis

1. Рассматривает систему с точки зрения объектов, объединяющих данные и процессы. Объекты представляют реальных людей, вещи, транзакции и события. По сравнению со структурным анализом, фазы 0-0 как правило, более интерактивны. Можно использовать waterfall модель или модель, подчеркивающую большую итеративность.

##### Object Modeling

1. The end product of 0-0 analysis is an **object model**, which represents the information system in terms of objects and 0-0 concepts.
2. systems analysts use UML to describe 0-0 systems (Object-oriented systems).

###### UML

1. UML Язык который представляет систему как набор объектов, которые принадлежат некоторым классам и у которых есть методы. А также бизнес логику всех существующих объектов. Механика такая же как у языков с ООП.   
     
   Объекты бывают двух типов

* актор (пациент)
* use case (пройти обследование).

## UML

### Ассоциация

В **UML (Unified Modeling Language)** **ассоциация** — это **связь между двумя (или более) классами**, которая показывает, что **один объект "знает" о другом и может с ним взаимодействовать**.

#### 🧩 Определение:

**Ассоциация (association)** — это **структурная связь**, указывающая на **отношения между объектами классов**, обычно через **ссылки (references)**.

#### 📌 Пример:

Если есть два класса:

Класс: Автомобиль

Класс: Двигатель

И в UML между ними нарисована **сплошная линия** — это **ассоциация**:

[Автомобиль] ────────── [Двигатель]

Это означает: **автомобиль имеет двигатель** (и знает о нём, может обращаться к нему).

#### 🔢 Мультиплицирование (множественность):

Ассоциация может сопровождаться **числовыми значениями** (мультиплицированием), например:

[Компания] 1 ───── \* [Сотрудник]

Означает:

* Компания имеет **много сотрудников**,
* Каждый сотрудник принадлежит **одной компании**.

#### 📘 Типы ассоциаций:

|  |  |  |
| --- | --- | --- |
| **Тип** | **Обозначение** | **Значение** |
| **Ассоциация** | Сплошная линия | Объекты знают друг о друге |
| **Агрегация** | Линия с **пустым ромбом** | "Целое–часть", слабая связь (например, класс и ученики) |
| **Композиция** | Линия с **закрашенным ромбом** | "Жёсткое целое–часть", сильная связь, часть не может существовать без целого (например, комната и дом) |

#### 🧠 В коде:

Ассоциация в UML часто отражается в виде **поля в классе**:

class Car:

def \_\_init\_\_(self, engine):

self.engine = engine # ассоциация с объектом Engine