ЛЕКЦИЯ 7

Диаграмма классов

Диаграмма классов

Диаграмма классов – диаграмма, предназначенная для представления модели статической структуры программной системы в терминологии классов объектно-ориентированного программирования.

Разработка диаграммы классов преследует следующие цели:

- определить сущности предметной области и представить их в форме классов с соответствующими атрибутами и операциями;
- определить взаимосвязи между сущностями предметной области и представить их в форме типовых отношений между классами;
- разработать исходную логическую модель программной системы для ее последующей реализации в форме физических моделей;
- подготовить документацию для последующей разработки программного кода.

Диаграмма классов

Классификатор – специальное понятие, предназначенное для классификации экземпляров, которые имеют общие характеристики.

Характеристика – понятие, предназначенное для спецификации особенностей структуры и поведения экземпляров классификаторов.

Структурная характеристика является типизированной характеристикой классификатора, которая специфицирует структуру его экземпляров.

Характеристика поведения является характеристикой классификатора, которая специфицирует некоторый аспект поведения его экземпляров.

Класс

Класс – элемент модели, который описывает множество объектов, имеющих одинаковые спецификации характеристик, ограничений и семантики.

Активный класс – класс, каждый экземпляр которого имеет свою собственную нить управления.

Пассивный класс – класс, каждый экземпляр которого выполняется в контексте некоторого другого объекта.

Имя класса

Имя класса – строка текста, предназначенная для идентификации класса на диаграмме классов.

Пример простого имени класса:
SqlConnection

Квалифицированное имя класса:
<Имя пакета 1>::.. ::<Имя пакета N>::<Имя класса>

Пример квалифицированного имени класса:
System::Data::Sql::SqlConnection

Имя абстрактного класса записывается курсивом

Атрибуты класса

Атрибут класса служит для представления отдельной структурной характеристики или свойства, которое является общим для всех объектов данного класса.

Описание атрибута:
<атрибут> ::= [<видимость>] [‘/’] <имя> [‘:’ <тип>]
[‘[’ <кратность> ‘]’] [‘=’ <значение по умолчанию>]
[‘{’ <модификатор атрибута> [‘,’ < модификатор атрибута >]* ‘}’]

Виды видимости:
‘-’ – закрытый,
‘#’ – защищенный,
‘+’ – общедоступный,
‘~’ – пакетный.

Статические атрибуты выделяются подчеркиванием строки описания атрибута

Атрибуты класса

readOnly	Атрибут является атрибутом только для чтения
Union	Атрибут является производным объединением его подмножеств
subsets <имя атрибута>	Атрибут является собственным подмножеством атрибута с именем <имя атрибута>
redefines <имя атрибута>	Атрибут переопределяет некоторый наследуемый атрибут с именем <имя атрибута>
ordered	Значение атрибута является упорядоченным
unique	Значения многозначного атрибута не могут иметь дубликатов
<ограничение атрибута>	Выражение специфицирует некоторое ограничение, применяемое к данному атрибуту

Атрибуты класса

Кратность является спецификацией допустимой мощности множества при инстанцировании соответствующего значения модели.

<кратность> ::= <диапазон-кратности> [‘{’ <указатель-упорядоченности> [‘,’ <указатель-уникальности>] ‘}’]
<диапазон-кратности> ::= [<нижняя-граница> ‘..’] <верхняя-граница>
<нижняя-граница> ::= <целое-число> | <спецификация-значения>
<верхняя-граница> ::= ‘*’ | <спецификация-значения>
<указатель-упорядоченности> ::= ‘ordered’ | ‘unordered’
<указатель-уникальности> ::= ‘unique’ | ‘nonunique’

Примеры:
+ имяСотрудника: String {readOnly}
# / возрастСотрудника: Integer
+ номерТелефона: Integer [*] {unique}
- заработнаяПлата: Currency = $500

Операции класса

Операция класса служит для представления отдельной характеристики поведения, которая является общей для всех объектов данного класса.

<операция> ::=
[<видимость>] <имя-операции> ‘(’ [<список-параметров>] ‘)’
[‘:’ <тип-результата>] [‘{’ <свойство-операции>
[‘,’ <свойство-операции>] ‘}’]

<список-параметров> ::= <параметр> [‘,’ <параметр>]*

<параметр> ::= [<направление>] <имя> ‘:’ <тип> [‘[’ <кратность> ‘]’]
[‘=’ <значение-по-умолчанию>] [‘{’ <свойсвто-параметра>
[‘,’ <свойсвто-параметра>]* ‘}’]

Направления параметров: in, inout, out, return

Предусловие для операции определяет условие, которое должно быть истинным, когда эта операция вызывается.

Постусловие для операции определяет условие, которое должно быть истинным, когда вызов операции успешно завершился, в предположении, что все предусловия удовлетворены.

Операции класса

query	Операция не изменяет внутреннее состояние объекта
Concurrency = sequential	Последовательное выполнение параллельных вызовов операции
Concurrency = guarded	“Охраняемое” выполнение параллельных вызовов операции
Concurrency = concurrent	Возможно параллельное выполнение вызовов операции

Примеры:

+ добавить(in номерТелефона: Integer [*] {unique})
- изменить (in заработнаяПлата: Currency)
+ создать () : Boolean

Отношения

На диаграмме классов могут присутствовать следующие отношения между классами:

- ассоциация;
- зависимость;
- агрегация;
- композиция;
- обобщение;
- реализация.

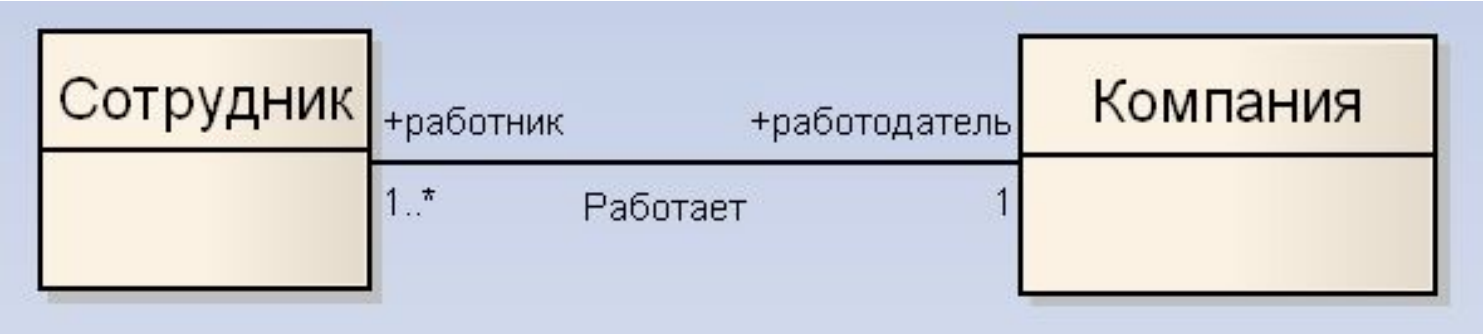
Ассоциация

Ассоциация – произвольное отношение или взаимосвязь между классами.

Имя конца ассоциации специфицирует роль, которую играет класс расположенный на соответствующем конце ассоциации.

Видимость конца ассоциации специфицирует возможность доступа к соответствующему концу ассоциации с других ее концов.

Кратность конца ассоциации специфицирует возможное количество экземпляров соответствующего класса, которое может соотносится с одним экземпляром класса на другом конце ассоциации.

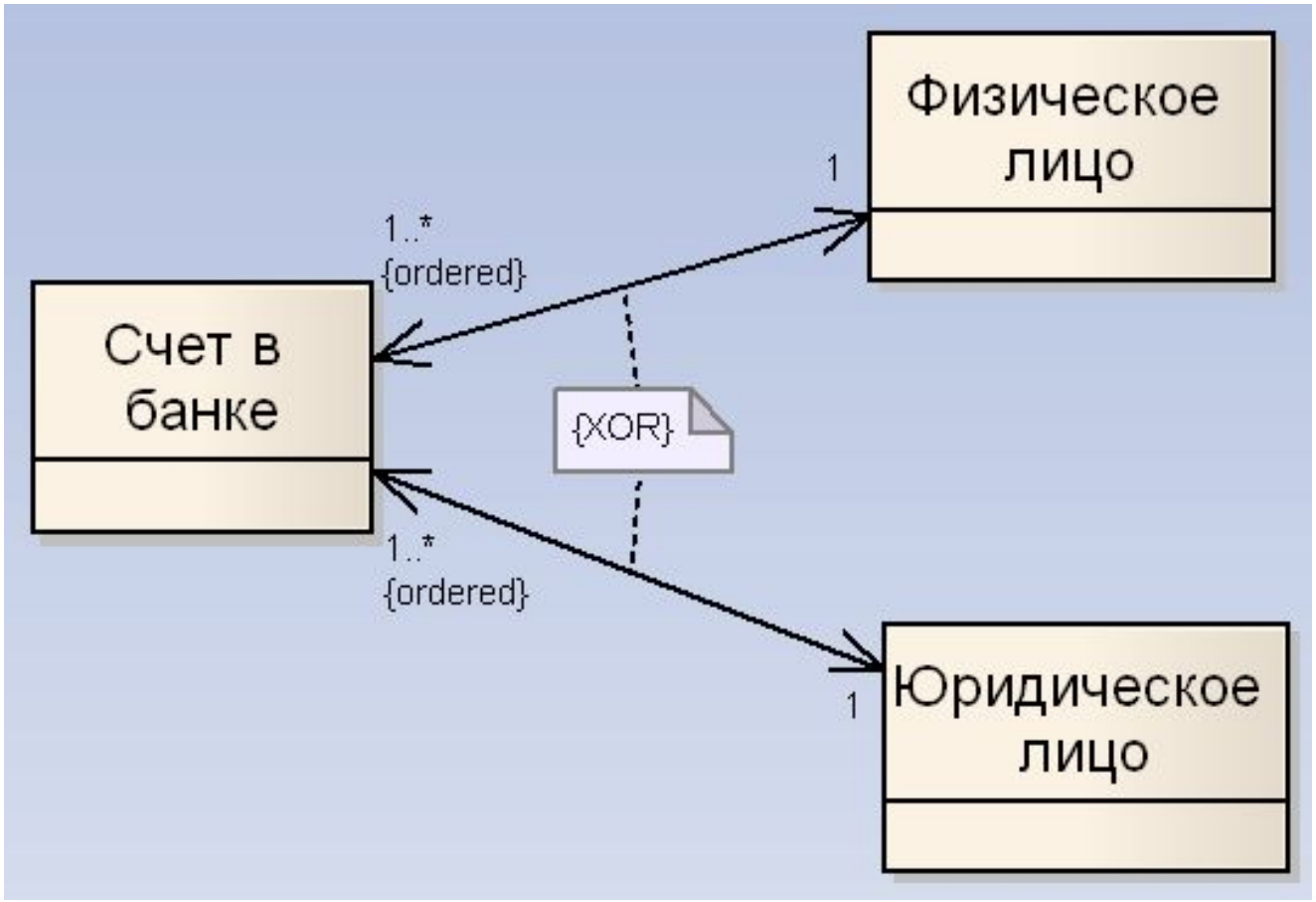


Ассоциация

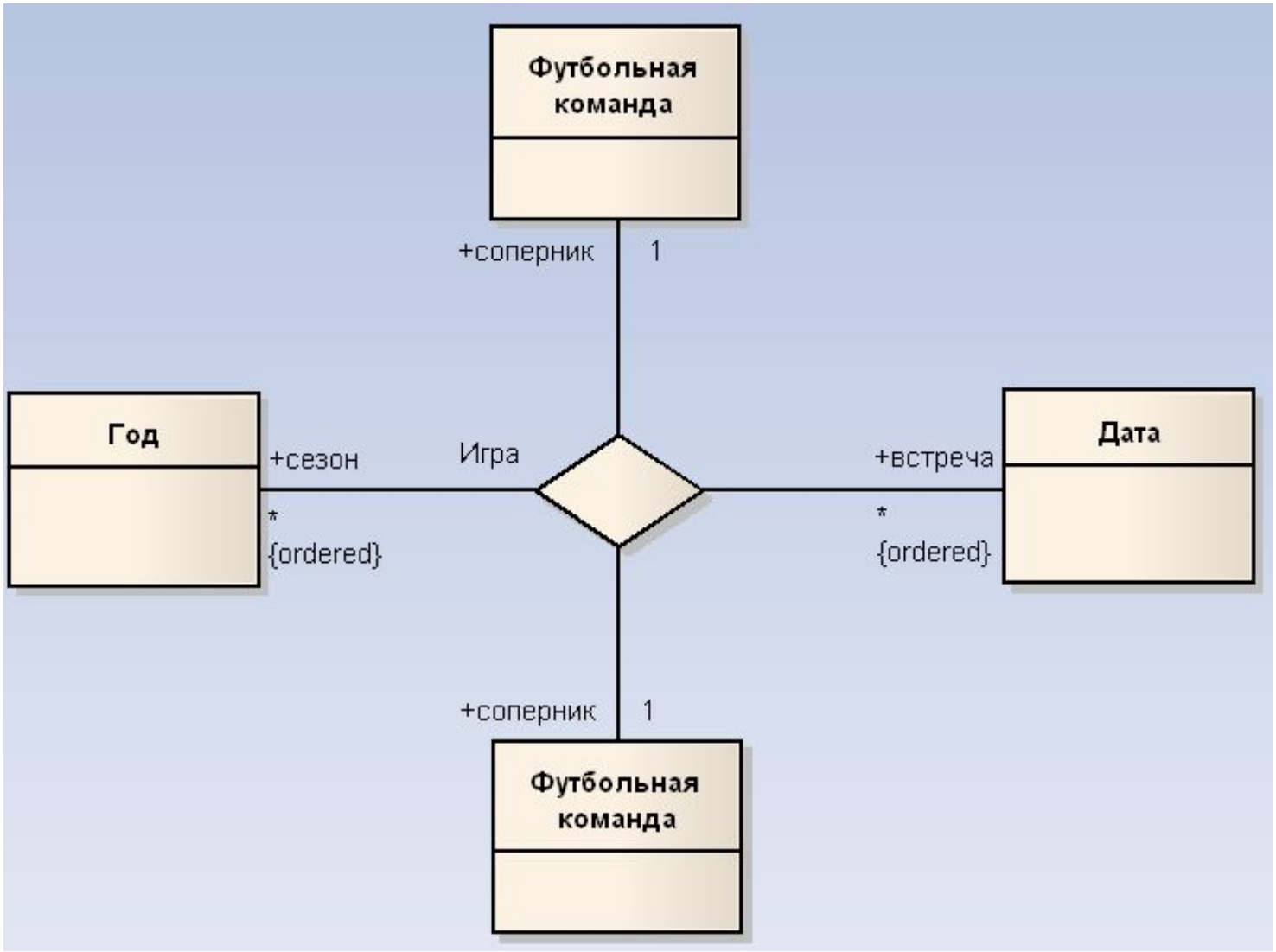
Символ наличия навигации, указывает на то, что соответствующий класс является доступным для навигации со стороны классов на других концах ассоциации.

Символ отсутствия навигации указывает на то, что соответствующий класс является недоступным для навигации со стороны классов на других концах ассоциации.

Исключающая ассоциация

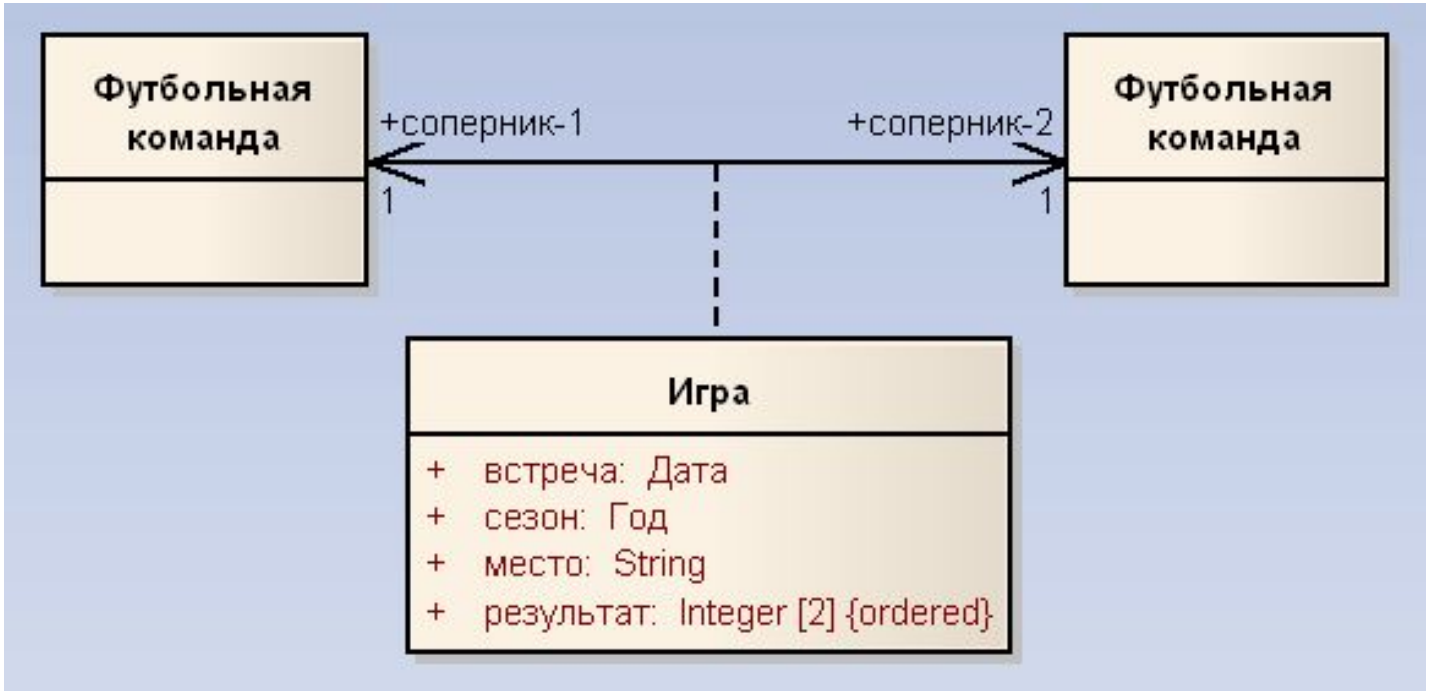


N-арная ассоциация



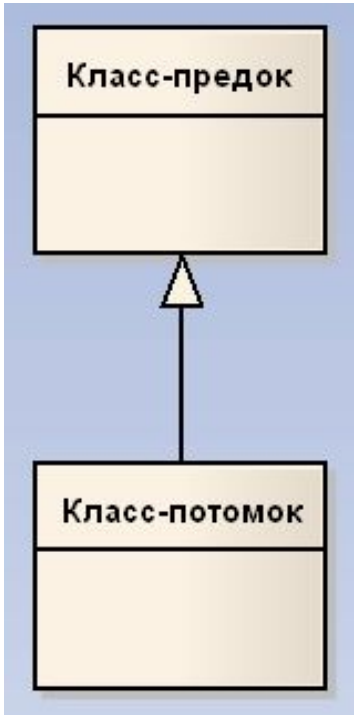
Ассоциация-класс

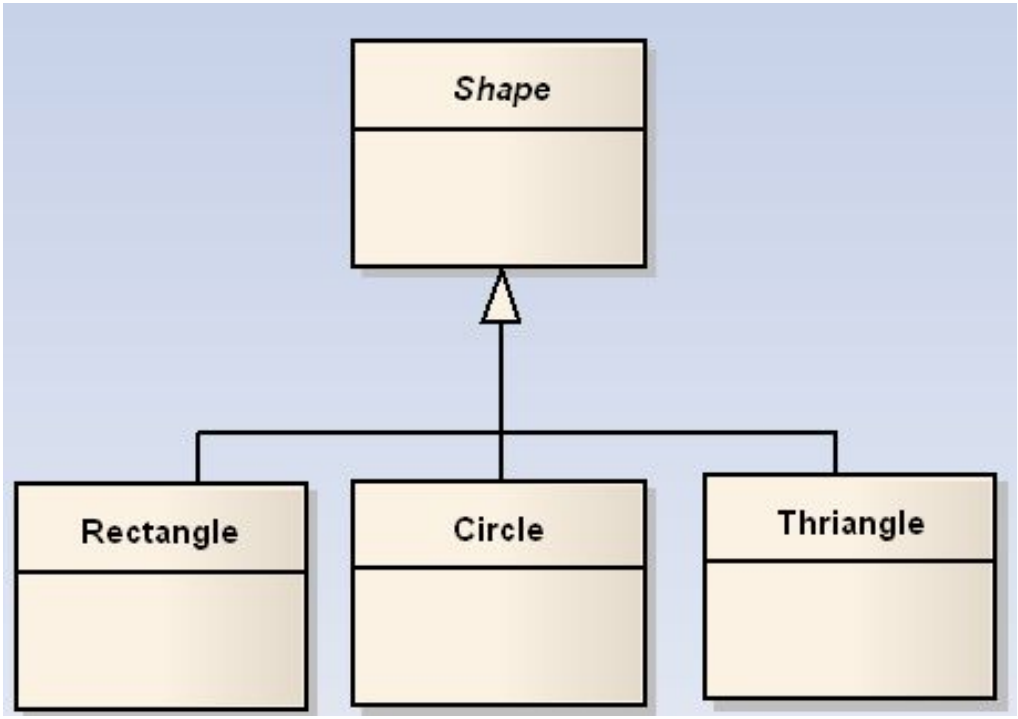
Ассоциация-класс – элемент модели, который имеет свойства как ассоциации, так и класса, и предназначенный для спецификации дополнительный свойств ассоциации в форме атрибутов и, возможно, операций классов.



Обобщение

Обобщение – таксономическое отношение между более общим классификатором (родителем или предком) и более специальным классификатором (дочерним или потомком)





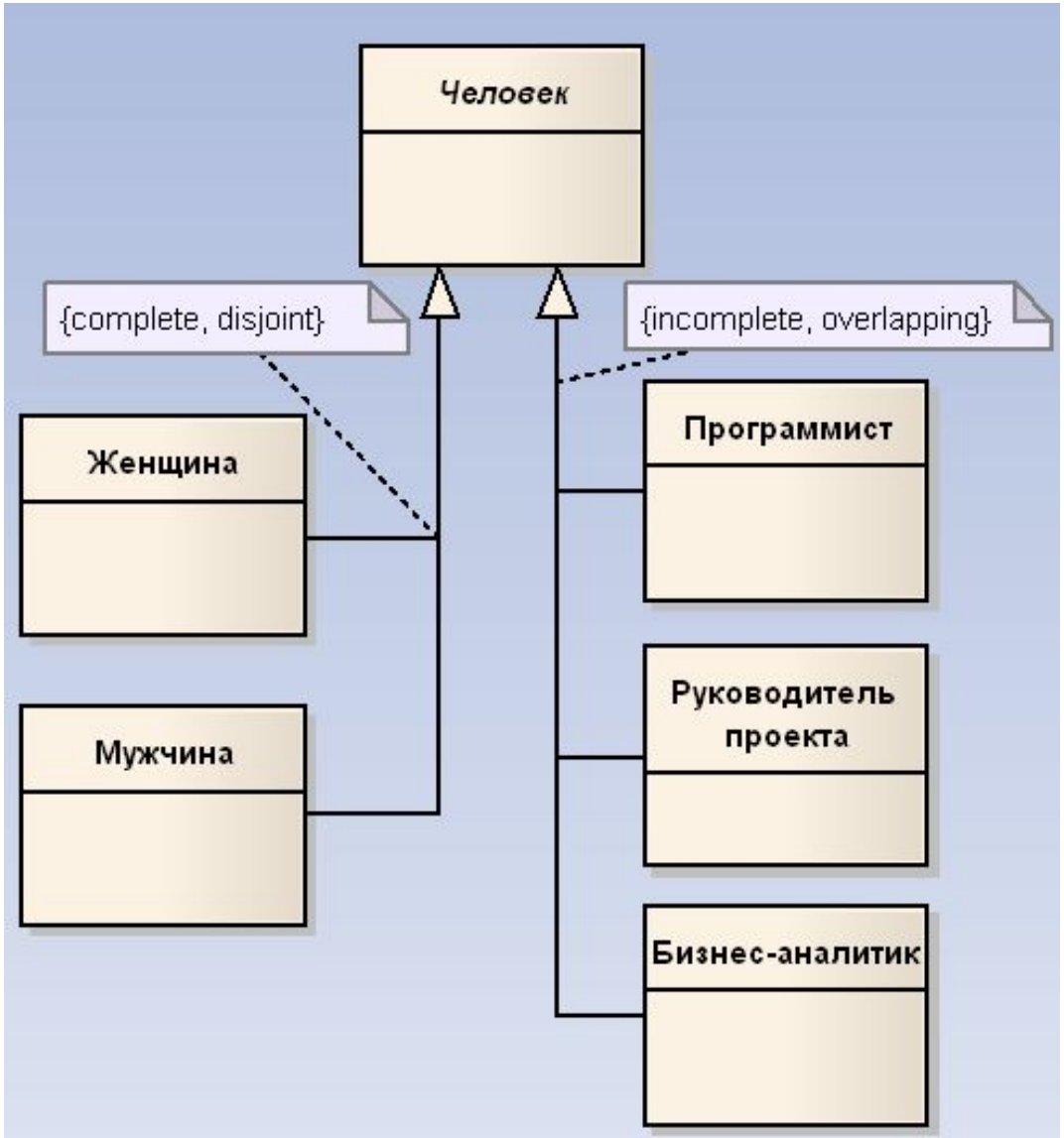
Обобщение

Множество обобщения – элемент модели, экземпляры которого определяют коллекции подмножеств отношения обобщения.

Ограничения множества обобщений:

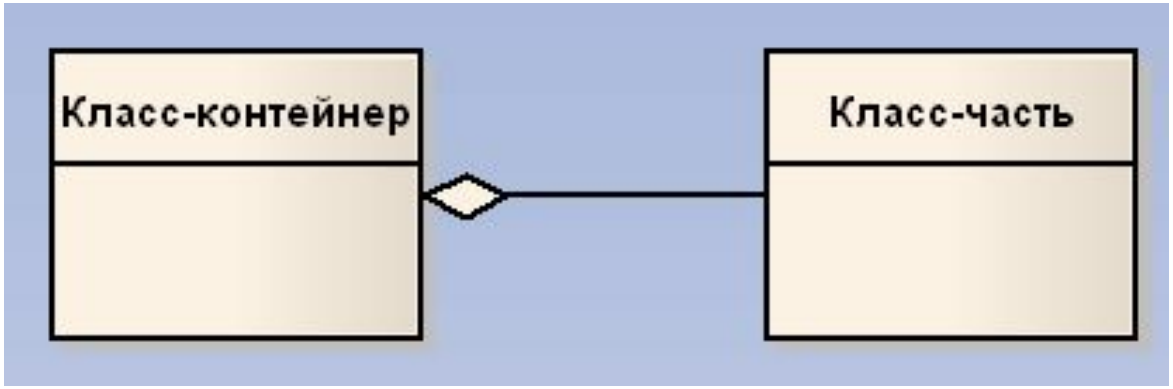
- {complete, disjoint} – данное множество обобщения является покрывающим и его специальные классы не имеют общих классификаторов.
- {incomplete, disjoint} – данное множество обобщения не является покрывающим и его специальные классы не имеют общих экземпляров.
- {complete, overlapping} – данное множество обобщения является покрывающим и его специальные классы имеют общие экземпляры.
- {incomplete, overlapping} – данное множество обобщения не является покрывающим и его специальные классы имеют общие экземпляры.

Обобщение

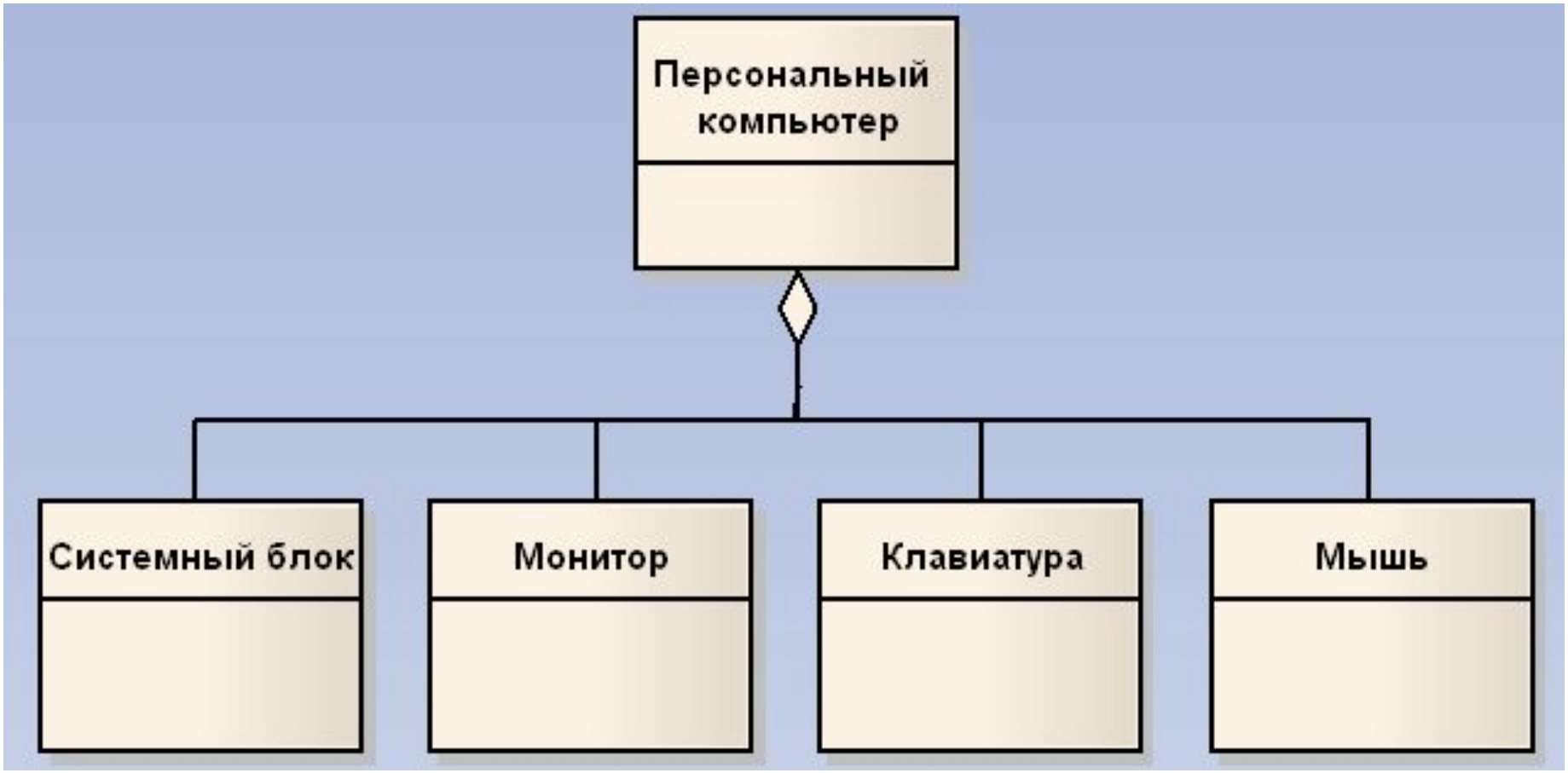

Человек
{complete, disjoint}
{incomplete, overlapping}
Женщина
Мужчина
Программист
Руководитель проекта
Бизнес-аналитик

Агрегация

Агрегация – направленное отношение между двумя классами, предназначенное для представления ситуации, когда один из классов представляет собой некоторую сущность, которая включает в себя в качестве составных частей другие сущности.

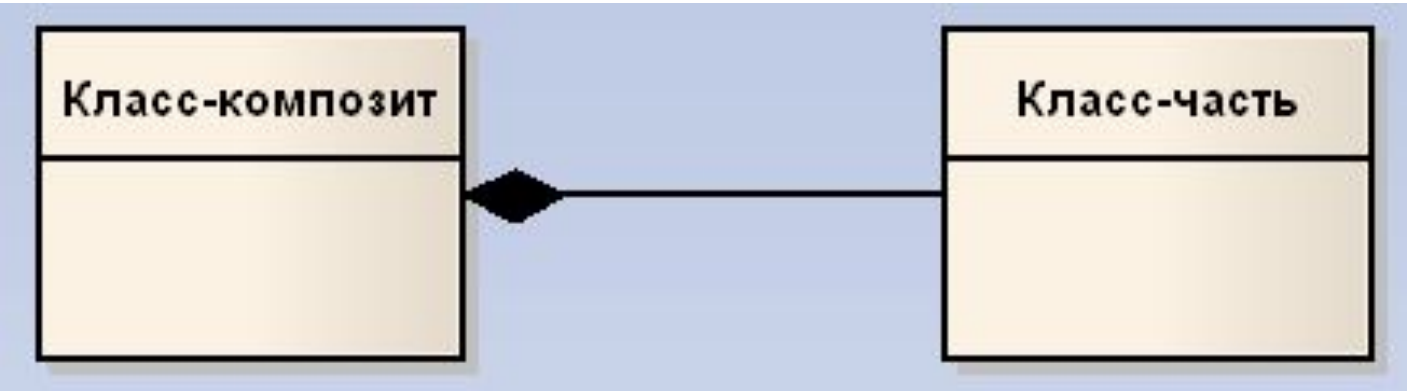


Агрегация

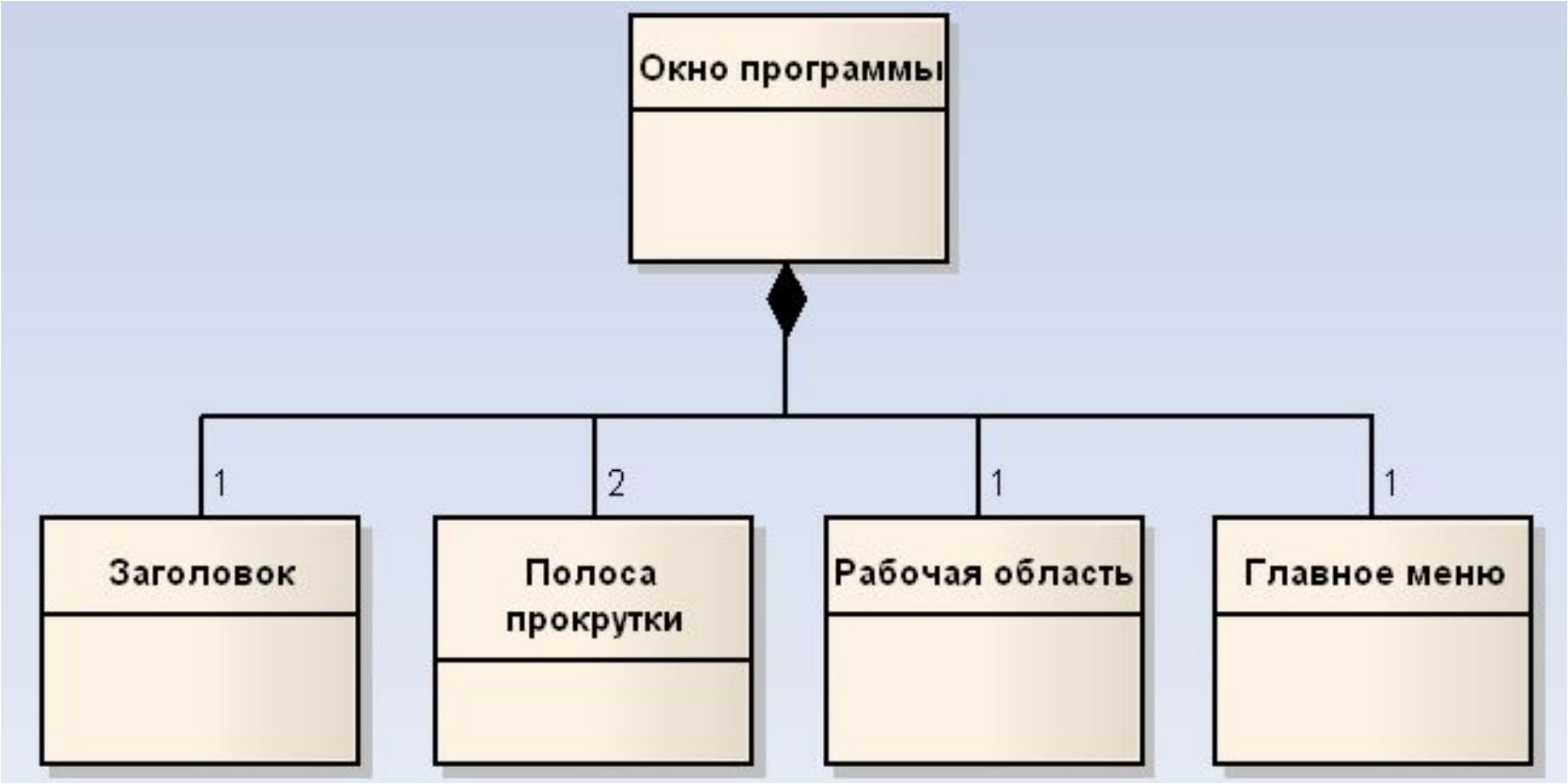

Персональный компьютер
Системный блок
Монитор
Клавиатура
Мышь

Композиция

Композиция (или композитная агрегация) предназначена для спецификации более сильной формы отношения «часть-целое», при которой с уничтожением объекта класса-контейнера уничтожаются и все объекты, являющиеся его составными частями.

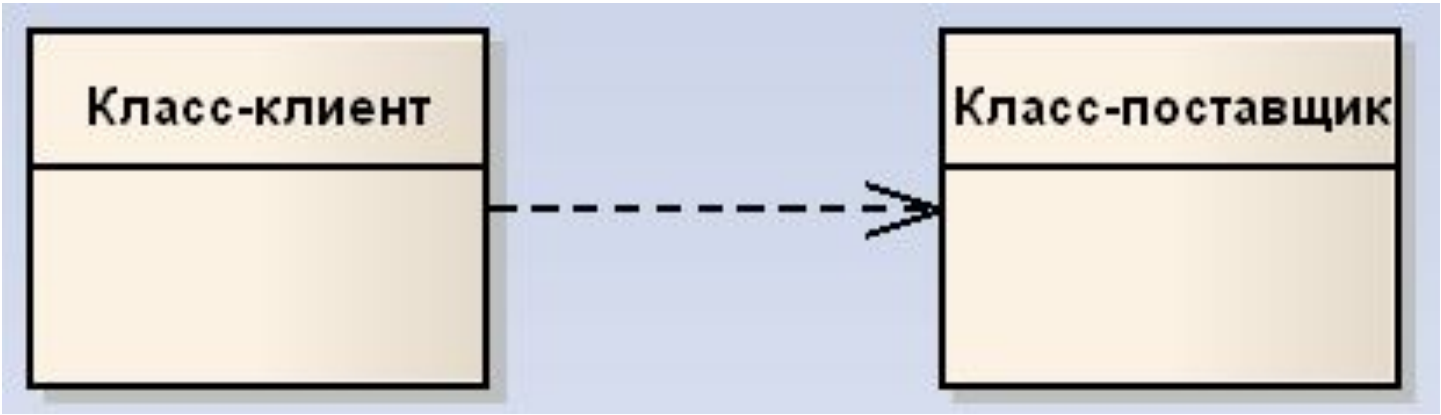


Композиция


Окно программы
1
2
1
1
Заголовок
Полоса прокрутки
Рабочая область
Главное меню

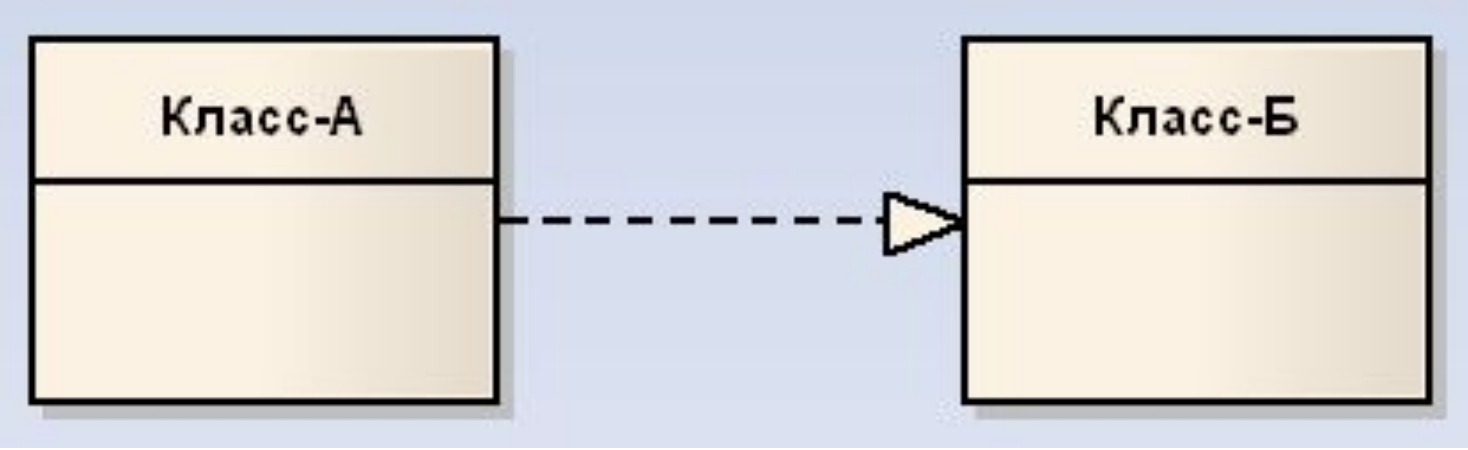
Зависимость

Зависимость – отношение, предназначенное для описания ситуации, когда отдельному элементу или множеству элементов модели требуются другие элементы модели для своей спецификации или реализации.



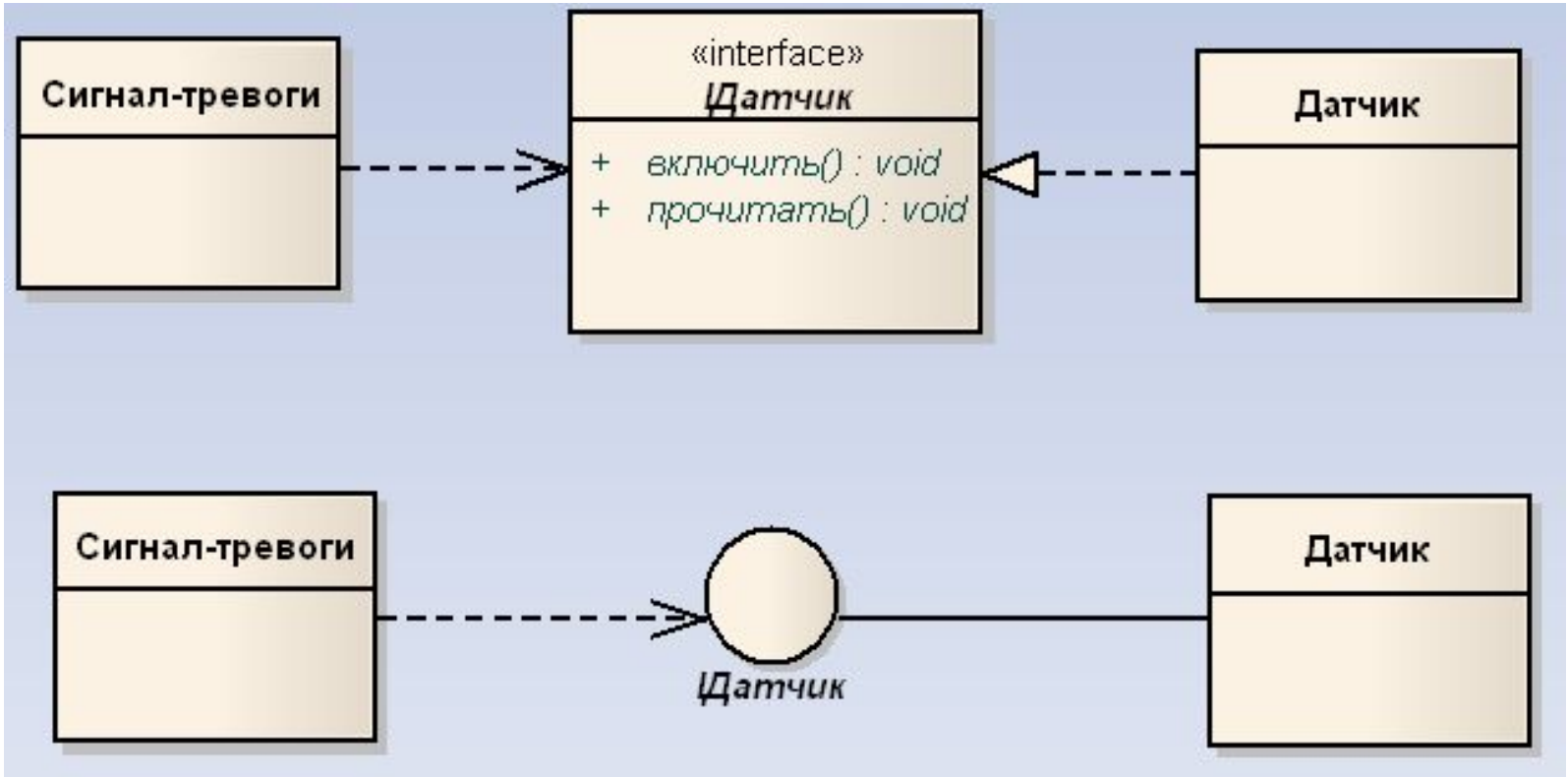
Реализация

Реализация – специализированное отношение зависимости между двумя элементами модели, один из которых представляет некоторую спецификацию (поставщик), а другой представляет его реализацию (клиент).



Интерфейс

Интерфейс – вид класса, который представляет собой объявление общедоступных характеристик и обязанностей.

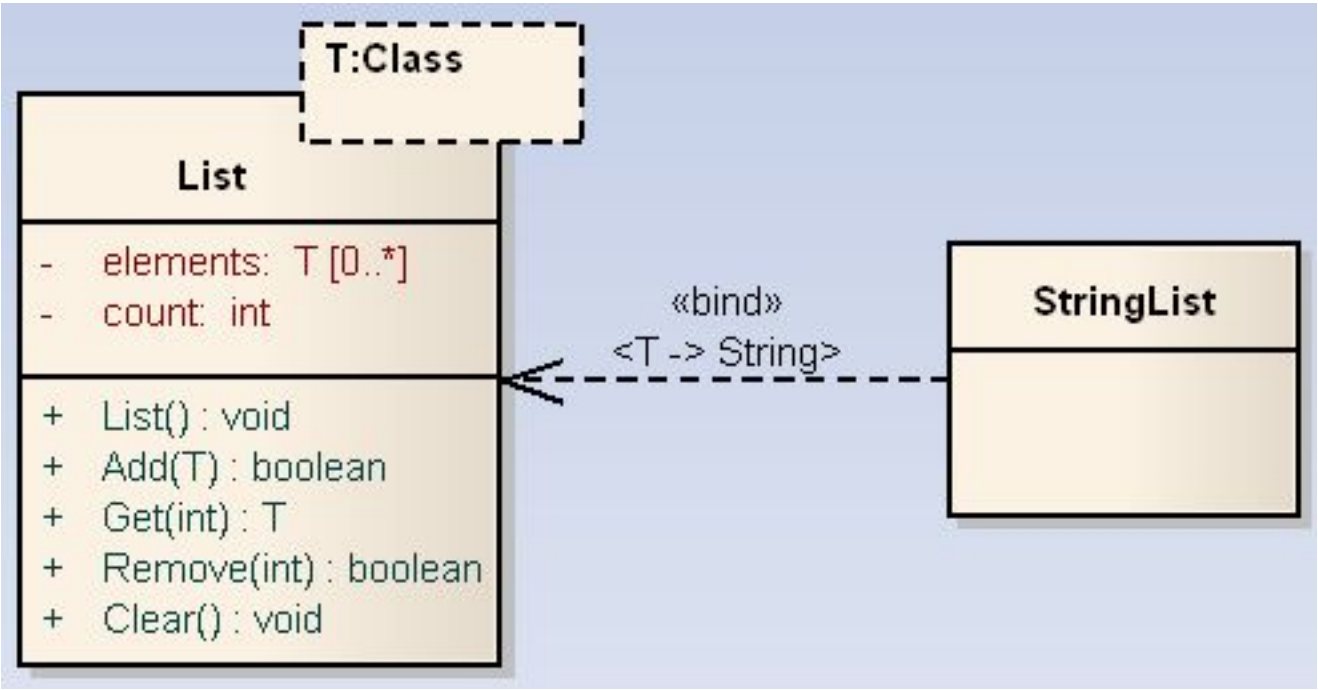


Шаблон

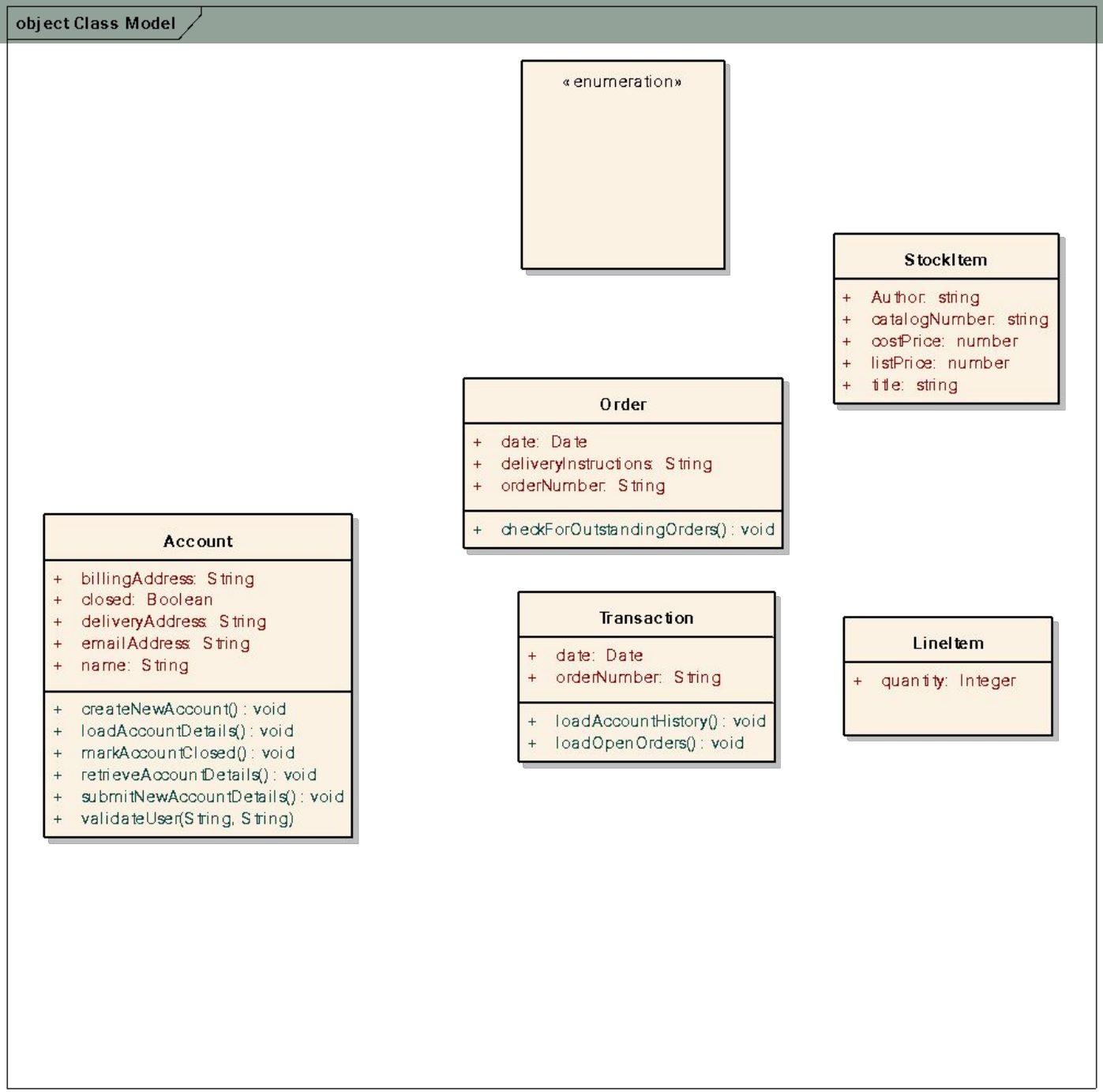
Шаблон – классификатор, который в своем описании имеет несколько формальных параметров.

<параметр-шаблона> ::=
<имя-параметра-шаболна> [‘:’ <вид-параметра>] [‘=’ <по-умолчанию>]

<подстановка-параметар-шаблона> ::=
<имя-параметра-шаблона> ‘->’ <действительный-параметр-шаблона>



Примеры


object Class Model
«enumeration»
StockItem
+ Author: string
+ catalogNumber: string
+ costPrice: number
+ listPrice: number
+ title: string
Order
+ date: Date
+ deliveryInstructions: String
+ orderNumber: String
+ checkForOutstandingOrders() : void
Account
+ billingAddress: String
+ closed: Boolean
+ deliveryAddress: String
+ emailAddress: String
+ name: String
+ createNewAccount() : void
+ loadAccountDetails() : void
+ markAccountClosed() : void
+ retrieveAccountDetails() : void
+ submitNewAccountDetails() : void
+ validateUser(String, String)
Transaction
+ date: Date
+ orderNumber: String
+ loadAccountHistory() : void
+ loadOpenOrders() : void
LineItem
+ quantity: Integer