Лекция 3: Моделирование, UML

Юрий Литвинов
yurii.litvinov@gmail.com

05.10.2017г

Моделирование

- **Модель** — упрощённое подобие объекта или явления
- Нужны для изучения некоторых их свойств, абстрагируясь от сложности “настоящего” объекта или явления
- Модели используются повсеместно
  - Математические модели
  - Модели как реальные объекты
  - Модели в разработке ПО

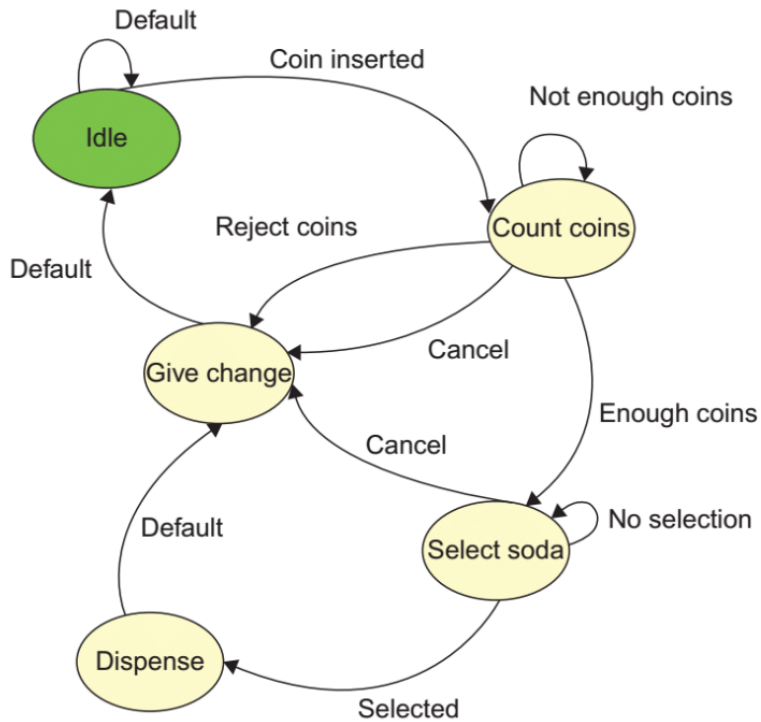
Общие свойства моделей

- Содержат меньше информации, чем реальность
- Существуют для определённой цели
- Модели субъективны, что позволяет отделить существенные свойства от несущественных
- Модели ограничены

All models are wrong, some are useful

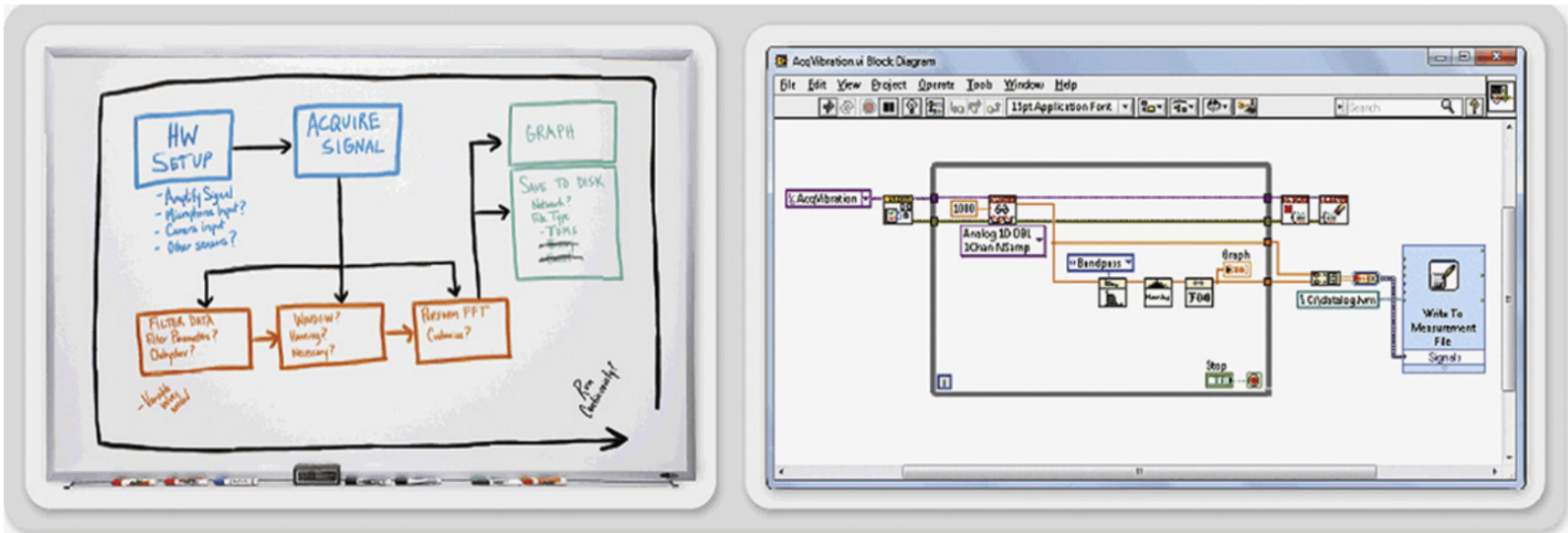
Моделирование ПО

- Предназначены прежде всего для управления сложностью
- Могут моделировать как саму систему, так и окружение
- Позволяют понять, проанализировать и протестировать систему до её реализации



Модели бывают разные

- Используемые нотации и способы моделирования зависят от целей моделирования
  - От неформальных набросков до исполнимых моделей

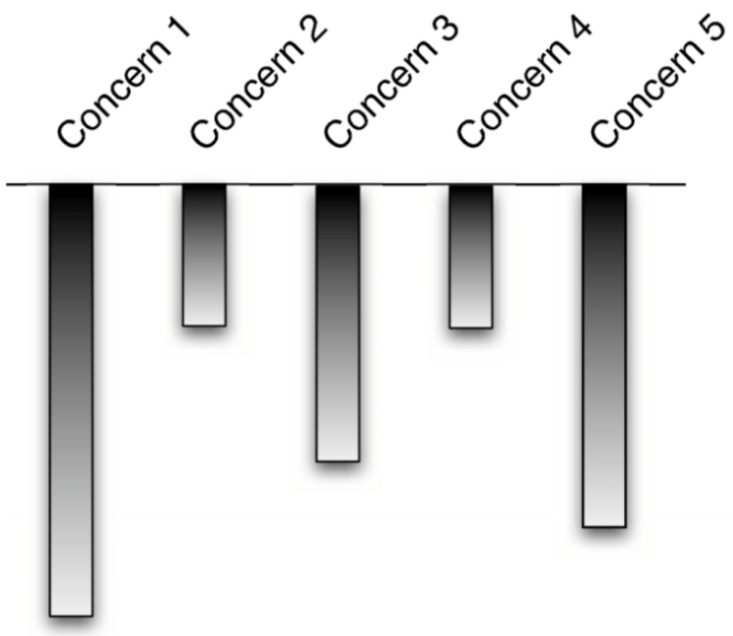


Архитектурные модели

- Архитектура — это набор основных решений, принятых для данной системы
- Архитектурная модель — это некоторый артефакт, который отражает некоторые или все эти решения
- Архитектурное моделирование — это процесс уточнения и документирования этих решений
- Моделирование непосредственно связано с используемой нотацией
  - Нотация архитектурного моделирования — это язык или другое средство описания архитектурных решений

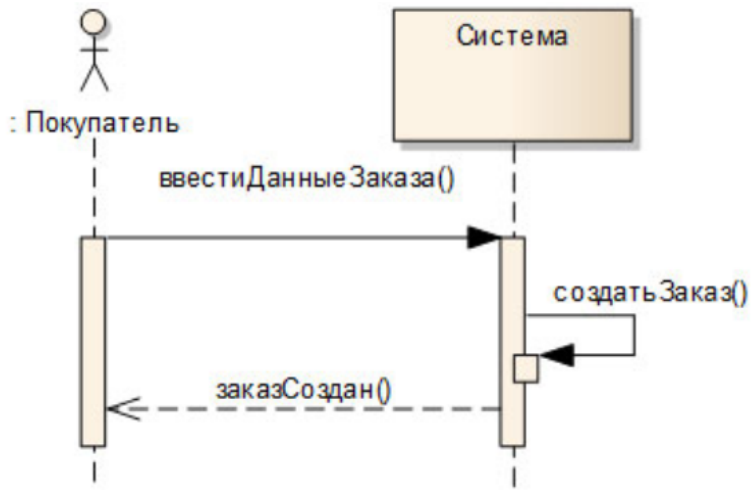
Как выбрать, что моделировать?

- При моделировании надо определиться с:
  - Какие архитектурные решения нуждаются в моделировании
  - На каком уровне детализации
  - Насколько формально
- Необходимо учитывать соотношение трудозатрат и выгоды
  - Стоимость создания *и поддержания* модели не должна быть больше преимуществ от её использования



Возможные преимущества моделей

- Инструмент, направляющий и облегчающий проектирование
- Средство коммуникации между разработчиками
- Наглядный инструмент для общения с заказчиком
- Средство документирования и фиксации принятых решений
- Исходник для генерации кода?



Виды моделей

Естественные языки

- Обычный текст — вполне себе инструмент моделирования
- Очень выразителен, не требует специальных знаний, максимально гибок
- Неоднозначен, неформален, не строг, слишком многословен, бесполезен для автоматической обработки

"The Lunar Lander application consists of three components: a **data store** *component, a* **calculation** *component, and a* **user interface** *component.*

The job of the **data store** *component is to store and allow other components access to the height, velocity, and fuel of the lander, as well as the current simulator time.*

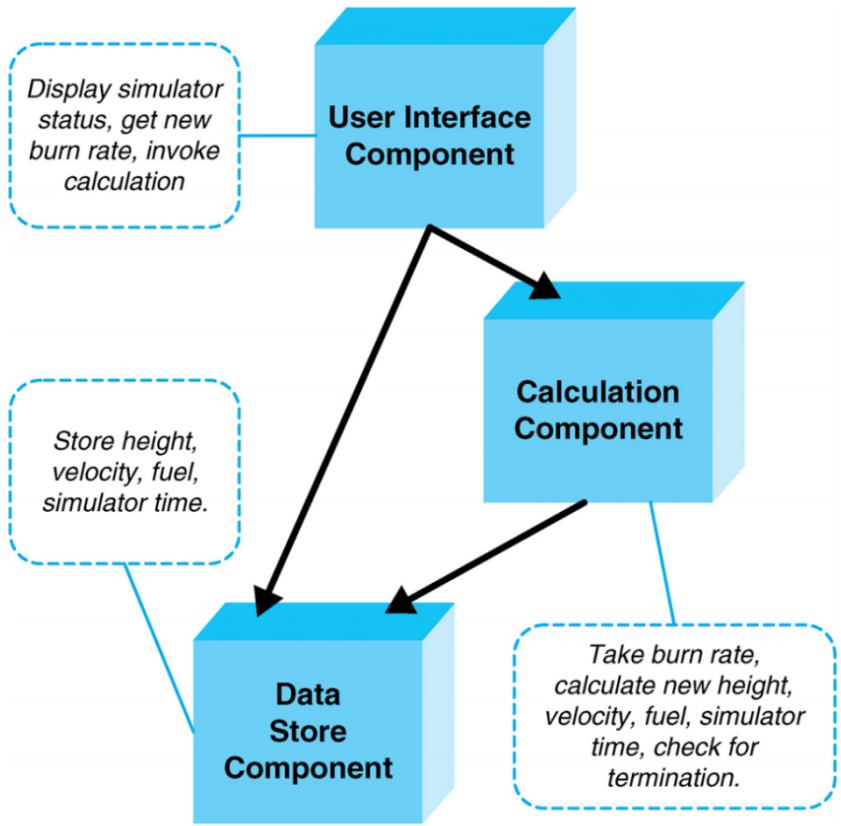
The job of the **calculation** *component is to, upon receipt of a burn-rate quantity, retrieve current values of height, velocity, and fuel from the* data store *component, update them with respect to the input burn-rate, and store the new values back. It also retrieves, increments, and stores back the simulator time. It is also responsible for notifying the calling component of whether the simulator has terminated, and with what state (landed safely, crashed, and so on).*

The job of the **user interface** *component is to display the current status of the lander using information from both the* calculation *and the* data store *components. While the simulator is running, it retrieves the new burn-rate value from the user, and invokes the* calculation *component."*

© N. Medvidovic

Неформальные графические модели

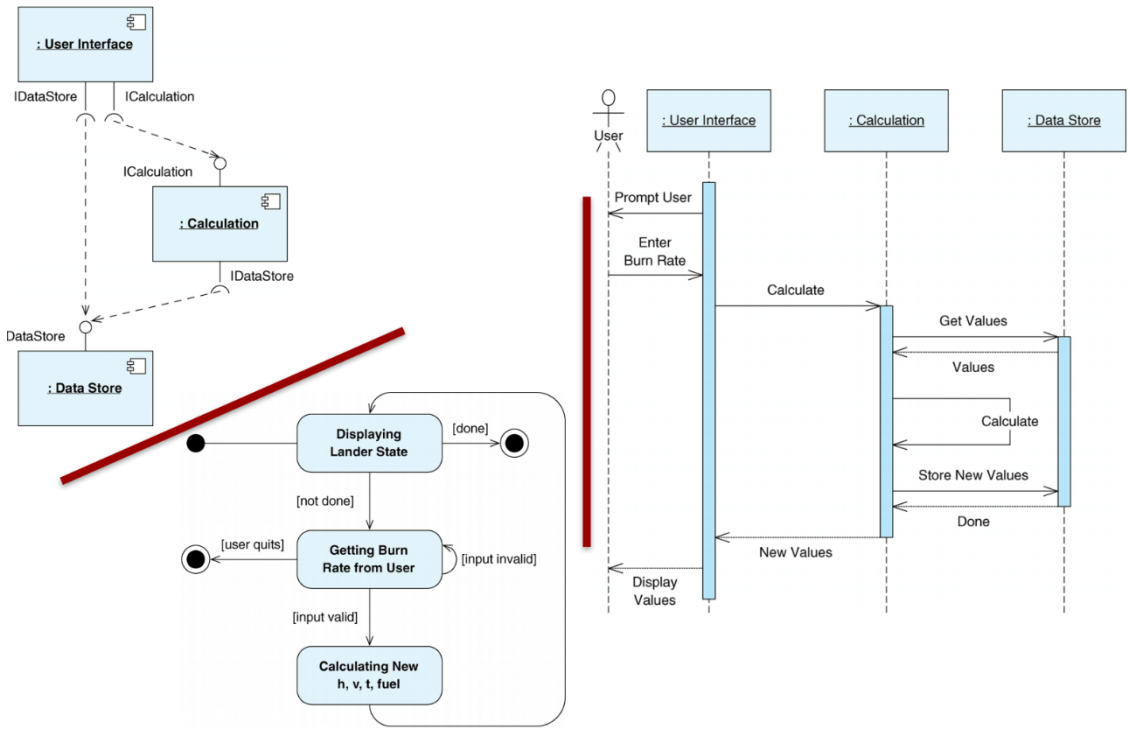
- Диаграммы, рисуемые в PowerPoint, InkScape и подобном
- Могут быть красивыми, как правило, простые, очень гибкая нотация
- Неформальны, неоднозначны, не строги
  - Но часто воспринимаются наоборот
- Практически бесполезны для автоматической обработки



© N. Medvidovic

UML и SysML

- Несколько слабо связанных нотаций (“диаграмм”)
- Поддерживают много точек зрения, общеприняты, широкая поддержка инструментами
- Нет строгой семантики, сложно обеспечить консистентность, сложно расширять



AADL и другие текстовые формальные языки

- Хороши для моделирования встроенных систем и систем реального времени
- Описывают одновременно “железо” и “софт”, продвинутые инструменты анализа
- Слишком многословны и детальны, сложны в изучении и использовании

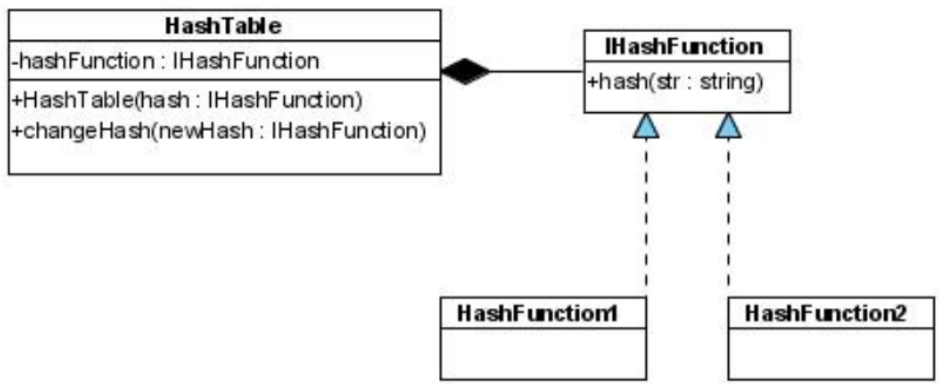
```
data lander_state_data
end lander_state_data;
bus lan_bus_type
end lan_bus_type;

bus implementation lan_bus_type.ethernet
properties
  Transmission_Time => 1 ms .. 5 ms;
  Allowed_Message_Size => 1 b .. 1 kb;
end lan_bus_type.ethernet;
system calculation_type
features
  network : requires bus access
            lan_bus.calculation_to_datastore;
  request_get    : out event port;
  response_get   : in event data port lander_state_data;
  request_store  : out event port lander_state_data;
  response_store : in event port;
end calculation_type;

system implementation calculation_type.calculation
subcomponents
  the_calculation_processor :
              processor calculation_processor_type;
  the_calculation_process : process
              calculation_process_type.one_thread;
```

Вернёмся к визуальным моделям

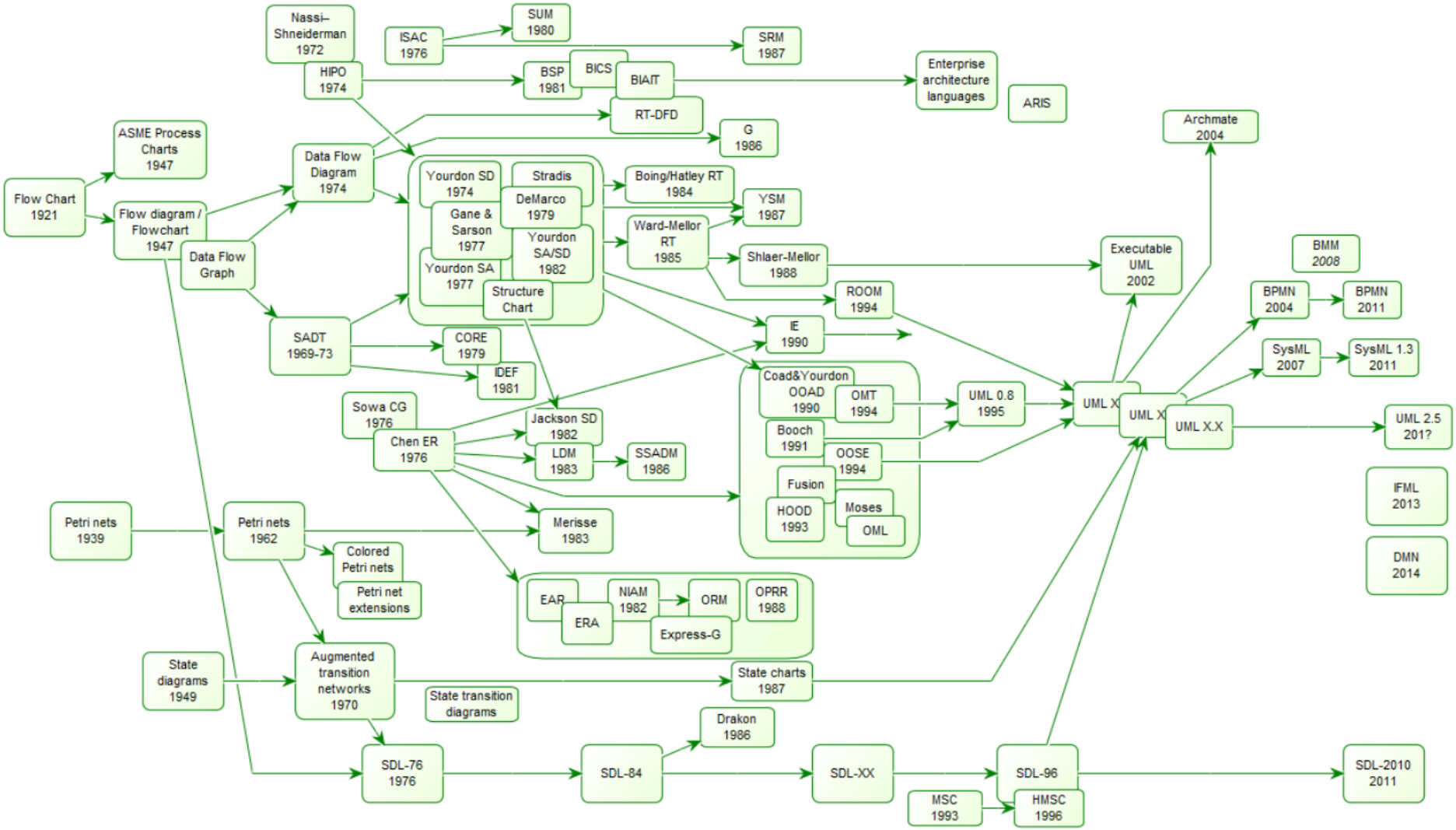
- **Метаформа визуализации** — договорённость о том, как будут представляться сущности языка
- **Точка зрения моделирования** — какой аспект системы и для кого моделируется
- Бывают одноразовые модели, документация и графические исходники
  - **Семантический разрыв** — неспособность модели полностью специфицировать систему



Unified Modeling Language

- Семейство графических нотаций
  - 14 видов диаграмм
- Общая метамодель
- Стандарт под управлением Object Management Group
  - UML 1.1 — 1997 год
  - UML 2.0 — 2005 год
  - UML 2.5 — июнь 2015 года
- Прежде всего, для проектирования ПО
  - После UML 2.0 стали появляться нотации и для инженеров
- Расширяем
  - Профили — механизм легковесного расширения
  - Метамоделирование

История


Nassi–Shneiderman 1972
ISAC 1976
SUM 1980
SRM 1987
HIPO 1974
BSP 1981
BICS
BIAIT
Enterprise architecture languages
ARIS
RT-DFD
G 1986
ASME Process Charts 1947
Flow Chart 1921
Flow diagram / Flowchart 1947
Data Flow Diagram 1974
Data Flow Graph
Yourdon SD 1974
Stradis
DeMarco 1979
Gane & Sarson 1977
Yourdon SA/SD 1982
Yourdon SA 1977
Structure Chart
Boing/Hatley RT 1984
YSM 1987
Ward-Mellor RT 1985
Shlaer-Mellor 1988
Archmate 2004
Executable UML 2002
BMM 2008
BPMN 2004
BPMN 2011
ROOM 1994
IE 1990
SysML 2007
SysML 1.3 2011
SADT 1969-73
CORE 1979
IDEF 1981
Sowa CG 1976
Chen ER 1976
Jackson SD 1982
LDM 1983
SSADM 1986
Coad&Yourdon OOAD 1990
OMT 1994
Booch 1991
OOSE 1994
Fusion
HOOD 1993
Moses
OML
UML 0.8 1995
UML X
UML X
UML X.X
UML 2.5 201?
IFML 2013
DMN 2014
Petri nets 1939
Petri nets 1962
Colored Petri nets
Petri net extensions
Merisse 1983
EAR
ERA
NIAM 1982
ORM
Express-G
OPRR 1988
State diagrams 1949
Augmented transition networks 1970
State transition diagrams
State charts 1987
Drakon 1986
SDL-76 1976
SDL-84
SDL-XX
SDL-96
SDL-2010 2011
MSC 1993
HMSC 1996

Виды диаграмм

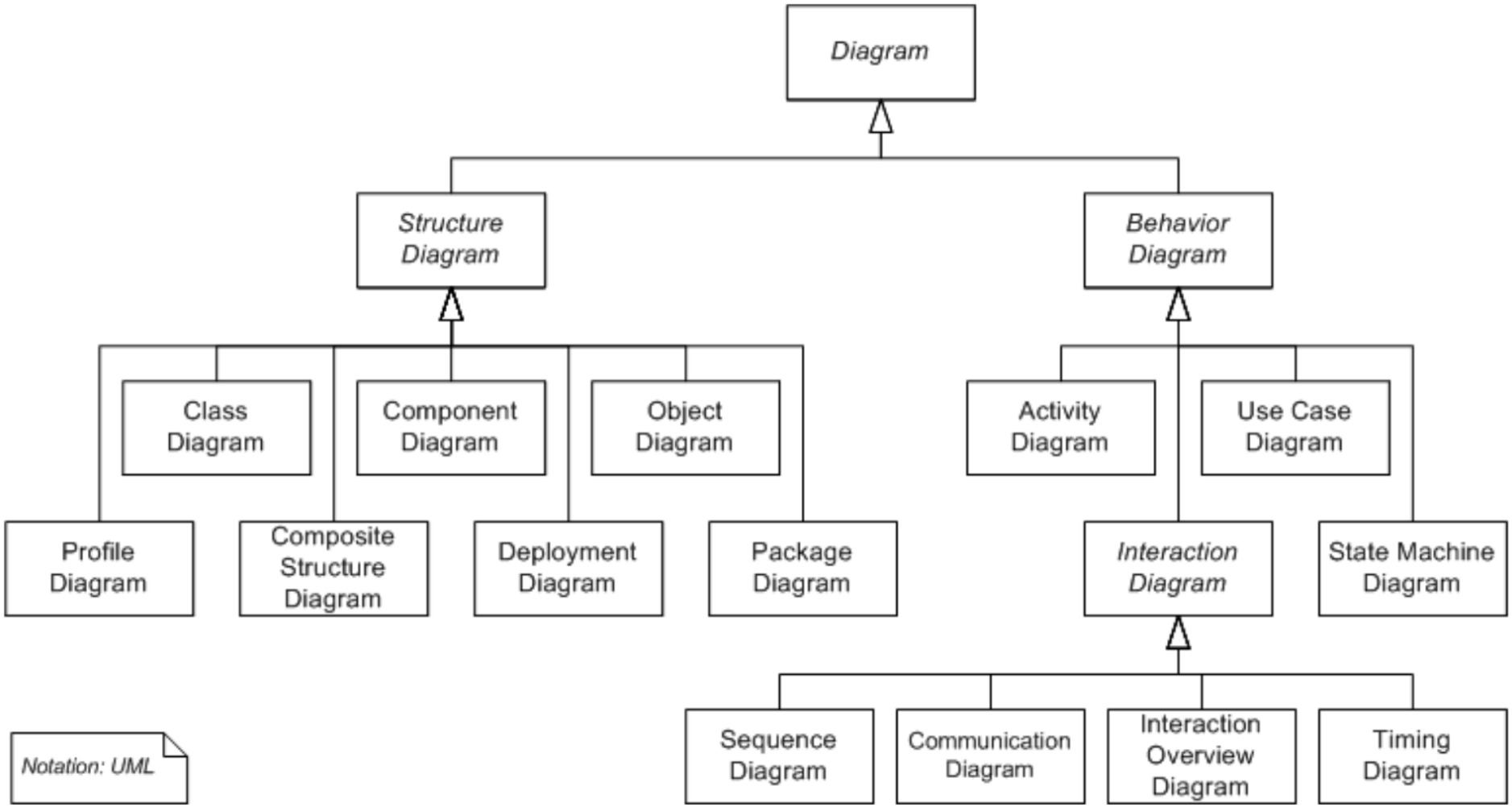
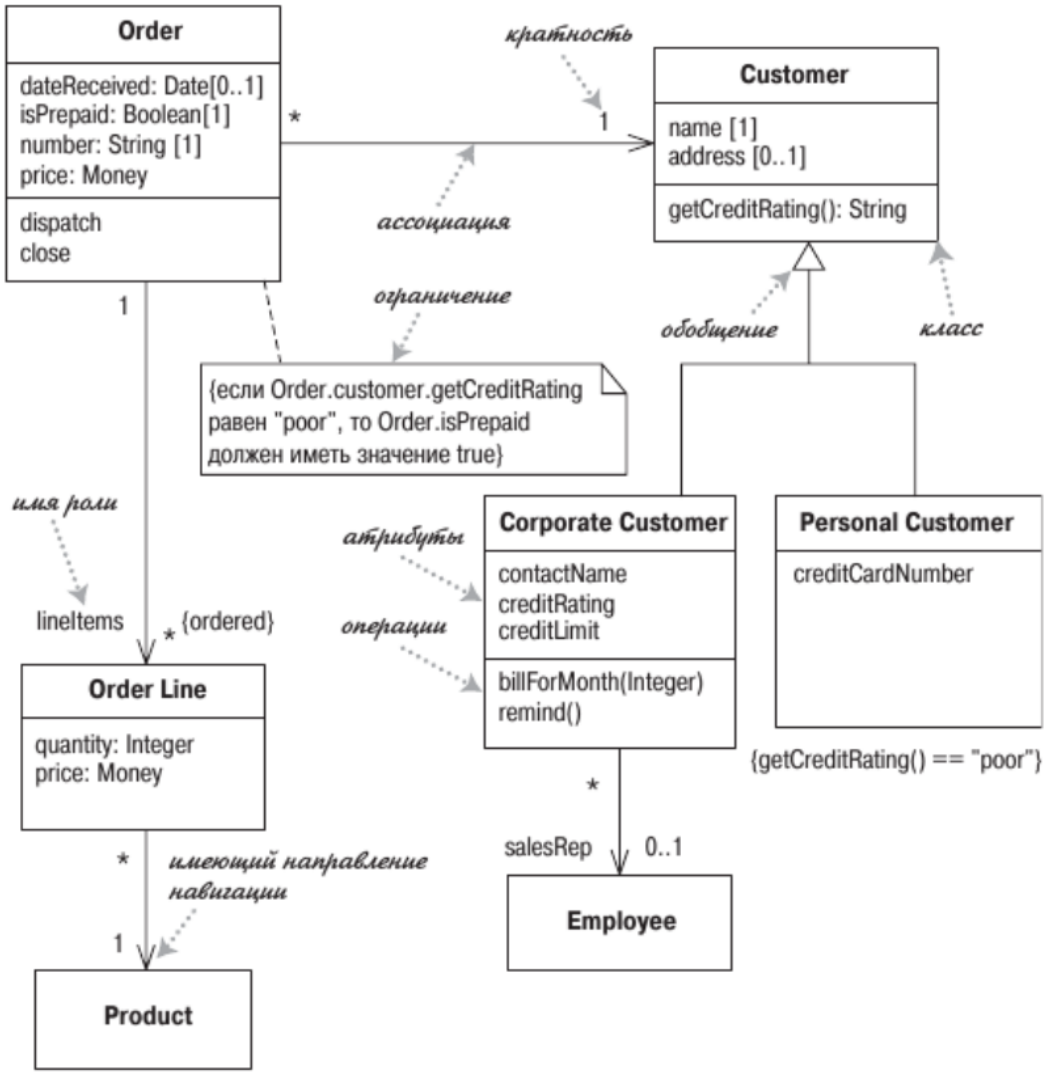


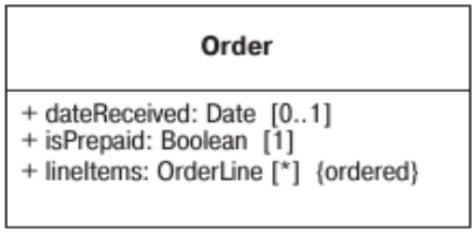
Диаграмма классов



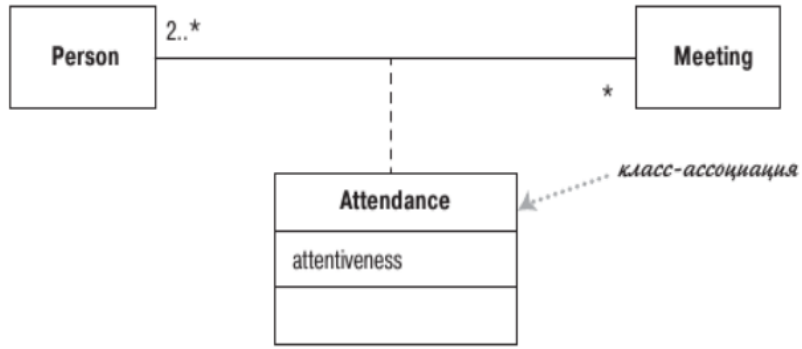
© М. Фаулер. “UML. Основы”

Свойства

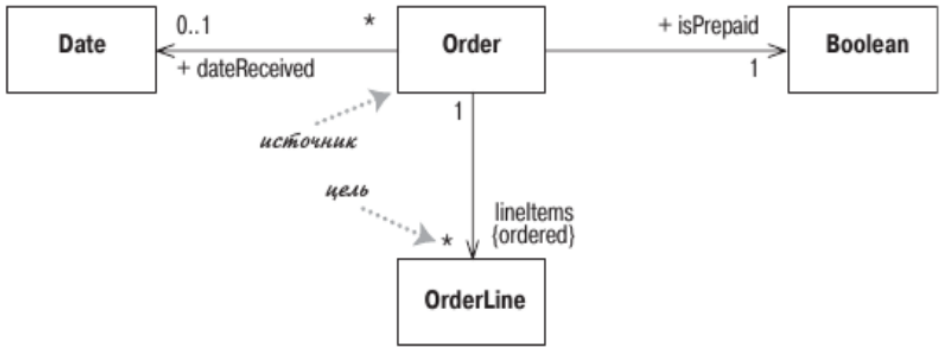
Атрибуты:



Ассоциация-класс:



Ассоциации:



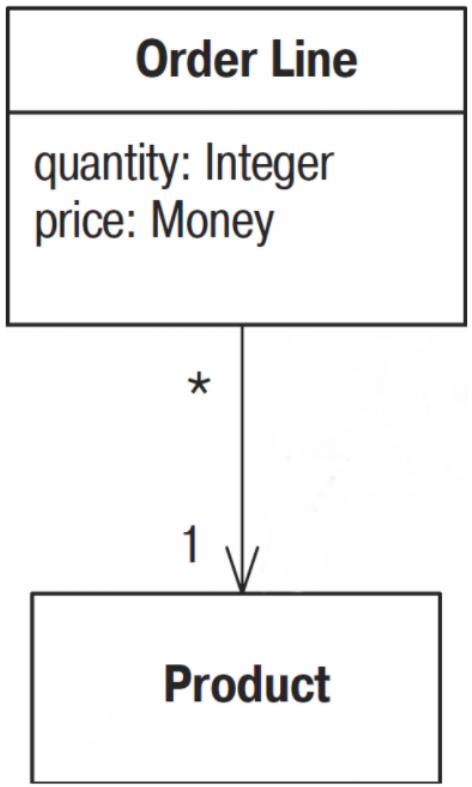
© М. Фаулер. “UML. Основы”

Синтаксис свойств

- Объявление поля:
  - видимость имя: тип кратность = значение по умолчанию {строка свойств}
- Видимость:
  - $+$ (public), $-$ (private), # (protected), ~(package)
- Кратность:
  - 1 (ровно 1 объект), 0..1 (ни одного или один), * (сколько угодно), 1..*, 2..*

Как это связано с кодом

```
public class OrderLine {
    private int quantity;
    private Product product;
    public int getQuantity() {
        return quantity;
    }
    public void setQuantity(int quantity) {
        this.quantity = quantity;
    }
    public Money getPrice() {
        return product.getPrice().multiply(quantity);
    }
}
```

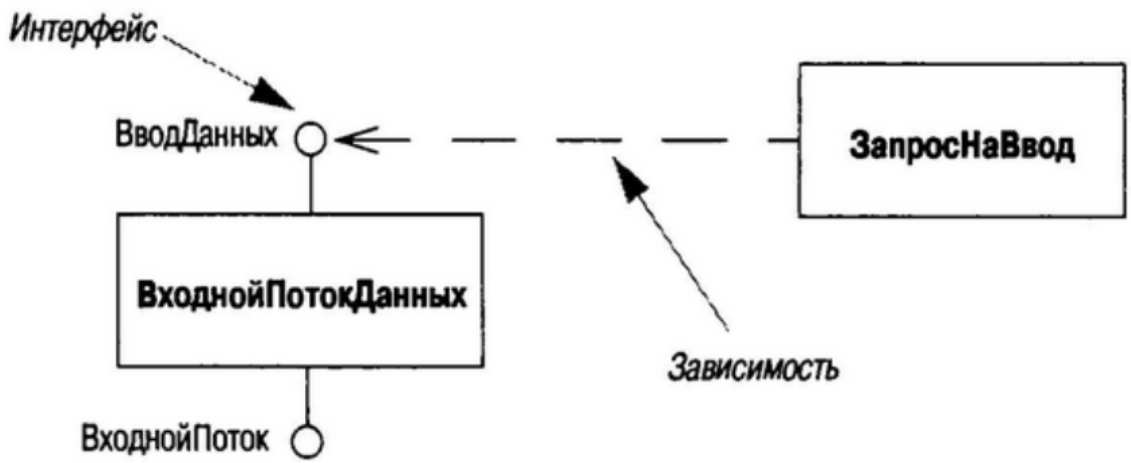


Двунаправленные ассоциации


```
class Car {
   public Person Owner {
      get { return _owner; }
      set {
         if (_owner != null)
            _owner.friendCars().Remove(this);
         _owner = value;
         if (_owner != null)
            _owner.friendCars().Add(this);
      }
   }
   private Person _owner;
}
```

```
class Person {
   public IList Cars {
      get { return ArrayList.ReadOnly(_cars); }
   }
   public void AddCar(Car arg) {
      arg.Owner = this;
   }
   private IList _cars = new ArrayList();
   internal IList friendCars() {
      // должен быть использован
      // только Car.Owner
      return _cars;
   }
}
```

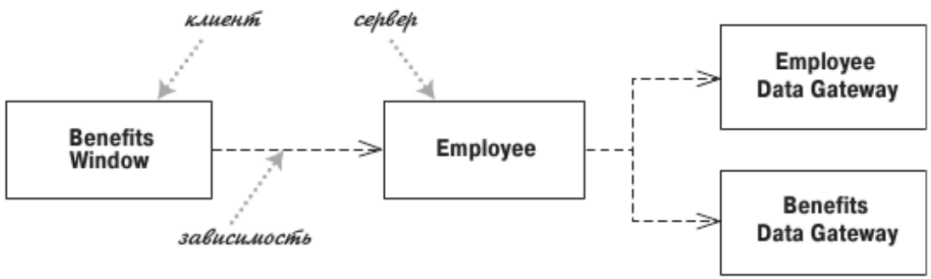
Интерфейсы



© М. Фаулер. “UML. Основы”

Зависимости

- call
- create
- derive
- permit
- realize
- substitute
- instantiate
- refine
- trace
- use
- ...

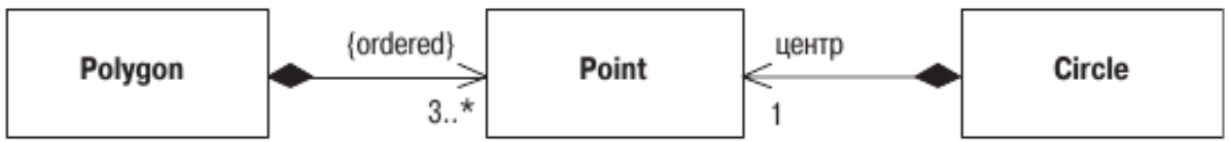


© М. Фаулер. “UML. Основы”

Агрегация и композиция

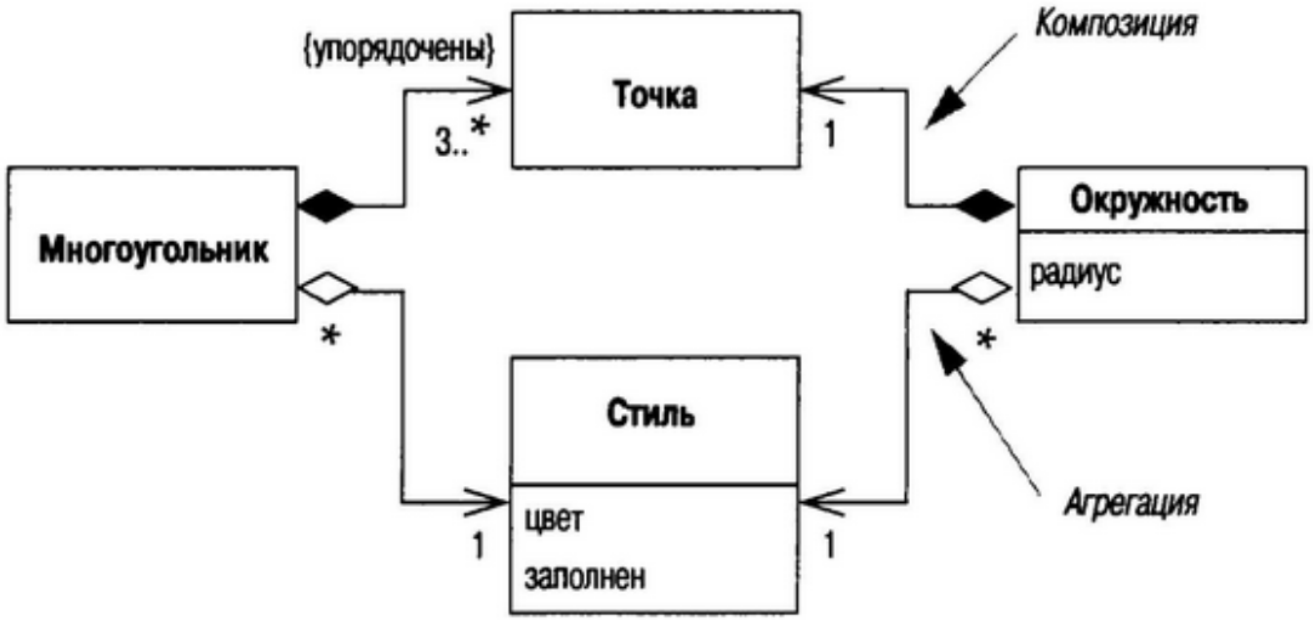
Агрегация:

Композиция:



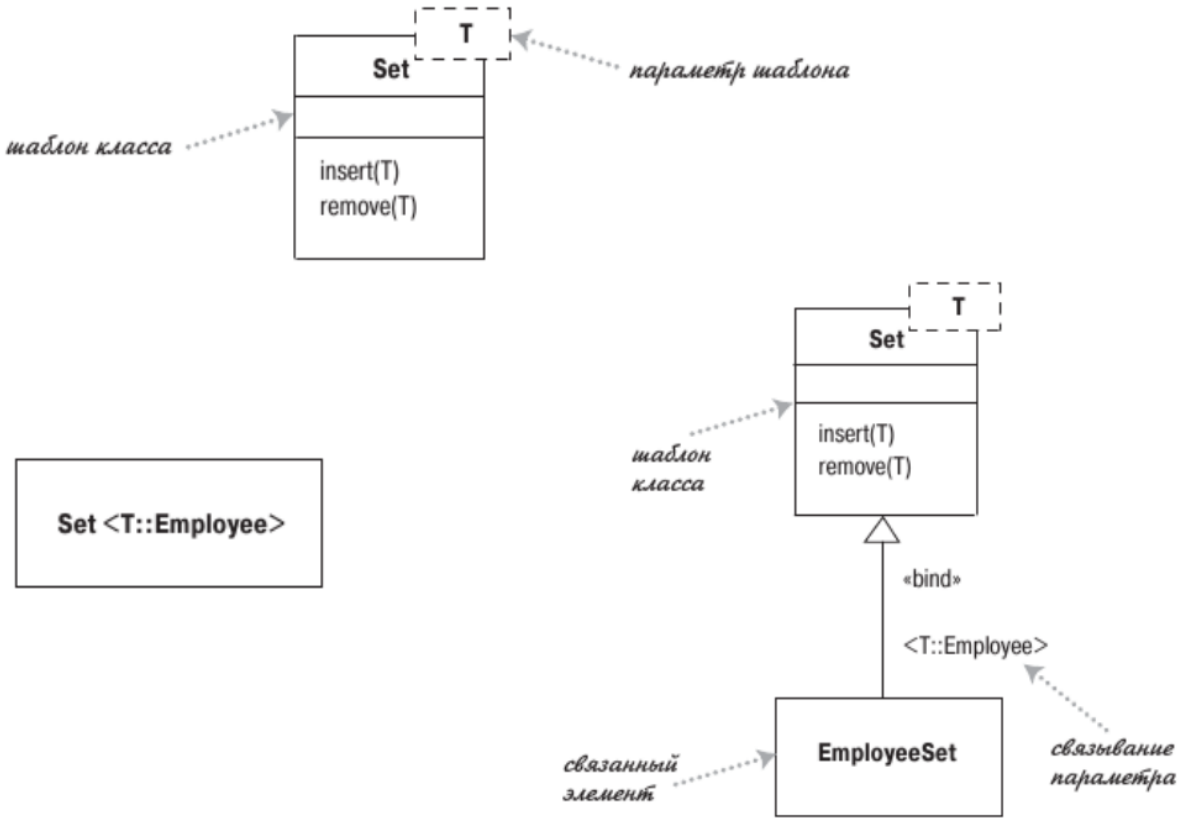
© М. Фаулер. “UML. Основы”

Агрегация и композиция, пример

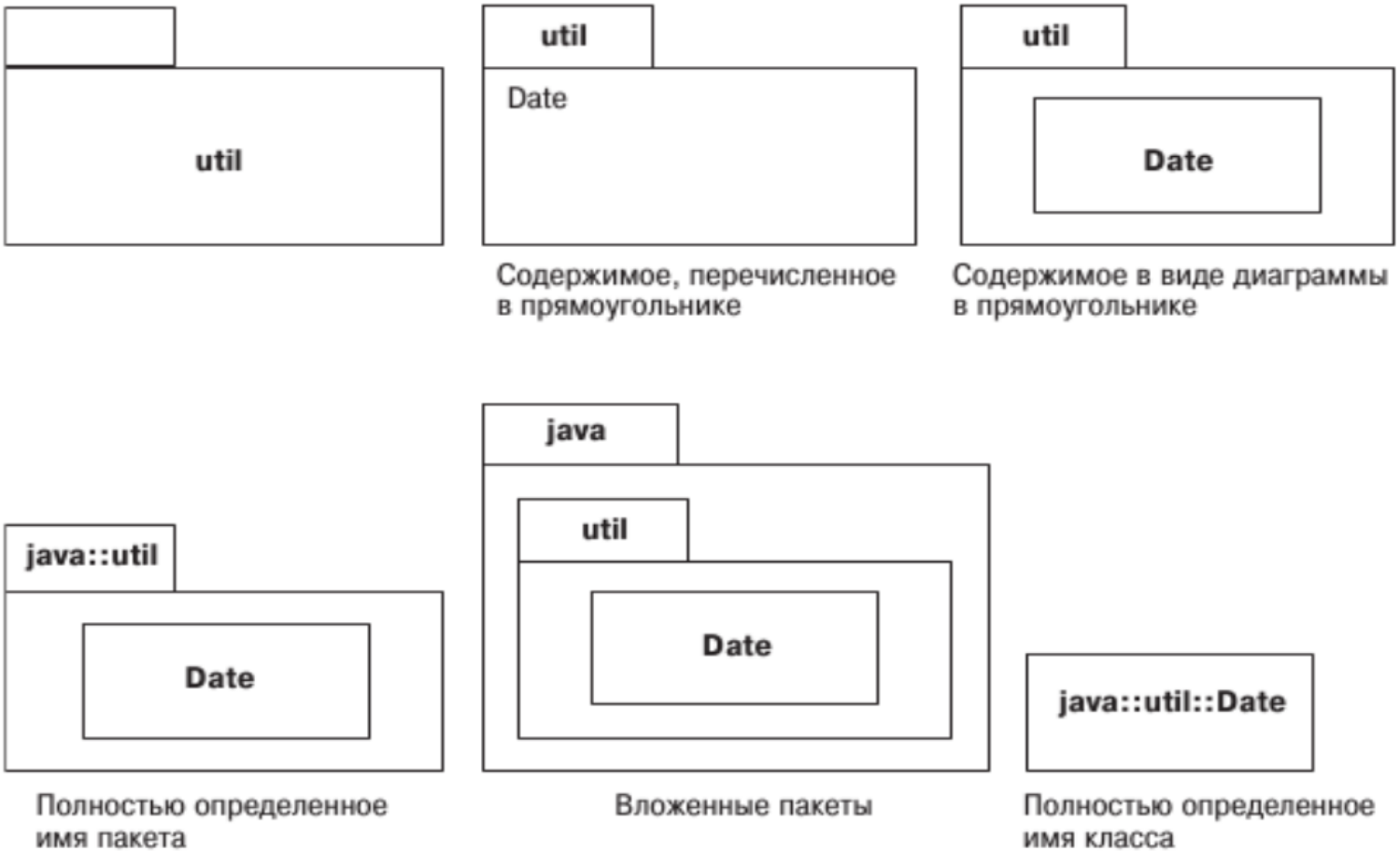


© М. Фаулер. “UML. Основы”

Шаблоны и перечисления

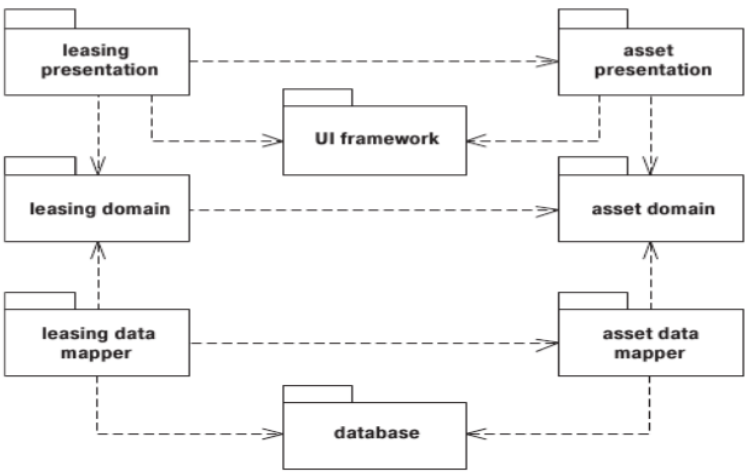


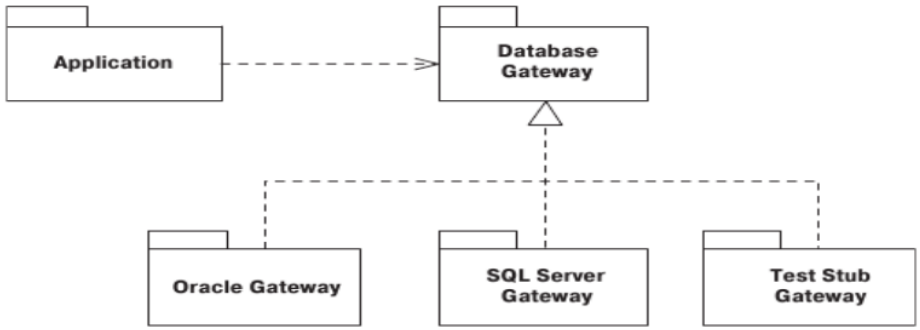
Диаграммы пакетов



© М. Фаулер. “UML. Основы”

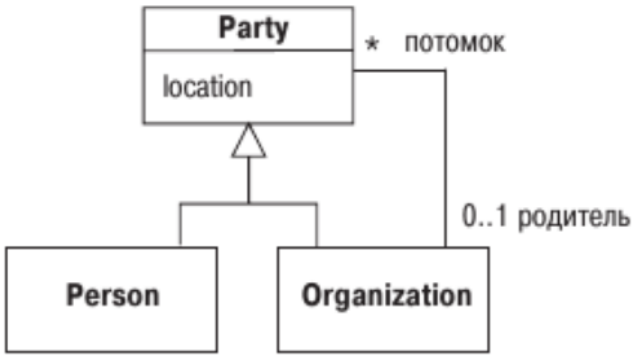
Диаграммы пакетов, зависимости

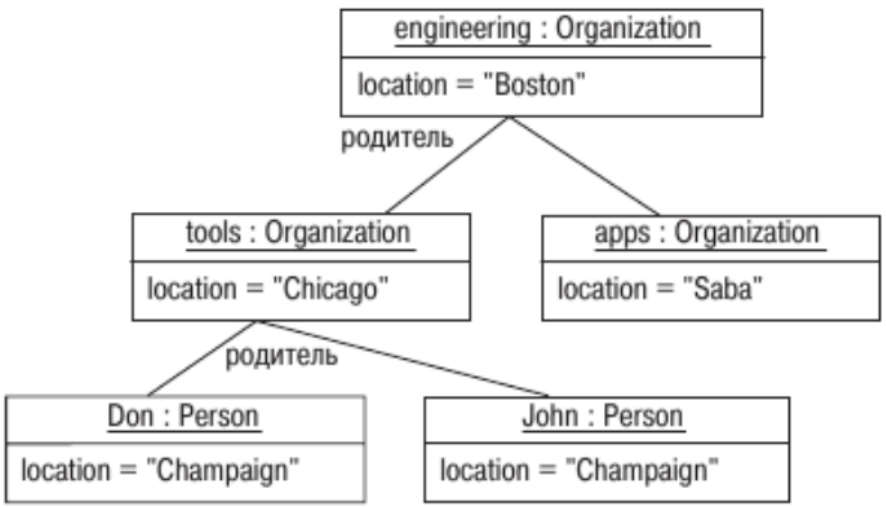




© М. Фаулер. “UML. Основы”

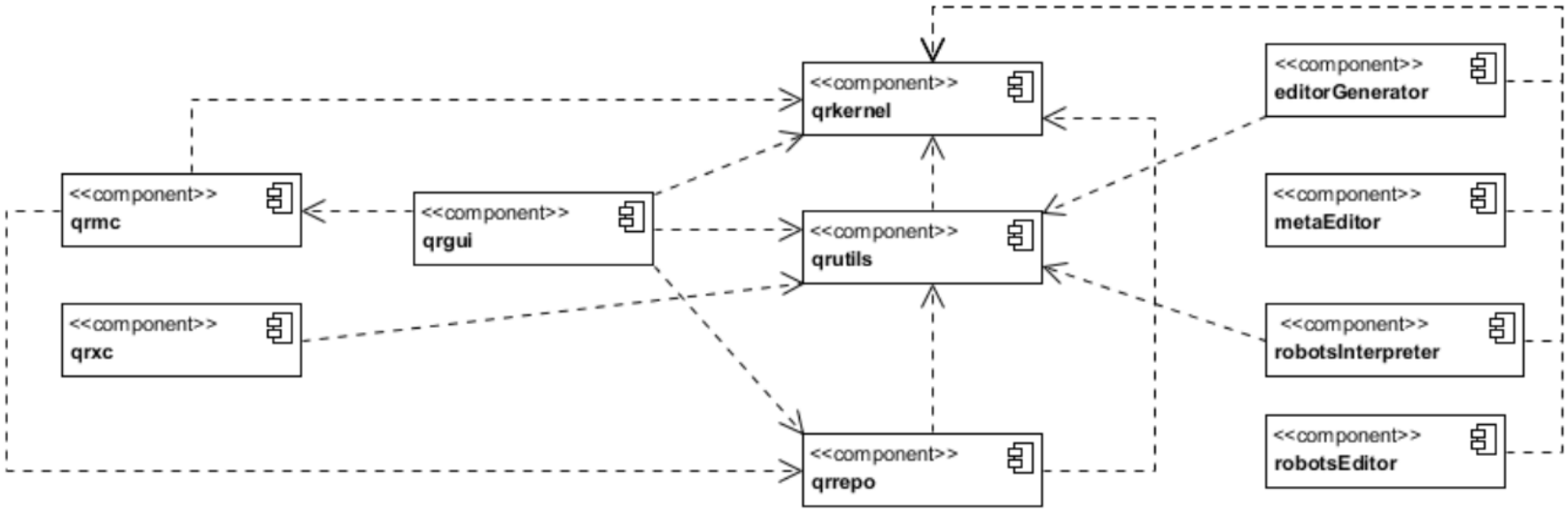
Диаграммы объектов



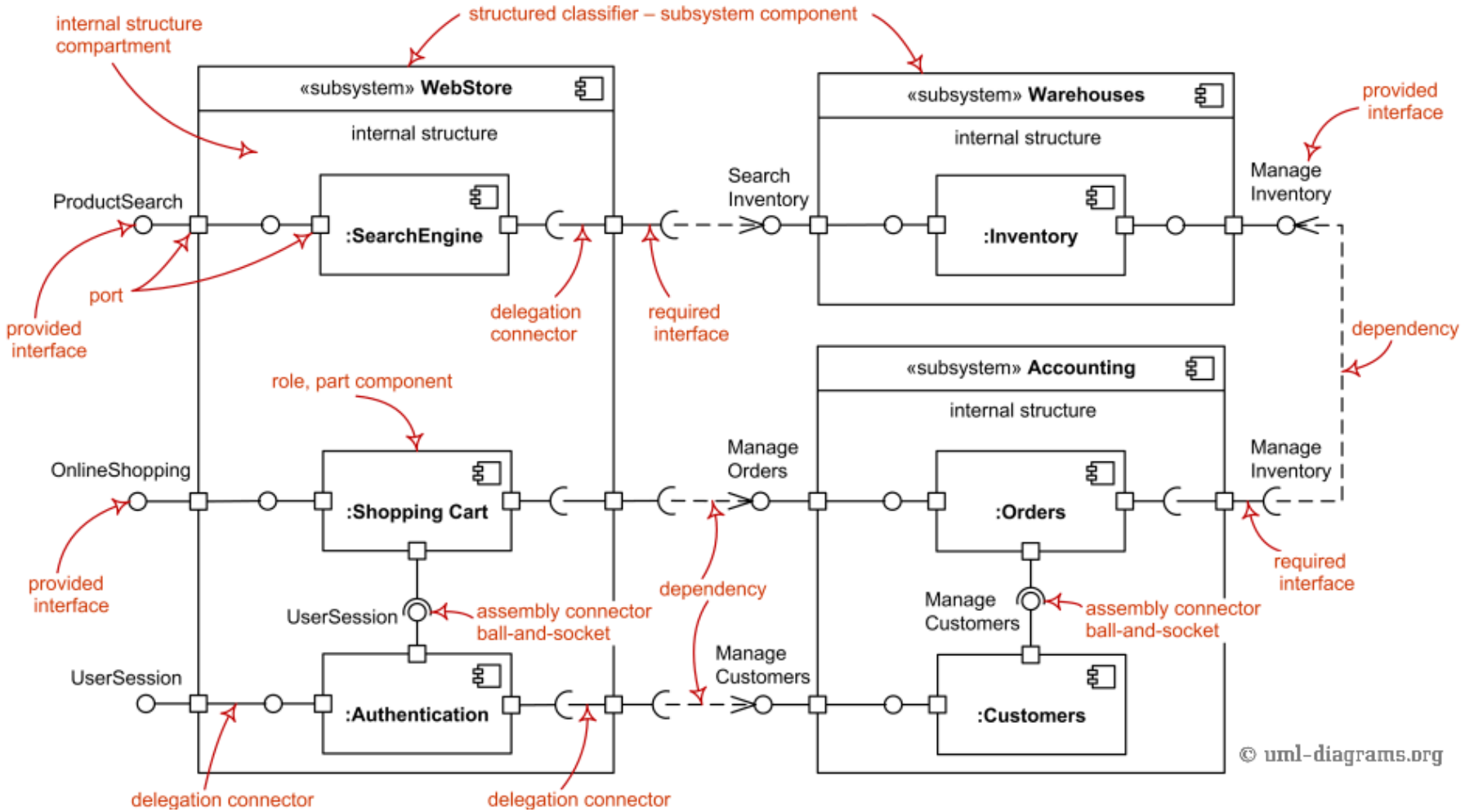


© М. Фаулер. “UML. Основы”

Диаграммы компонентов



Более подробно



Computer-Aided Software Engineering

- В 80-е годы термином CASE называли всё, что помогает разрабатывать ПО с помощью компьютера
  - Даже текстовые редакторы
- Теперь — прежде всего средства для визуального моделирования (UML-диаграммы, ER-диаграммы и т.д.)
- Отличаются от графических редакторов тем, что “понимают”, что в них рисуют
- Нынче чаще используются термины “MDE tool”, “UML tool” и т.д.

Типичная функциональность CASE-инструментов

- Набор визуальных редакторов
- Репозиторий
- Набор генераторов
- Текстовый редактор
- Редактор форм
- Средства обратного проектирования (reverse engineering)
- Средства верификации и анализа моделей
- Средства эмуляции и отладки
- Средства обеспечения командной разработки
- API для интеграции с другими инструментами
- Библиотеки шаблонов и примеров

Примеры CASE-инструментов

- “Рисовалки”
  - Visio
  - Dia
  - SmartDraw
  - Creately
- Полноценные CASE-системы
  - Enterprise Architect
  - Rational Software Architect
  - MagicDraw
  - Visual Paradigm
  - GenMyModel
- Забавные штуки
  - https://www.websequencediagrams.com/
  - http://yuml.me/
  - http://plantuml.com/