7. Классы и отношения между ними

7.1. Классы

- ***Класс*** это реализация типа объектов, т.е. *класс* определяет множество объектов и множество операций, допустимых над этими объектами.
- Объекты, принадлежащие классу, характеризуются структурными свойствами и поведением.
- ***Структурные свойства объектов*** класса задаются атрибутами или данными-членами класса и определяют, какой информацией владеют (хранят) объекты.
- ***Поведение объектов*** класса задается методами или функциями-членами класса и определяет, что объекты могут делать.

Правила именования классов

- При моделировании системы следует использовать следующие правила именования классов:
  - имя класса должно быть существительным или оборотом существительного;
  - имя класса должно отображать роль, которую класс исполняет в системе;
  - имя класса должно выбираться из глоссария системы;
  - имя класса должно начинаться с прописной (большой) буквы;
  - если имя класса содержит несколько слов, то каждое из них должно писаться с большой буквы.

7.2 Спецификация класса

- ***Спецификация класса*** представляет собой описание класса, которое позволяет генерировать его интерфейс и поясняет использование объектов класса.

Спецификация интерфейса

- ***Name*** – имя класса;
- ***Language*** – язык программирования, на котором будет генерироваться интерфейс класса;
- ***Type*** – тип класса, который может принимать одно из следующих значений:
  - ***Class*** – обычный класс;
  - ***Parameterized Class*** – шаблон класса;
  - ***Instantiated Class*** – установленный шаблон класса;
  - ***Class Utility*** – набор функций для обеспечения дополнительной функциональности;
  - ***Parameterized Class Utility*** – шаблон набора функций;
  - ***Instantiated Class Utility*** – установленный шаблон набора функций;
  - ***Metaclass*** – класс, экземплярами которого являются другие классы;
- ***Export Control*** (Visibility) – специфицирует видимость класса вне пакета, может принимать одно из следующих значений:
  - ***Public*** – видим вне пакета;
  - ***Protected*** – видим вне пакета только друзьями класса и вложенными классами;
  - ***Private*** – видим вне пакета только друзьями класса;
  - Implementation (Package) – видим только в пакете;
- ***Attributes*** – атрибуты класса;
- ***Operations*** – операции класса.

Спецификация объектов

- ***Multiplicity*** (Cardinality) – определяет количество ожидаемых экземпляров (объектов) класса, может принимать одно из следующих значений:
  - n – неограниченное количество объектов;
  - 0..0 – ни одного объекта;
  - 0..1 – ни одного или один объект;
  - 0..n – ни одного или более объектов;
  - 1..1 – один объект;
  - 1..n – один или более объектов;
  - <literal> – точное количество объектов, например, <5>;
  - <literal>..n – от literal до n объектов;
  - <literal>..<literal> – диапазон объектов;
  - <literal>..<literal>, <literal> – диапазон или точное количество объектов;
  - <literal>..<literal>, <literal>..<literal> – один из диапазонов объектов;
- ***Space*** – объем памяти, требуемый объектом класса;

- ***Persistence*** – определяет время существования объектов класса, может принимать одно из следующих значений:
  - ***Persistent*** – объекты класса существуют больше, чем исполняется приложение, т.е. информация, хранимая в объектах, сохраняется в постоянной памяти, например, в базе данных;
  - ***Transient*** – объекты класса существуют только во время исполнения приложения;
- ***Concurrency*** – описывает поведение класса в многопоточном приложении, может принимать одно из следующих значений:
  - ***Sequential*** – объекты класса могут использоваться только в однопоточном приложении;
  - ***Guarded*** – объекты класса могут использоваться в многопоточных приложениях, но необходимо обеспечить взаимное исключение при доступе к объекту;
  - ***Active*** – класс имеет свой поток управления;
  - ***Synchronous*** – объекты класса могут использоваться в многопоточных приложениях, взаимное исключение к объектам класса обеспечивается самим классом;
- ***Protocol*** – описание ограничений на порядок вызова операций класса, просто текст, не влияет на генерацию кода.

7.3. Атрибуты класса

- ***Структурные свойства*** объектов класса задаются ***атрибутами*** или ***данными-членами*** класса.
- Каждый атрибут класса имеет свой тип.
- Класс может иметь произвольное количество атрибутов.
- Допускаются классы без атрибутов.
- Класс, который содержит только атрибуты, обычно называется ***структурой***.

Именование атрибутов

- При проектировании классов следует использовать следующие правила именования атрибутов класса:
  - имя атрибута должно быть существительным или оборотом существительного и отображать принадлежность свойства классу;
  - имена атрибутов должны писаться со строчной (малой) буквы;
  - если имя атрибута содержит несколько слов, то каждое из них за исключением первого должно писаться с большой буквы.

7.4. Спецификация атрибутов класса

- Спецификация атрибута класса представляет собой его точное описание, которое позволяет генерировать интерфейс этого класса.
- В спецификацию атрибута включаются следующие пункты:

- ***Name*** – имя атрибута;
- ***Type*** – тип атрибута; не имеет значения по умолчанию; может принимать значения предопределенных типов языка программирования или имен классов;
- ***Visibility*** (Scope) – область видимости атрибута, может принимать одно из следующих значений:
  - ***Public***;
  - ***Protected***;
  - ***Private***;
- ***Multiplicity*** – количество значений, которые может принимать атрибут; задается в виде:
  - lower_bound..upper_bound, где ‘*’ отмечает бесконечность;
- ***Ordering*** – если атрибут может хранить более одного значения, то обозначает упорядоченность этих значений; может принимать одно из значений:
  - ***unordered*** – неупорядочено (используется по умолчанию);
  - ***ordered*** – упорядочено;

- ***Initial value*** – начальное значение атрибута; не имеет значения по умолчанию;
- ***Containment*** – способ физического хранения атрибута, может принимать одно из значений:
  - ***By Value*** – хранение по значению;
  - ***By Reference*** – хранение через ссылку или указатель;
  - ***Unspecified*** – не определено (используется по ***умолчанию);***
- ***Static*** – статический атрибут;
- ***Derived*** – вычисляемый атрибут;
- ***Const*** – атрибут константа;
- ***Constraints*** – ограничения на атрибут.

7.5. Операции класса

- Поведение объекта класса характеризуется ***операциями***, которые может выполнять объект.
- Реализация операции называется ***методом***.

Взаимодействие объектов

- ***Взаимодействием*** двух объектов называется посылка одним объектом сообщения другому объекту.
- Посылка сообщения выполняется посредством вызова метода объекта, которому передается сообщение.
- Взаимодействие двух объектов это ***динамическая связь*** или ***отношение*** между этими объектами.
- При взаимодействии двух объектов объект, методы которого вызываются, называется ***сервером***, а объект, который вызывает методы сервера, называется ***клиентом***.

Роли объектов при взаимодействии

- При взаимодействии объекты классифицируются следующим образом:
  - ***актер*** – это объект, который может воздействовать на другие объекты, но сам никогда не подвергается воздействию других объектов;
  - ***сервер*** – это объект, который может только подвергаться воздействию со стороны других объектов, но он никогда не выступает в роли воздействующего объекта;
  - ***агент*** – это объект, который может выступать и как актер, и как сервер.

Правила именования операций

1. Имя операции должно быть глаголом или оборотом глагола и отображать действие, которое можно выполнять над объектом класса.
2. Имена операций должны писаться со строчной (малой) буквы.
3. Если имя операции содержит несколько слов, то каждое из них за исключением первого должно писаться с большой буквы.

7.6. Спецификация операций класса

- Спецификация операции класса представляет собой описание её сигнатуры, ограничений на вызов и семантики, которое позволяет:
  - генерировать интерфейс класса, которому принадлежит операция;
  - правильно вызывать операцию, учитывая контекст окружения;
  - реализовать операцию.

Спецификация интерфейса операции

- ***Name*** – имя операции;
- ***Visibility*** – область видимости операции; может принимать одно из следующих значений:
  - ***Public***;
  - ***Private***;
  - ***Protected***;
  - ***Implementation***;
- ***Parameters*** – список параметров операции;
- ***Return type*** – тип возвращаемого методом значения;
- ***Qualifications*** – свойства операций, зависимые от языков программирования;
- ***Exceptions*** – типы исключений, которые может генерировать операция.

Спецификация для реализации и использования операции

- ***Size*** – объем памяти, необходимый для выполнения операции;
- ***Time*** – время, необходимое для выполнения операции;
- ***Concurrency*** – возможность вызова операции в многопоточном приложении, может принимать одно из значений:
  - ***Sequential*** – может вызываться только в однопоточном приложении;
  - ***Guarded*** – может вызываться в многопоточном приложении, но нужно обеспечить взаимное исключение при вызове операции;
  - ***Synchronous*** – может вызываться в многопоточных приложениях;
- ***Semantics*** (Behavior) – семантика, смысл операции, грубо говоря, алгоритм исполнения операции;
- ***Preconditions*** – предусловия;
- ***Post conditions*** – постусловия.

7.7. Отношение ассоциации между классами

- ***Отношение ассоциации*** – это наиболее общее отношение между классами, которое показывает наличие некоторой семантической (смысловой) связи между классами.
- Отношению ассоциации следует давать имя, которое описывает эту ассоциацию.
- ***Association class*** – это класс, который связывается отношением зависимости с отношением ассоциации и показывает, что он является реализацией этой ассоциации.

Специфические типы ассоциации

- Отношение “***is-a***” – определяет отношение ***наследования*** (inheritance) или ***обобщения*** (generalization) между классами.
- Это отношение показывает, что объекты производного класса наследуют свойства и методы объектов базового класса или что объекты производного класса также являются объектами базового класса.
- Отношение “***has-a***” – отношение ***включения*** между классами.
- Это отношение показывает, что объекты одного класса вложены в объекты другого класса.

7.5. Отношение зависимости между классами

- ***Отношение зависимости*** показывает, что между классами существует некоторая зависимость, т. е. изменение одного класса вызывает изменение другого класса.
- Клиенты класса видят только его интерфейс, который содержит открытые члены класса. Поэтому зависимость между классами подразумевает зависимость классов-клиентов от интерфейсов классов-серверов.
- Отношению зависимости следует давать имя, которое описывает эту зависимость.

7.6. Спецификация отношений между классами

- Спецификация отношения между классами поясняет ***статическую связь*** между классами или ***динамические взаимодействие*** объектов классов.
- Можно сказать, что специфицируя отношения между классами, мы превращаем логическую структуру системы в логическую архитектуру.

- Спецификация отношения между классами включает следующие пункты:
  - Navigable – направление отношения;
  - Role – имя, которое определяет роли классов в ассоциации; роли имеют следующие спецификации:
    - ***Name*** – имя;
    - ***Constraints*** – ограничения на роль;
    - ***Multiplicity*** – количество экземпляров роли;
    - ***Aggregate*** – показывает, что роль является агрегатом (входит в) другой роли;
    - ***Static*** – показывает, что роль является статической;
    - ***Friend*** – роль является другом;
    - ***Containment*** – показывает, как хранится роль, может принимать значения:
      - ***By Value*** – хранение по значению;
      - ***By Reference*** – хранение через ссылку или указатель;
      - ***Unspecified*** – не определенно (используется по умолчанию);
  - ***Key*** – используется для ограничения отношения на член класса;
  - ***Qualifier*** – устанавливает вид связи между объектами в формате 1..n или n..n.