Концептуальная модель UML и ее элементы

Лекция 3

Содержание лекции

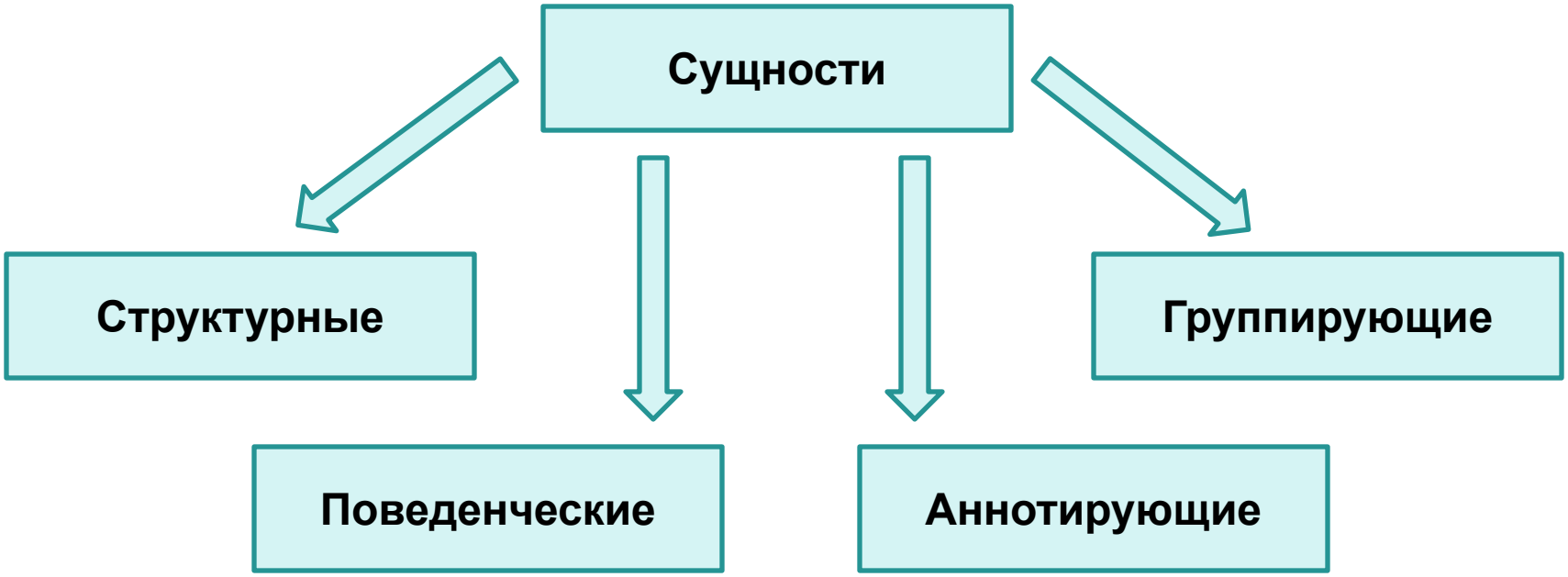
- Сущности
- Связи
- Диаграммы
- Архитектура программной системы в UML

Модель и ее элементы

- ***Модель UML* (UML model)** – это совокупность конечного множества конструкций языка, главные из которых – это **сущности** и **отношения между ними**.

Сущности

- **Сущности** *(things)* – это абстракции, которые являются основными элементами модели, **связи** *(relationships)* соединяют их между собой, а **диаграммы** *(diagrams)* группируют представляющие интерес наборы сущностей.



Структурные сущности = классификаторы (classifiers)

- **Класс** (class) – это описание набора объектов с одинаковыми атрибутами, операциями, связями и семантикой.

Класс реализует один или несколько интерфейсов.

- **Объект** (object) – сущность, обладающая уникальностью и инкапсулирующая в себе состояние и поведение.

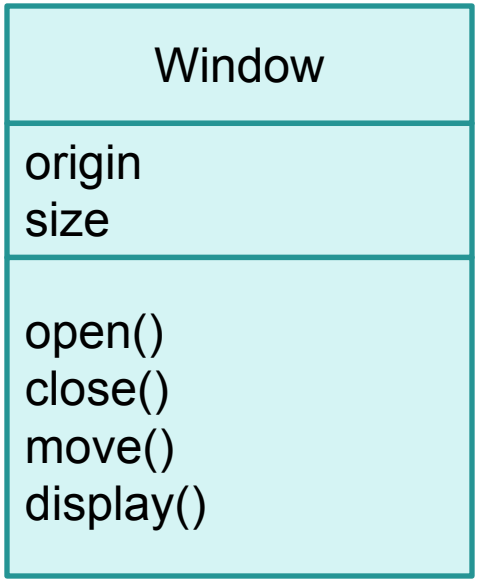


Рис.1. Графическое представление класса

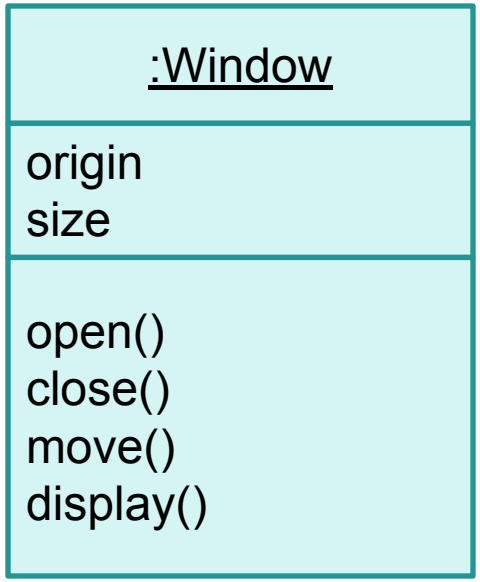
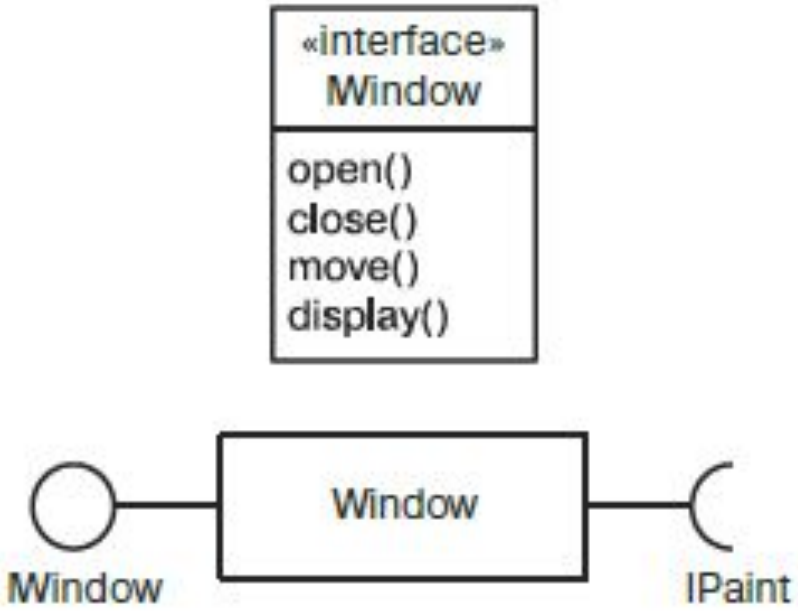


Рис.2. Графическое представление объекта

Структурные сущности = классификаторы (classifiers)

- **Интерфейс** (*interface*) – это набор операций, который специфицирует сервис (набор услуг) класса или компонента, т.е. это множество операций, определяющее набор услуг, которые могут быть запрошены потребителем и предоставлены поставщиком услуг.



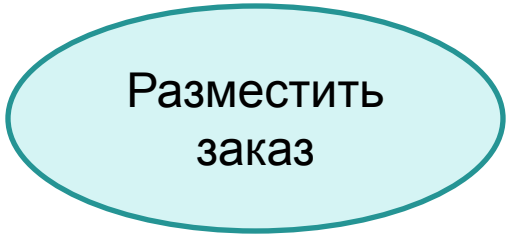
Структурные сущности = классификаторы (classifiers)

- **Кооперация** (*collaboration*) – совокупность ролей и других элементов, которые функционируют вместе, обеспечивая некоторое совместное поведение, представляющее нечто большее ,чем сумма поведений отдельных элементов.
- **Кооперация** (*collaboration*) – совокупность объектов, которые взаимодействуют для достижения некоторой цели.

Конкретный класс или объект могут участвовать в нескольких кооперациях.

Структурные сущности = классификаторы (classifiers)

- **Вариант использования** (*use case*) – это описание последовательности действий, выполняемых системой и приносящих значимый результат конкретному **действующему лицу** (*actor*).
- **Действующее лицо** (*actor*) – сущность, находящаяся вне моделируемой системы и непосредственно взаимодействующая с ней.

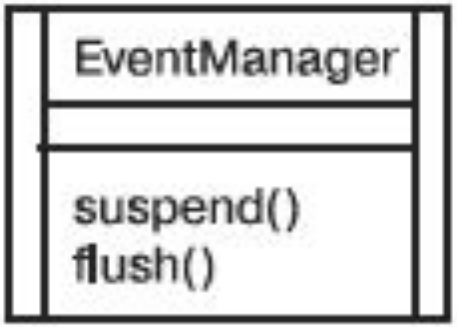


Вариант использования

Действующее лицо

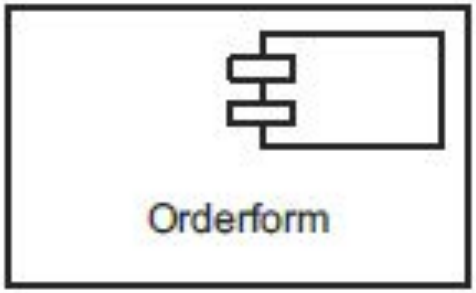
Структурные сущности = классификаторы (classifiers)

- **Активный класс** – это класс, объекты которого являются владельцами одного или нескольких процессов или потоков (threads) и, таким образом, могут инициировать управляющие воздействия.



Структурные сущности = классификаторы (classifiers)

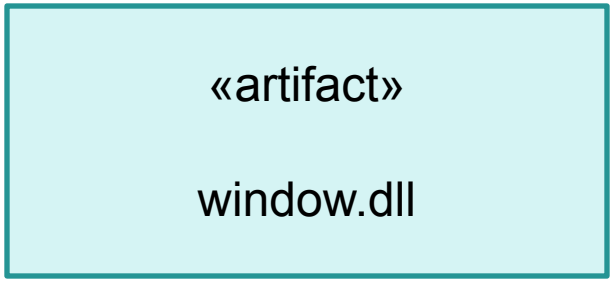
- **Компонент** – это модульная часть системы, которая скрывает свою реализацию за набором внешних интерфейсов.
- **Компонент** (*component*) – модульная часть системы с четко определенным набором требуемых и предоставляемых интерфейсов.



Структурные сущности = классификаторы (classifiers)

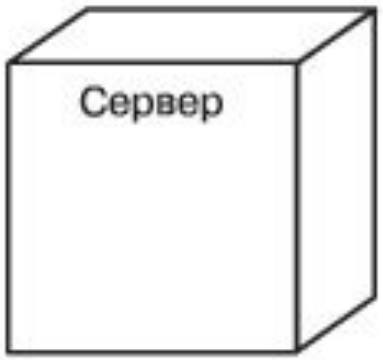
- **Артефакт** (*artifact*) – это элемент информации, который используется или порождается в процессе разработки программного обеспечения.
- Другими словами, артефакт – это физическая единица реализации, получаемая из элемента модели (например, класса или компонента).

Пример: файлы исходного кода, скрипты, исполняемые программы и т.п.



Структурные сущности = классификаторы (classifiers)

- **Узел** (*node*) – это физический элемент, который существует во время исполнения и представляет собой *вычислительный ресурс*, обычно имеющий по меньшей мере некоторую память и часто – *вычислительные возможности*.
- Т.е., **узел** – это вычислительный ресурс, на котором размещаются и при необходимости исполняются артефакты.



Поведенческие сущности

- **Взаимодействие** (*interaction*) – поведение, которое заключается в обмене сообщениями между наборами объектов или ролей в определенном контексте для достижения некоторой цели.

- Взаимодействие включает:
  - Сообщения
  - Действия (actic 

  - Коннекторы (соединения между объектами)

Во взаимодействии внимание сосредоточено на наборе взаимодействующих объектов.

Поведенческие сущности

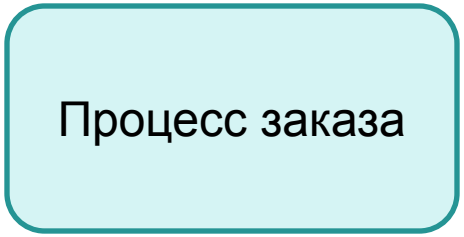
- **Автомат** (*state machine*) – поведение, характеризуемое последовательностью состояний объекта, в которых он оказывается на протяжении своего жизненного цикла в ответ на события, вместе с его реакцией на эти события.
- Автомат включает в себя множество других элементов:
  - **состояния**,
  - **переходы** (из одного состояния в другое)
  - **события** (сущности, которые инициируют переходы),
  - **действия** (реакции на переходы).

Графическое представление состояния

В автомате внимание сосредоточено на жизненном цикле одного объекта.

Поведенческие сущности

- **Деятельность** (*activity*) определяет последовательность шагов процесса вычислений.
- Отдельный шаг деятельности называется **действием** (*action*).



Графическое представление действия

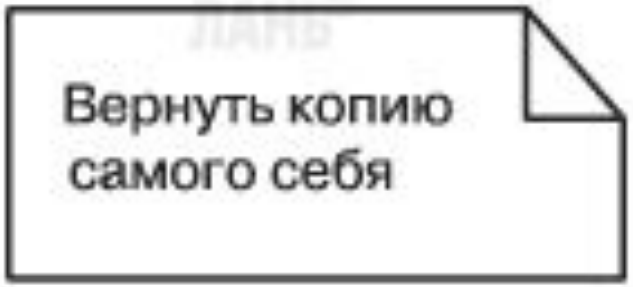
В деятельности внимание сосредоточено на последовательности шагов безотносительно объектов, выполняющих каждый шаг.

Группирующие сущности – организационная часть моделей UML

- **Пакет** (*package*) – это механизм общего назначения для организации проектных решений, который упорядочивает конструкции реализации.
- Существуют вариации пакетов: *подсистемы*, *каркасы* (frameworks).

Аннотирующие сущности – поясняющая часть модели UML

- **Примечание** (*note*) – простой символ, служащий для описания ограничений или комментариев, относящихся к элементу (набору элементов) модели.



Отношения (связи)

- Существует 4 типа связей в UML:

1. Зависимость
2. Ассоциация
3. Обобщение
4. Реализация

Примечание. Есть и различные вариации этих типов отношений, например, включение (include), уточнение (refinement), расширение (extend), след (trace).

Отношения (связи)

- **Зависимость** (*dependency*) – связь между двумя элементами модели ,в которой изменение одного элемента (независимого) может привести к изменению семантики другого элемента (зависимого).

- **Ассоциация** (*association*) – структурная связь между классами, которая описывает набор связей, существующими между объектами – экземплярами классов. **Агрегация** (*aggregation*) – особая разновидность ассоциации, определяющая структурную связь целого с его частями.

Отношения (связи)

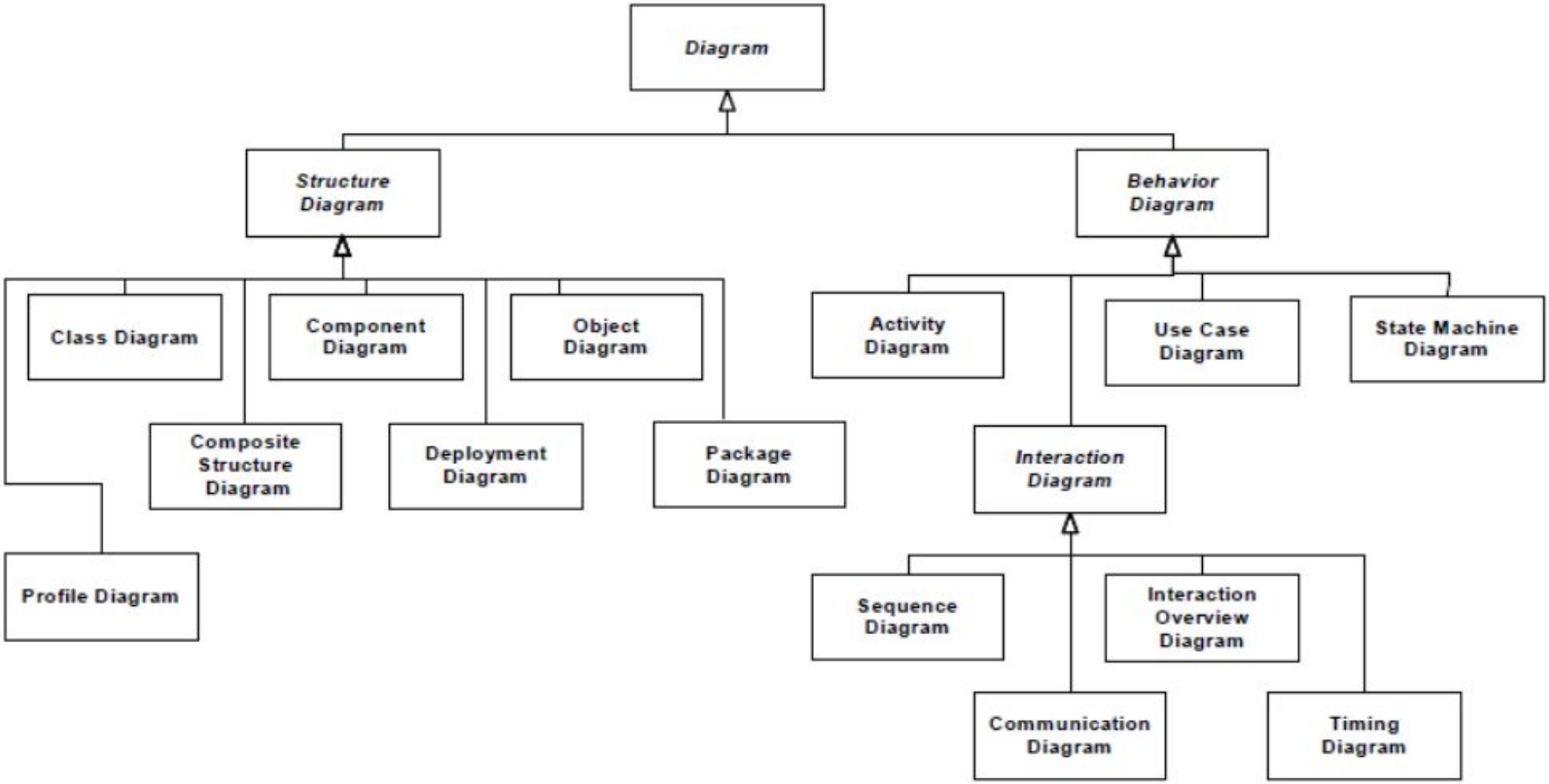
- **Обобщение** (*generalization*) – выражает специализацию или обобщение, в котором специализированный элемент (потомок) строится по спецификациям обобщенного элемента (родителя). Потомок разделяет структуру и поведение родителя.

- **Реализация** (*realization*) – это семантическая связь между классификаторами, когда один из них специфицирует соглашение, которого второй обязан придерживаться. *Отношение реализации* указывает, что одна сущность является реализацией другой.

Диаграммы UML

- Диаграмма классов (class diagram)
- Диаграмма объектов (object diagram)
- Диаграмма компонентов (component diagram)
- Диаграмма составной/композитной структуры (composite structure diagram)
- Диаграмма вариантов использования (use case diagram)
- Диаграмма последовательности (sequence diagram)
- Диаграмма коммуникации (communication diagram)
- Диаграмма состояний (state diagram)
- Диаграмма деятельности (activity diagram)
- Диаграмма размещения (deployment diagram)
- Диаграмма пакетов (package diagram)
- Временная диаграмма (timing diagram)
- Диаграмма обзора взаимодействий (interaction overview diagram)
- Диаграмма профилей (profile diagram)

Таксономия диаграмм UML 2.5



http://www.omg.org/spec/UML/2.5/PDF/

Архитектура программной системы в UML