2.3. Основные принципы объектной модели

Объектно-ориентированная технология основывается на объектной модели (ОМ). Ее основными принципами являются: абстрагирование, инкапсуляция, модульность, иерархия, типизация, параллелизм, сохраняемость. Первые четыре элемента являются главными, потому что без любого из них модель не будет объектно-ориентированной. Все принципы не новы, но в объектно-ориентированной модели они впервые применены в совокупности. Подробнее рассмотрим каждый из перечисленных принципов.

2.3.1. Абстрагирование

Абстракция выделяет существенные характеристики некоторого объекта, отличающие его от всех других объектов и, таким образом, четко определяет его концептуальные границы с точки зрения наблюдателя.

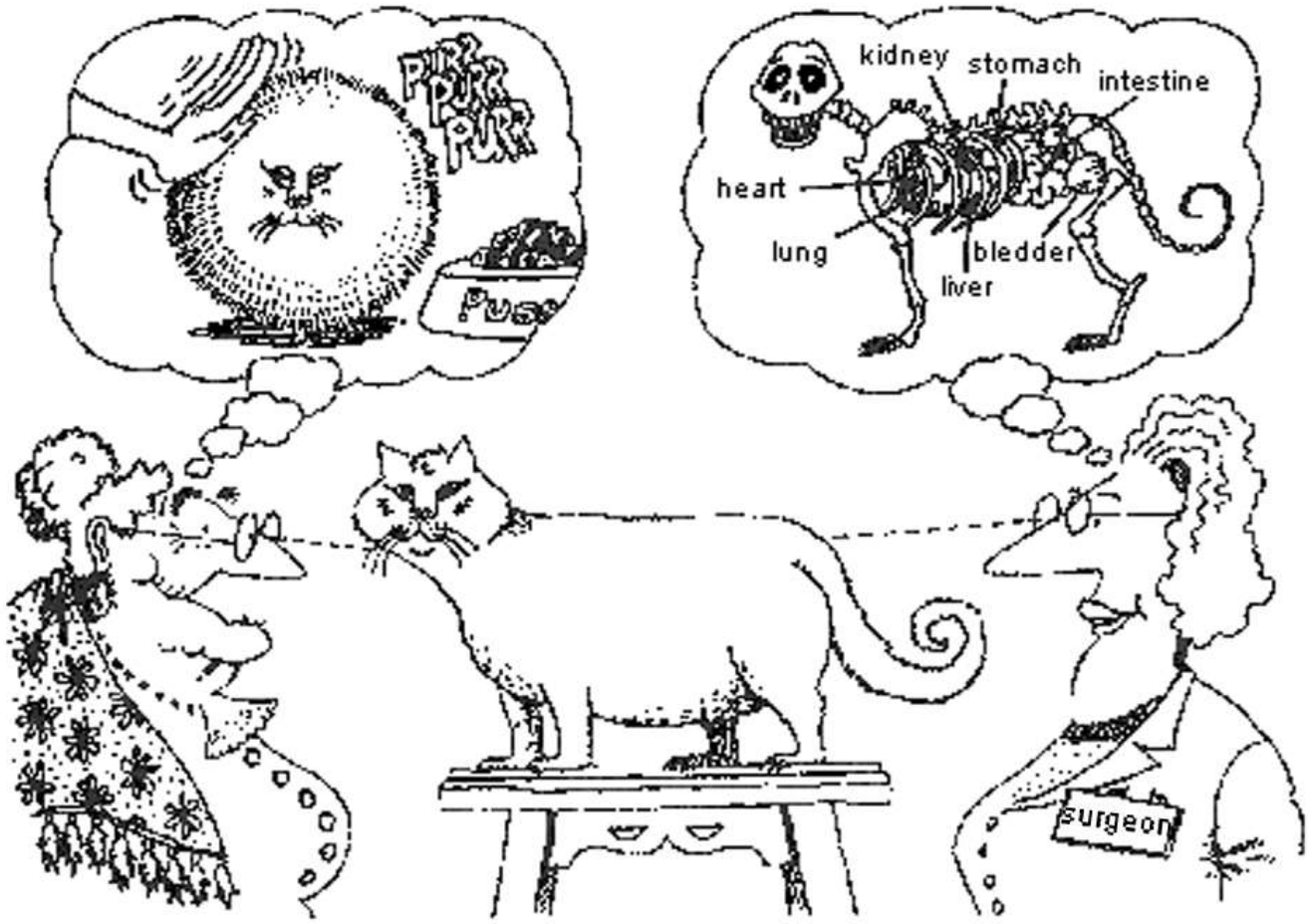

Рис. 2.5. Абстракция фокусируется на существенных с точки зрения наблюдателя характеристиках объекта

2.3.2. Инкапсуляция

Абстракция и инкапсуляция дополняют друг друга. Абстрагирование направлено на (наблюдаемое) поведение объектов, а инкапсуляция занимается (внутренним) устройством. Чаще всего инкапсуляция выполняется посредством скрытия информации всех внутренних деталей, не влияющих на внешнее поведение. Обычно скрывается и внутренняя структура объекта, и реализация его методов. Таким образом, инкапсуляция скрывает детали реализации объекта.

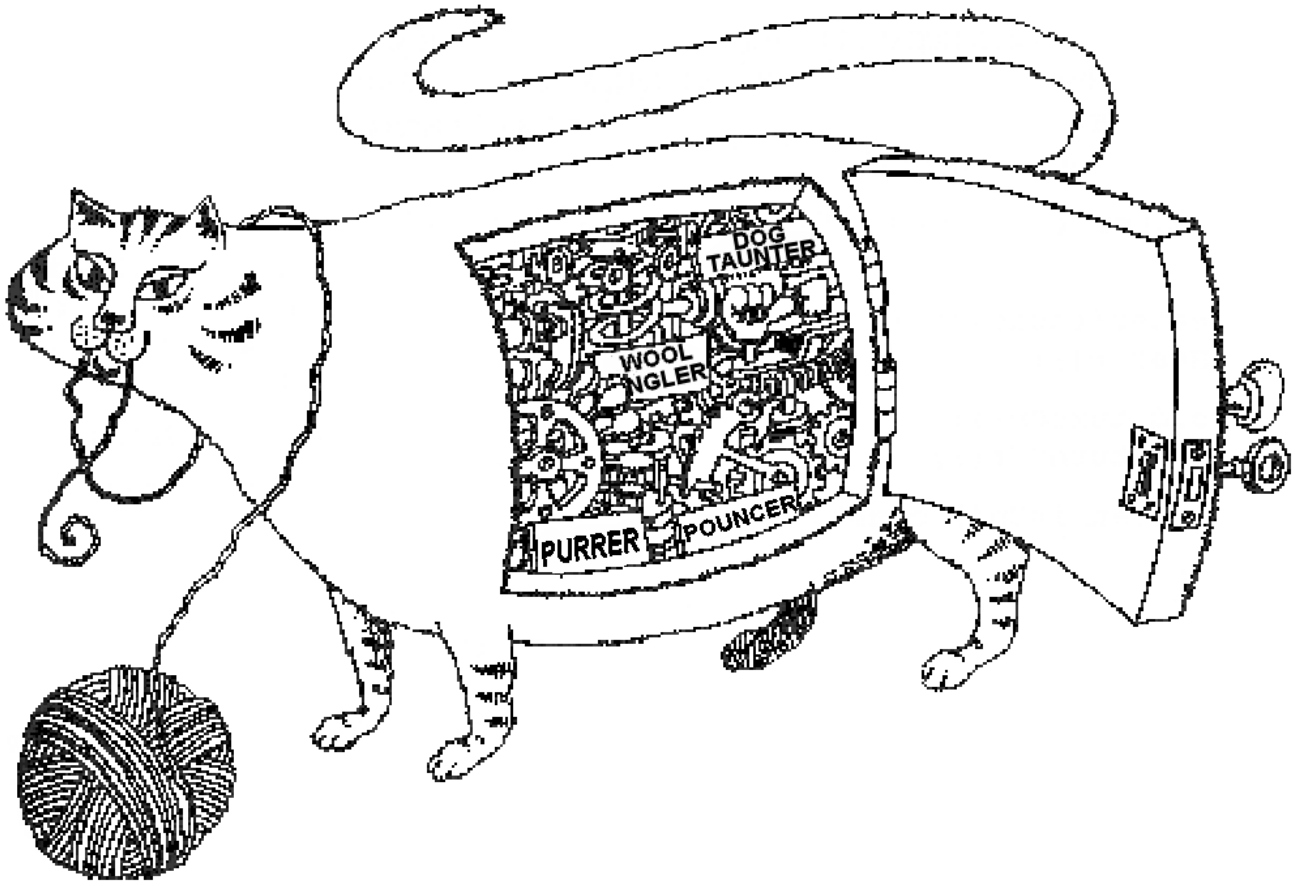

Рис. 2.6. Инкапсуляция скрывает детали реализации объекта

Практически совместная работа абстракции и инкапсуляции означает, что в классе существует две части - интерфейс и реализация. Интерфейс отражает внешнее поведение объекта, описывая абстракцию поведения всех объектов данного класса. Внутренняя реализация описывает представление этой абстракции и механизмы достижения желаемого поведения объекта. В интерфейсной части собрано все, что касается взаимодействия данного объекта с любыми другими объектами; реализация скрывает от

других объектов все детали, не имеющие отношения к процессу взаимодействия объектов.

2.3.3. Модульность

В традиционном структурном проектировании под ***модульным принципом*** понимается разделение программы на части (модули). Для уменьшения сложности программы стремятся чтобы модули были небольшими и в высокой степени независимыми. Независимость модулей достигается двумя методами оптимизации: усилением связей внутри модуля и ослаблением взаимосвязи между модулями. Таким образом, системы должны иметь внутренние связанные, но слабо сцепленные между собой модули.

В объектно-ориентированном проектировании модульность – это искусство физически разделять классы и объекты, составляющие логическую структуру проекта. Следует стремиться построить модули так, чтобы объединить логически связанные абстракции и минимизировать взаимные связи между модулями.

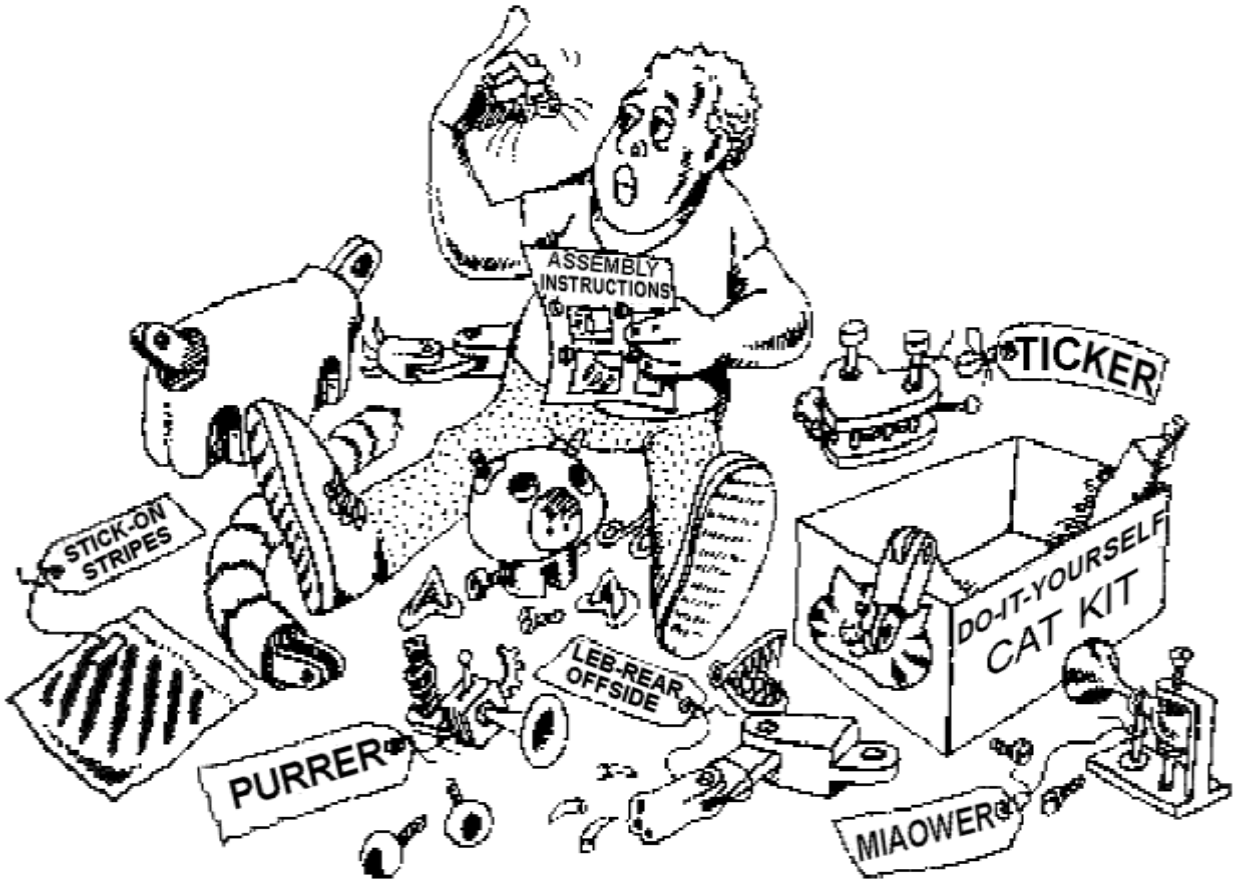

Рис. 2.7. Модульность позволяет хранить абстракции раздельно

Принципы абстрагирования, инкапсуляции и модульности являются взаимодополняющими. Объект логически определяет границы абстракции, а инкапсуляция и модульность делают их физически незыблемыми.

2.3.4. Иерархия

Абстракция - полезная вещь. Однако часто число абстракций в системе намного превышает предел, доступный нашим умственным возможностям. Инкапсуляция позволяет до какой-то степени устранить этот недостаток, убрав из поля зрения внутреннее содержание абстракций. Модульность также упрощает задачу, объединяя логически связанные абстракции в группы. Однако этого недостаточно. Значительное упрощение понимания сложной задачи достигается за счет образования из абстракций иерархической структуры.

Рис. 2.8. Абстракции образуют иерархию

Иерархия – это упорядочение абстракций, разложение их по уровням. Основными видами иерархических структур применительно к сложным системам являются структура классов (иерархия «is-a») и структура объектов (иерархия «part of»). С иерархией в объектно-ориентированной парадигме связаны такие понятия как наследование, полиморфизм и агрегация, подробнее они рассматриваются разделе «Классы».

2.3.5. Типизация

Типизация – это способ исключить использование объектов одного класса вместо другого или управление таким использованием. Понятие типа взято из теории абстрактных типов данных. Идея согласования типов занимает в типизации центральное место. Примером типизации может служить использование физических единиц измерения. Деля расстояние на время, мы ожидаем получить скорость, а не вес. Умножение температуры на силу не имеет смысла, а умножение расстояния на силу имеет смысл. Все это примеры типизации, когда прикладная область накладывает правила и ограничения на использование и сочетание абстракции.

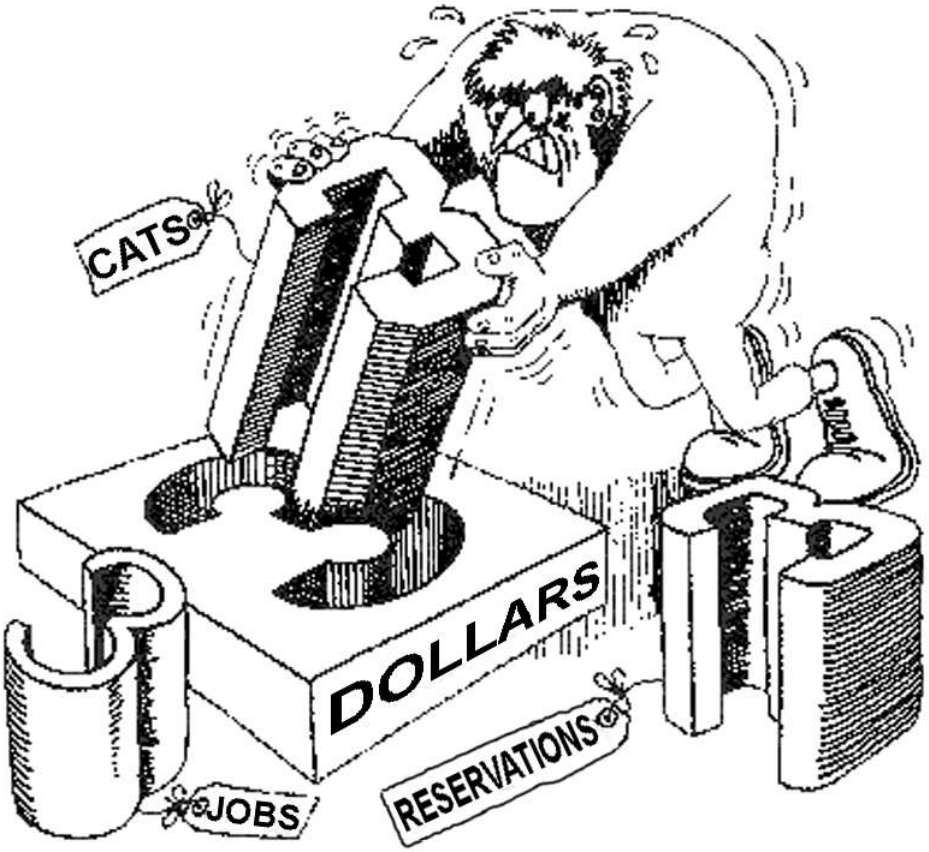

Рис. 2.9. Строгая типизация предотвращает смешивание абстракций

Строгая типизация в объектно-ориентированном проектировании предотвращает смешивание абстракций.

Языки программирования могут иметь механизмы типизации, а могут и не иметь.

2.3.6. Параллелизм

Для задач, в которых одновременно обрабатываются несколько событий или потребность в вычислительной мощности превышает ресурсы одного процессора, используют несколько компьютеров или задействуют многозадачность на многопроцессорном компьютере.

Параллельность достигается лишь на многопроцессорных системах. Системы с одним процессором только имитируют параллельность за счет алгоритмов разделения времени.

Объектно-ориентированные системы хорошо подходят для параллельной обработки. Параллелизм позволяет различным объектам действовать одновременно.

2.3.7. Сохраняемость

Любой программный объект существует в памяти и во времени. Спектр сохраняемости объектов охватывает следующий диапазон:

1) промежуточные результаты вычисления выражений;
2) локальные переменные в вызове процедур;
3) собственные переменные, глобальные переменные, динамические, создаваемые данные;
4) данные, сохраняющиеся между сеансами выполнения программ;
5) данные, которые нужно сохранять при переходе на новую версию программы;
6) данные, которые нужно сохранять даже после устаревания версии программы.

Традиционно первыми тремя уровнями занимаются языки программирования, а последними тремя - базы данных. Однако сейчас программисты используют специальные схемы для сохранения объектов в период между запусками программы, а конструкторы баз данных внедряют в свою технологию коротко живущие объекты.

Сохраняемость – это способность объектов существовать во времени, переживая породивший их процесс, и пространстве, перемещаясь из своего первоначального адресного пространства.