Вступ. Об'єктно-орієнтований підхід. Методологія об'єктно-орієнтованого аналізу

Общие понятия

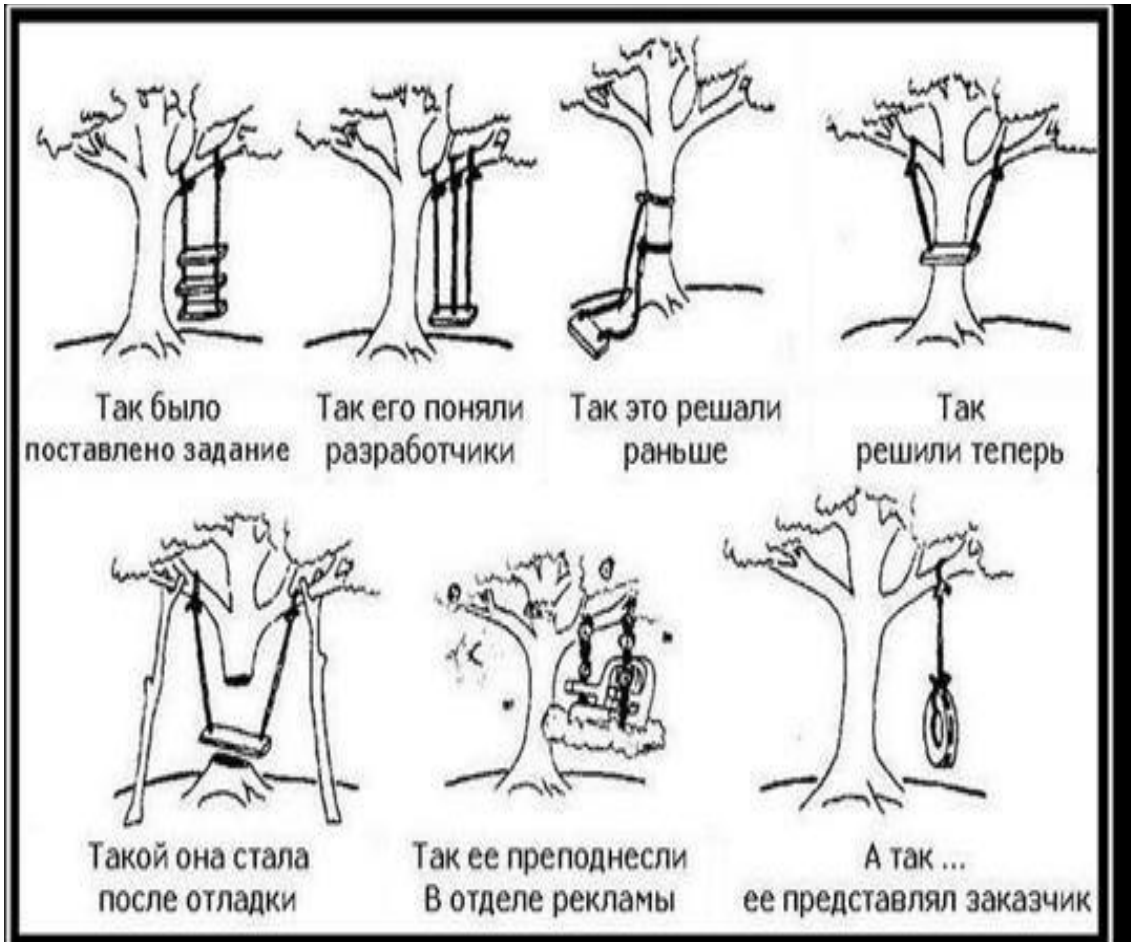
- Объектно-ориентированный анализ и проектирование (ООАП, Object-Oriented Analysis/Design) - технология разработки программных систем, в основу которых положена объектно-ориентированная методология представления предметной области в виде объектов, являющихся экземплярами соответствующих классов
- Предметная область (domain) - часть реального мира, которая имеет существенное значение или непосредственное отношение к процессу функционирования программы.
- Ахитектура системы – описание структуры классов и объектов

Простые и сложные программные системы

- «Эйнштейн утверждал, что должны существовать простые объяснения природных процессов, так как Бог не действует из каприза или по произволу. У программиста нет такого утешения: сложность, с которой он должен справиться, лежит в самой природе системы»
- Простые программные системы:
  - Создаются одним разработчиком
  - Ограничена область применения
  - Короткий период существования
  - Легче заменить чем-то новым, чем исправлять или модернизировать

Причины сложности ПО

- Сложность предметной области
  - Сложные элементы в решаемых задачах
  - Много требований к ПО(м.б. взаимоисключающих или неявно формируемых, напр., удобство, надежность)
  - У пользователя и разработчика разные «взгляды» на проблему/задачу (видео)
  - Изменение требований в процессе разработки



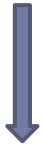
Причины сложности ПО

- Трудность управления процессом разработки
  - Основная задача разработчиков - создать иллюзию простоты
  - Исходный код д.б. компактным
  - Многомодульность систем ⟹ сложность коллективной разработки

Больше разработчиков ⟹ сложнее связи между ними ⟹ сложнее координация их работы

Причины сложности ПО

- Необходимость обеспечить достаточную степень гибкости
  - Разработчик может сам обеспечить все необходимые модули, относящимися к любому уровню абстракции



нет «стандартов» на схожие программные элементы

Причины сложности ПО

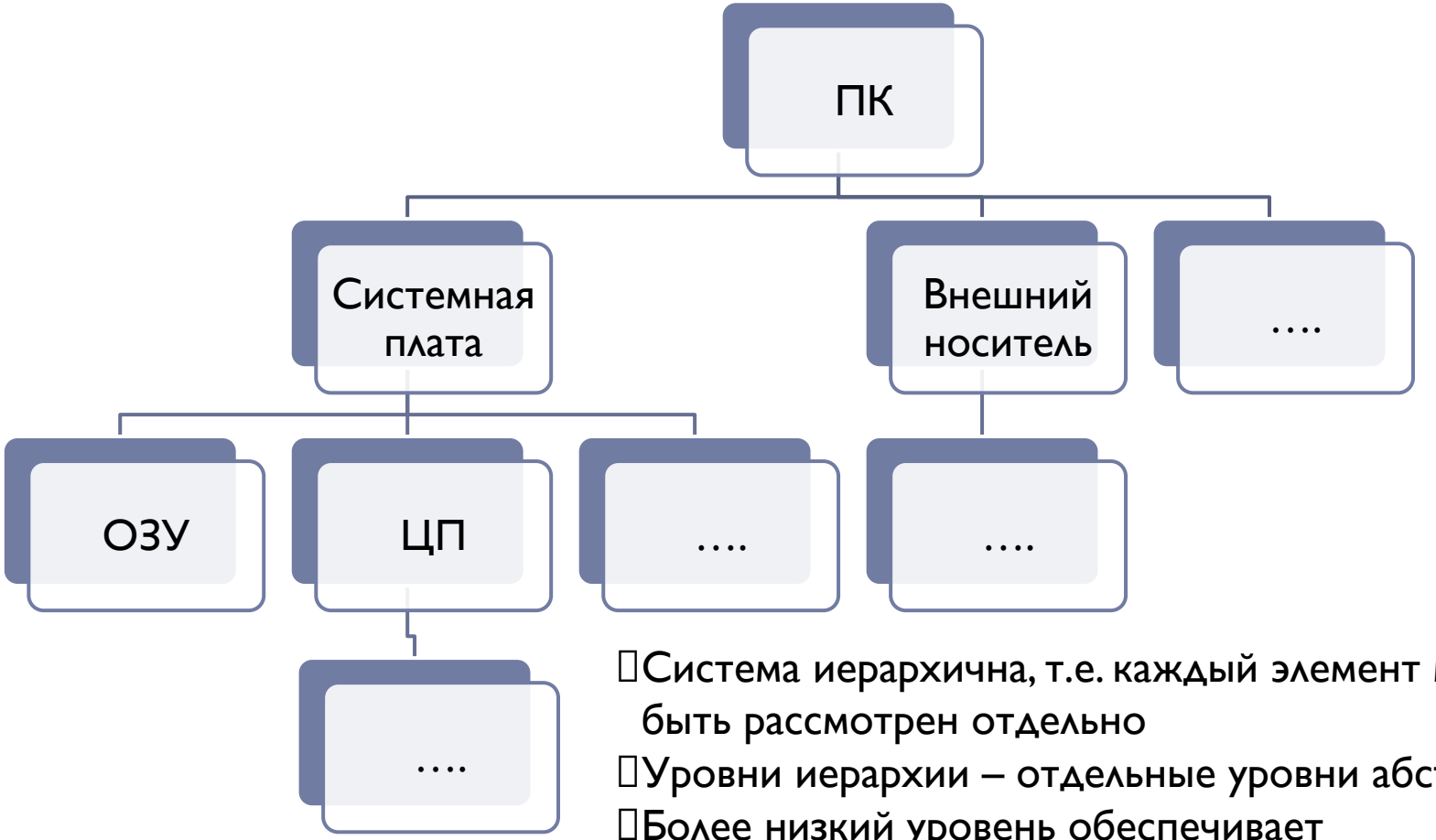
- Неудовлетворительные способы описания поведения больших дискетных систем
  - В реальном мире действуют строгие законы физики

  поведение физической системы → можно предугадать

  Компьютерная модель такой системы (дискретная) не подчинена законам физики
  - Наличие большого числа элементо/модулей и дискретных связей между ними → система легко подвергается внешнему «вторжению» в процессы

«Чем сложнее система, тем легче ее полностью развалить»

Пример системы средней сложности



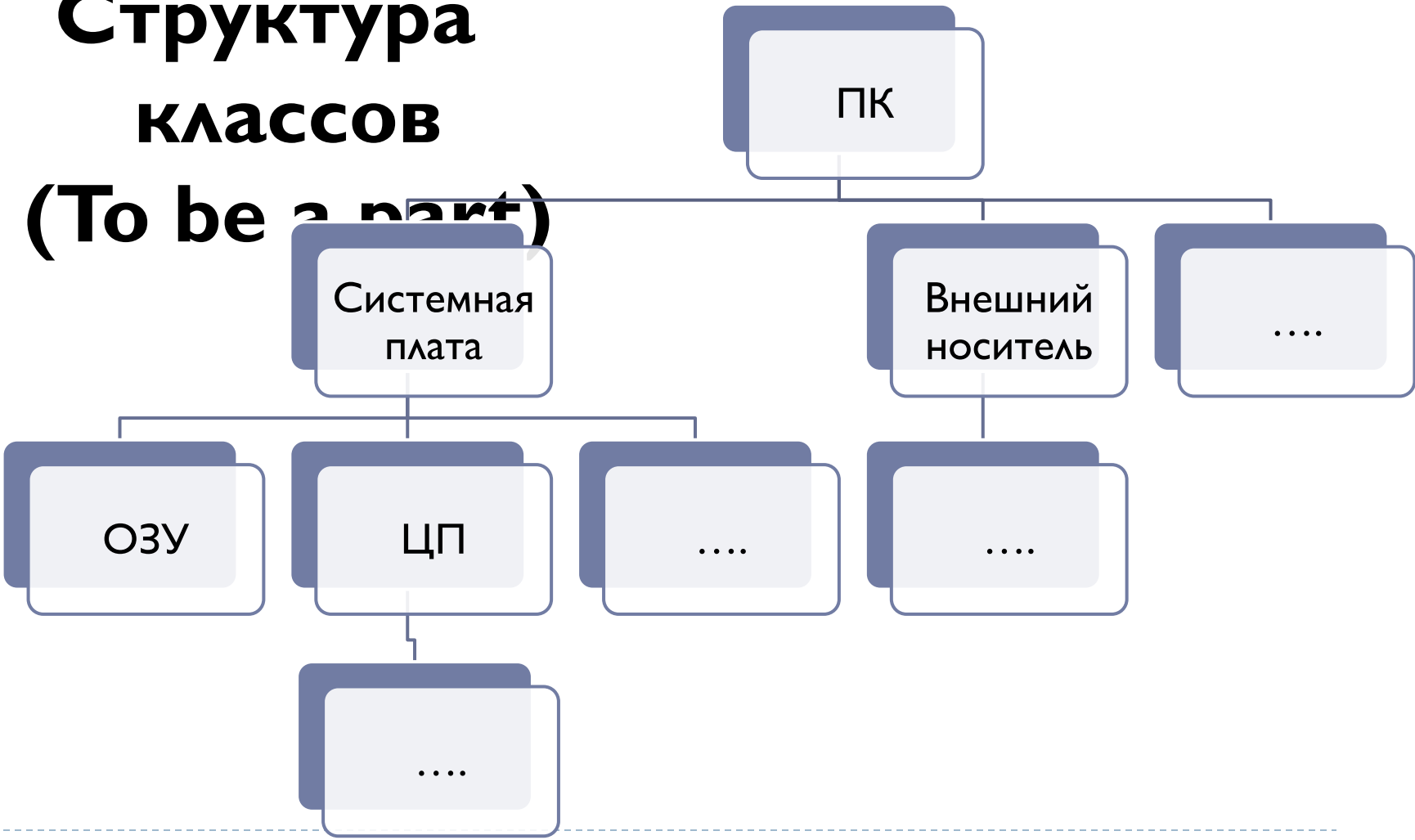
- Система иерархична, т.е. каждый элемент может быть рассмотрен отдельно
- Уровни иерархии – отдельные уровни абстракции
- Более низкий уровень обеспечивает функционирование более высокого уровня

Признаки сложных систем

- Сложные системы часто являются иерархическими
  - архитектура сложных систем складывается и из компонентов, и из иерархических отношений этих компонентов.
- Выбор, какие компоненты в системе считаются элементарными, оставляется на усмотрение исследователя.
- Внутрикомпонентная связь обычно сильнее, чем связь между компонентами.
- Иерархические системы обычно состоят из немногих типов подсистем, по-разному скомбинированных и организованных
  - разные сложные системы содержат одинаковые структурные части
- «*Любая работающая сложная система является результатом развития работавшей более простой системы... Сложная система, спроектированная «с нуля», никогда не заработает. Следует начинать с работающей простой системы*»

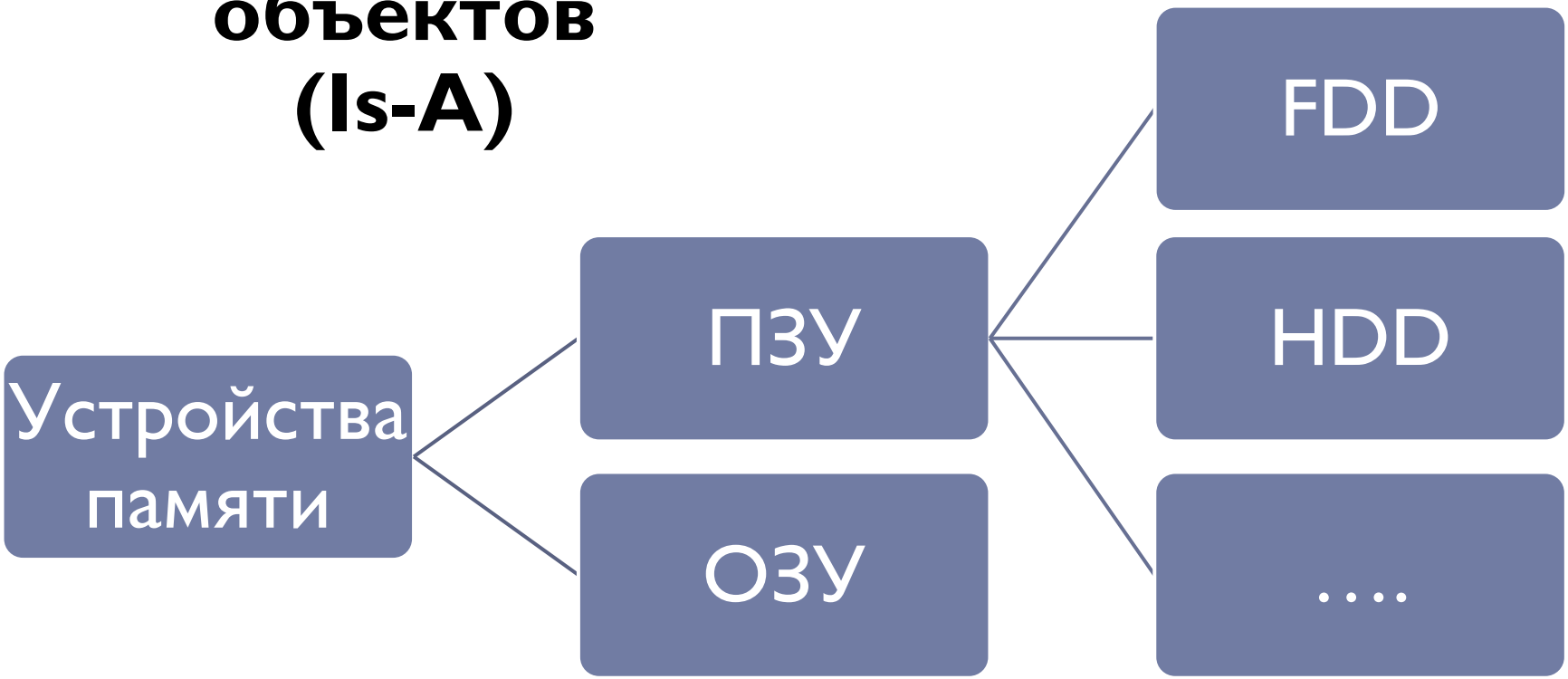
Формы сложной системы

Структура классов (To be a part)



Формы сложной системы

Структура объектов (Is-A)

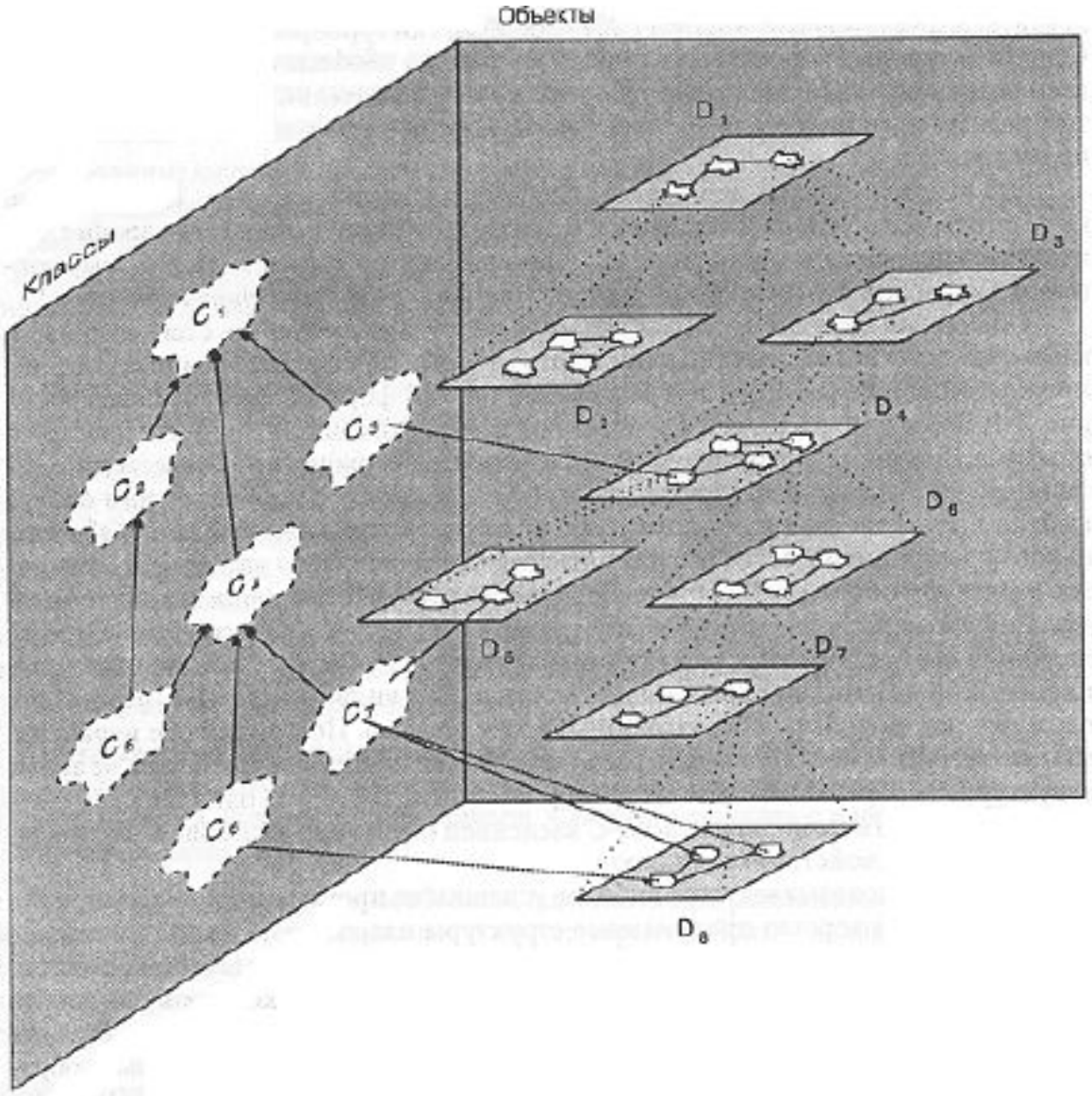


Формы сложной системы

Обнаружение общих абстракций и механизмов значительно облегчает понимание сложных систем.

Наиболее успешны те программные системы, в которых заложены хорошо продуманные структуры классов и объектов и которые обладают пятью признаками сложных систем

Каноническая форма сложной системы



Способ организации систем

- «Способ управления сложными системами был известен еще в древности — *divide et impera* (разделяй и властвуй)» (Дейкстра Э).
  - **Декомпозиция** – алгоритмическая и объектно-ориентированная
- **Абстракция** - игнорирование не слишком важных деталей и работа с обобщенной, идеализированной моделью объекта
- **Методы проектирования** упорядочивают процесс создания сложных программных систем, как общие средства доступные для всей группы разработчиков
- Объектная структура (**иерархии**) иллюстрирует схему взаимодействия объектов друг с другом