УО «БОБРУЙСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНО-ЭКОНОМИЧЕСКИЙ КОЛЛЕДЖ»

Концептуальная модель UML. Приемы моделирования

Дисциплина: Технологии разработки программного обеспечения

Преподаватель: Гайшун Алеся Александровна ©

План лекции:

1. Общие принципы
2. Сущности
3. Отношения
4. Диаграммы

1. Общие принципы

UML – это графический язык для построения точных и полных графических моделей, касающиеся анализа, проектирования и реализации, которые должны приниматься в процессе разработки и развертывания системы программного обеспечения.

Использование UML эффективно в:

· информационных системах масштаба предприятия;
· банковских и финансовых услугах;
· телекоммуникациях;
· на транспорте;
· оборонной промышленности, авиации и космонавтике;
· розничной торговле;
· медицинской электронике;
· науке;
· распределенных Web-системах.

Конструктивное использование языка UML основывается на понимании общих принципов моделирования сложных систем и особенностей процесса **объектно-ориентированного анализа и проектирования** (ООАП).

Принцип абстрагирования

включать в модель только те аспекты проектируемой системы, которые имеют непосредственное отношение к выполнению системой своих функций или своего целевого предназначения

Принцип иерархического построения моделей сложных систем

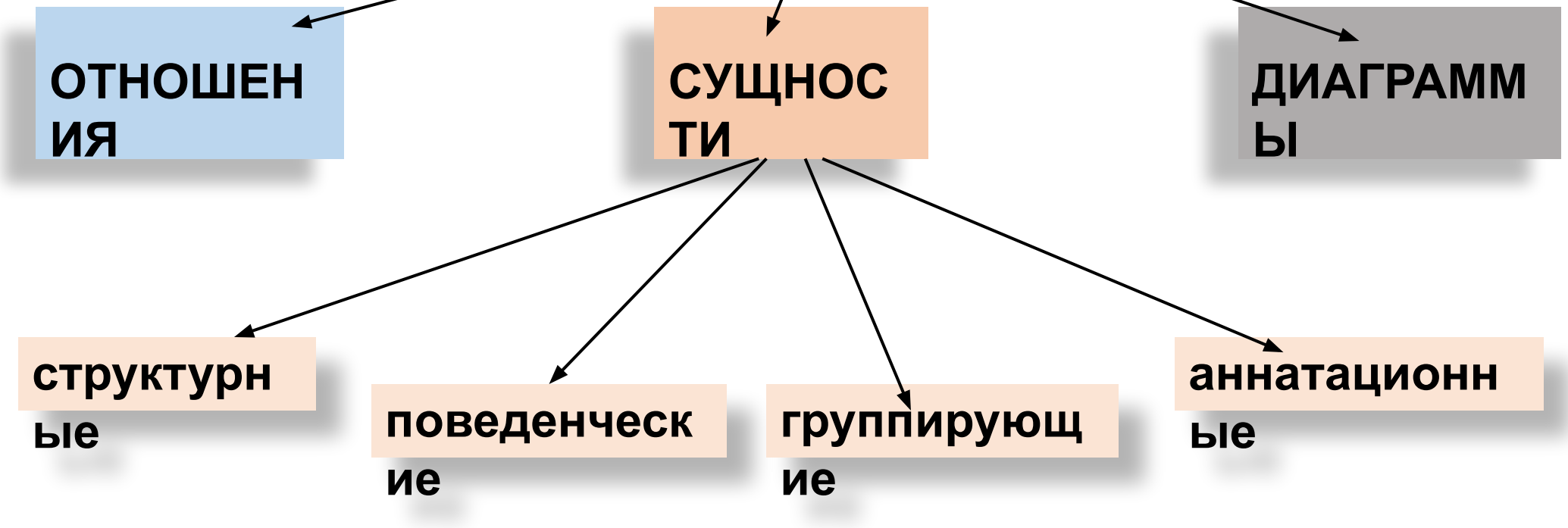
необходимо рассматривать процесс построения модели на разных уровнях абстрагирования или детализации в рамках фиксированных представлений

Принцип многомодельности

никакое единственное представление сложной системы не является достаточным для адекватного выражения всех ее особенностей

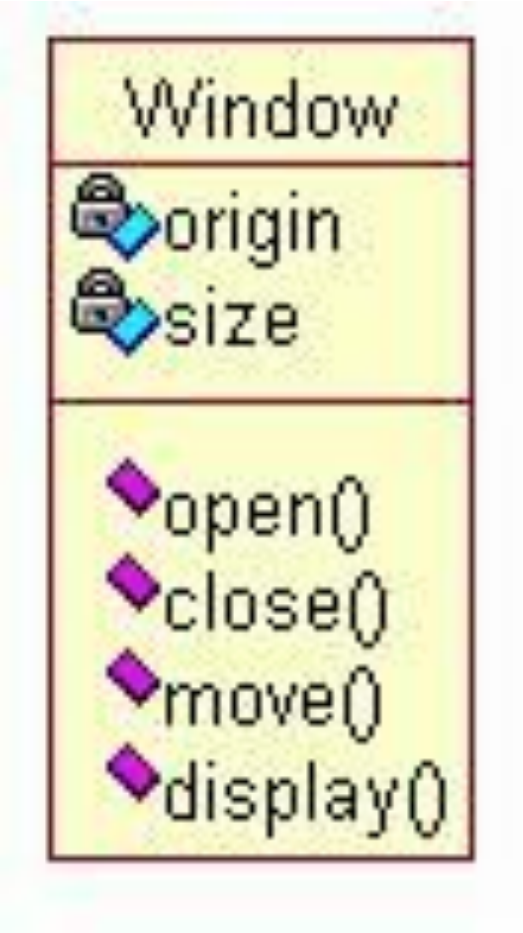
2. Сущности

Объектно-ориентированный анализ и проектирование системы предусматривает использование словаря языка UML, включающего три вида строительных блоков:



Структурные сущности – это имена существительные в моделях на языке UML. Они представляют собой статические части модели, соответствующие концептуальным или физическим элементам системы.

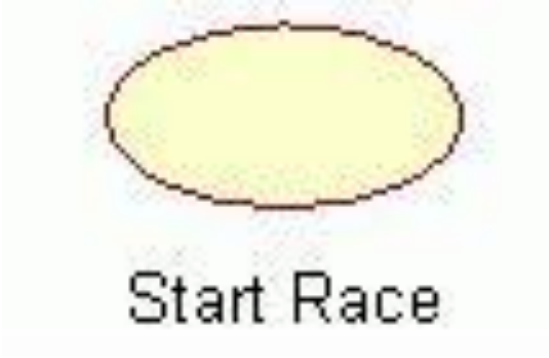
Класс (Class) – это описание совокупности объектов с общими атрибутами, операциями, отношениями и семантикой. Класс реализует один или несколько интерфейсов.



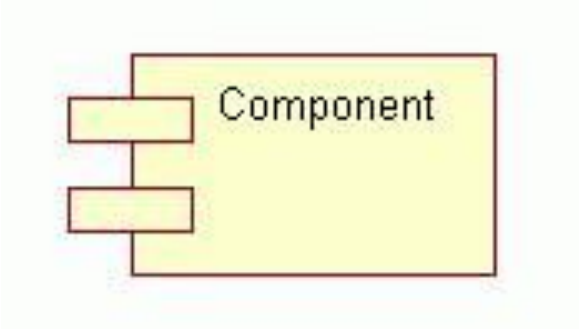
Интерфейс (Interface) – это совокупность операций, которые определяют сервис (набор услуг), предоставляемый классом или компонентом. Интерфейс описывает видимое извне **поведение элемента**. Интерфейс может представлять поведение класса или компонента полностью или частично; он определяет только спецификации операций (сигнатуры), но никогда – их реализации.

Кооперация (Collaboration) определяет взаимодействие; она представляет собой совокупность ролей и других элементов, которые, работая совместно, производят некоторый кооперативный эффект, не сводящийся к простой сумме слагаемых. Один и тот же класс может принимать участие в нескольких кооперациях; таким образом, они **являются реализацией образцов поведения**, формирующих систему.

Прецедент (Use case) – это описание последовательности выполняемых системой действий, которая производит наблюдаемый результат, значимый для какого-то определенного **актера** (Actor).
Прецедент применяется для **структурирования поведенческих сущностей модели**.



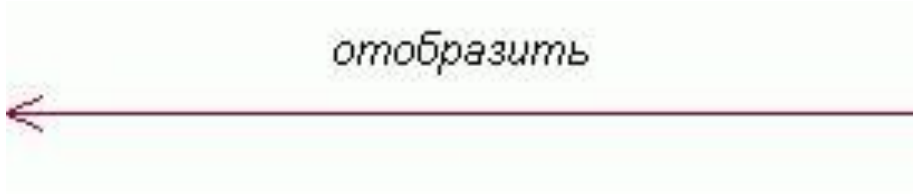
Компонент (Component) – это физическая заменяемая часть системы, которая соответствует некоторому набору интерфейсов и обеспечивает его реализацию (например файлы исходного кода). Компонент, как правило, представляет собой физическую упаковку логических элементов, таких как классы, интерфейсы и кооперации.



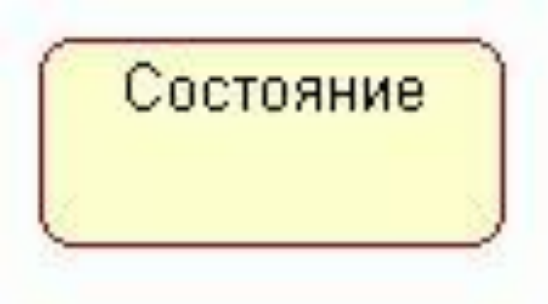
Узел (Node) – это элемент, представляющий вычислительный ресурс, обладающий памятью и способностью обработки. Совокупность компонентов может размещаться в узле, а также мигрировать с одного узла на другой.

Поведенческие сущности (Behavioral things) являются динамическими составляющими модели UML. Это глаголы языка: они описывают поведение модели во времени и пространстве. Существует всего два основных типа поведенческих сущностей.

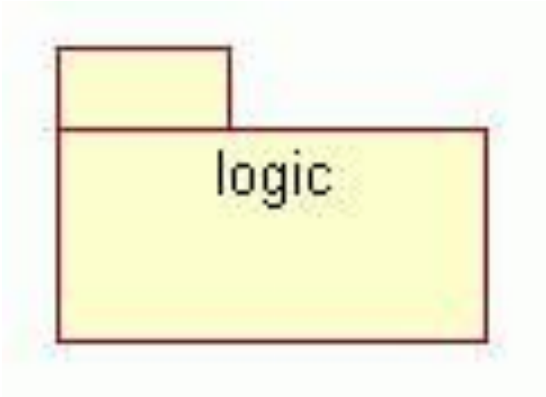
Взаимодействие (Interaction) – это поведение, суть которого заключается в обмене **сообщениями** между объектами в рамках конкретного контекста для достижения определенной цели.



Автомат (State machine) – это алгоритм поведения, определяющий **последовательность состояний**, через которые объект или взаимодействие проходят на протяжении своего жизненного цикла в ответ на различные события, а также реакции на эти события.



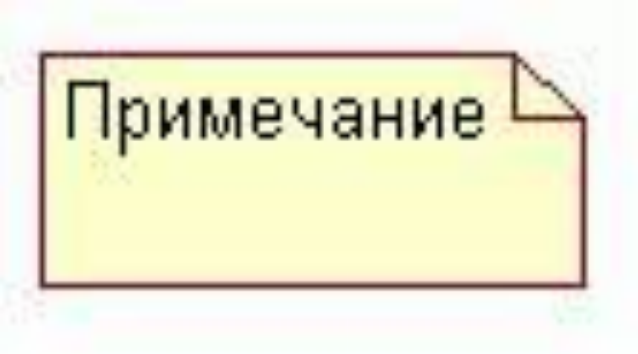
Группирующие сущности являются организующими частями модели UML. Это блоки, на которые можно разложить модель. Есть только одна первичная группирующая сущность, а именно пакет.



Пакеты (Packages) представляют собой универсальный механизм организации элементов в группы. В пакет можно поместить структурные, поведенческие и даже другие группирующие сущности. В отличие от компонентов, существующих во время работы программы, пакеты носят чисто концептуальный характер, то есть существуют только во время разработки.

Аннотационные сущности – пояснительные части модели UML. Это комментарии для дополнительного описания, разъяснения или замечания к любому элементу модели. Имеется только один базовый тип аннотационных элементов – **примечание** (Note).

Примечание – это просто символ для изображения комментариев или ограничений, присоединенных к элементу или группе элементов.



3. Отношения

В языке UML определены четыре типа отношений:
– зависимость;
– ассоциация;
– обобщение;
– реализация.

Зависимость (Dependency) – это семантическое отношение между двумя сущностями, при котором изменение одной из них, независимой, может повлиять на семантику другой, зависимой.

Ассоциация (Association) – структурное отношение, описывающее совокупность связей; связь – это соединение между объектами. Разновидностью ассоциации является агрегирование (Aggregation) – так называют структурное отношение между целым и его частями.

Обобщение (Generalization) – это отношение «специализация/обобщение», при котором объект специализированного элемента (потомок) может быть подставлен вместо объекта обобщенного элемента (родителя или предка). Таким образом, потомок (Child) наследует структуру и поведение своего родителя (Parent).

Реализация (Realization) – это семантическое отношение между классификаторами, при котором один классификатор определяет «контракт», а другой гарантирует его выполнение.

Отношения реализации встречаются в двух случаях: во-первых, между интерфейсами и реализующими их классами или компонентами, а во-вторых, между прецедентами и реализующими их кооперациями.

Диаграмма в UML – это графическое представление набора элементов, изображаемое в виде связанного графа с вершинами (сущностями) и ребрами (отношениями), используемое для визуализации системы с разных точ