# ПРОГРАММНАЯ ИНЖЕНЕРИЯ "

- ВПЕРВЫЕ ЭТОТ ТЕРМИН БЫЛ ИСПОЛЬЗОВАН КАК ТЕМА КОНФЕРЕНЦИИ, ПРОВОДИВШЕЙСЯ ПОД ЭГИДОЙ NATO В 1968 Г
- В 1975 Г., В ВАШИНГТОНЕ БЫЛА ПРОВЕДЕНА ПЕРВАЯ МЕЖДУНАРОДНАЯ КОНФЕРЕНЦИЯ, ПОСВЯЩЕННАЯ ПРОГРАММНОЙ ИНЖЕНЕРИИ.
- ТОГДА ЖЕ ПОЯВИЛОСЬ ПЕРВОЕ ИЗДАНИЕ, ПОСВЯЩЕННОЕ ПРОГРАММНОЙ ИНЖЕНЕРИИ, — IEEE TRANSACTIONS ON SOFTWARE ENGINEERING

# ОСНОВНЫЕ ЭТАПЫ СТАНОВЛЕНИЯ *SOFTWARE ENGINEERING*

70-е и 80-е гг. — систематизация и стандартизация процессов создания ПО (на основе структурного подхода)

90-е гг. — начало перехода к сборочному, индустриальному способу создания ПО (на основе объектно-ориентированного подхода).

В основе программной инженерии лежит одна фундаментальная идея: *проектирование ПО является формальным процессом, который можно изучать и совершенствовать.*

Освоение и правильное применение методов и средств создания ПО позволят повысить качество ИС, обеспечить управляемость процесса проектирования ИС и увеличить срок ее жизни

# ОСОБЕННОСТИ КРУПНЫХ ПРОЕКТОВ СОВРЕМЕННЫХ ИС -1

- **сложность описания** (большое количество функций и процессов, элементов данных и сложные взаимосвязи между ними), требующая тщательного моделирования и анализа данных и процессов;
- **наличие совокупности тесно взаимодействующих компонентов** (подсистем), **имеющих локальные задачи и цели функционирования** (например, традиционных приложений, связанных с обработкой транзакций и решением регламентных задач, и приложений аналитической обработки (поддержки принятия решений), использующих нерегламентированные запросы к данным);
- **отсутствие полных аналогов,** ограничивающее возможность использования каких-либо типовых проектных решений и прикладных систем;

# ОСОБЕННОСТИ КРУПНЫХ ПРОЕКТОВ СОВРЕМЕННЫХ ИС -2

- **необходимость интеграции** существующих и вновь разрабатываемых приложений;
- **функционирование в неоднородной среде** на нескольких аппаратных платформах;
- **разобщенность и разнородность отдельных групп разработчиков** по уровню квалификации и сложившимся традициям использования тех или иных инструментальных средств;
- **значительная временная протяженность проекта**, обусловленная, с одной стороны, ограниченными возможностями коллектива разработчиков и, с другой стороны, масштабами организации-за-казчика и различной степенью готовности отдельных ее подразделений к внедрению ИС.

# МОДЕЛЬ АРХИТЕКТУРЫ ПО

Для успешной реализации проекта объект проектирования (ПО ИС) должен быть прежде всего адекватно описан. Т.е. должны быть построены полные и непротиворечивые модели *архитектуры ПО,* определяющие совокупность структурных элементов системы и связей между ними, поведение элементов системы в процессе их взаимодействия, а также иерархию подсистем, объединяющих структурные элементы.

*Под моделью* понимается полное описание системы ПО с определенной точки зрения.

Модели представляют собой средства для визуализации, описания, проектирования и документирования архитектуры системы

# НЕОБХОДИМОСТЬ МОДЕЛИРОВАНИЯ

По мнению Гради Буча, **моделирование является центральным звеном всей деятельности по созданию качественного ПО.**

Модели строятся для того, чтобы **понять и осмыслить структуру и поведение будущей системы, облегчить управление процессом ее создания и уменьшить возможный риск, а также документировать принимаемые проектные решения**.

Разработка модели архитектуры системы ПО промышленного характера на стадии, предшествующей ее реализации или обновлению, также необходима, как и наличие проекта для строительства большого здания.

Хорошие модели являются основой взаимодействия участников проекта. Поскольку сложность систем повышается, важно располагать эффективными методами моделирования.

Необходимо наличие строгого стандарта языка моделирования.

# ЯЗЫК МОДЕЛИРОВАНИЯ

Язык моделирования должен включать:

- **элементы модели** — фундаментальные концепции моделирования и их семантику;
- **нотацию** — визуальное представление элементов моделирования;
- **руководство по использованию** — правила применения элементов в рамках построения тех или иных типов моделей ПО.

Очевидно, что **конечная цель разработки ПО** - это не моделирование, а **получение работающих приложений** (кода). Диаграммы— это всего лишь наглядные изображения, поэтому, **используя графические языки моделирования, очень важно понимать, чем они помогут при написании кода программ.**

# ЦЕЛЕСООБРАЗНОСТЬ ИСПОЛЬЗОВАНИЯ ГРАФИЧЕСКИХ ЯЗЫКОВ МОДЕЛИРОВАНИЯ

Использование графических языков моделирования целесообразно в случаях:

* **при *изучении методов проектирования***. Множество людей отмечает наличие серьезных трудностей, связанных, например, с освоением объектно-ориентированных методов. Графические средства облегчают решение этой проблемы;
* при ***общении с экспертами организации***. Графические модели представляют архитектуру системы и объясняют, что эта система будет делать;
* при ***получении общего представления о системе***. Графические модели показывают, какого рода абстракции существуют в системе и какие ее части нуждаются в дальнейшем уточнении.

# СТРУКТУРНЫЙ И ОБЪЕКТНО-ОРИЕНТИРОВАННЫЙ ПОДХОД

В 70-80-х гг. при разработке ПО широко применялись **структурные методы**, базирующиеся на строгих формализованных методах описания ПО и принимаемых технических решений. В настоящее время широкое распространение получают **объектно-ориентированные методы.**

**Эти методы основаны на использовании наглядных графических моделей**: для описания архитектуры ПО в различных аспектах (как статической структуры, так и динамики поведения системы) **используются схемы и диаграммы**. Наглядность и строгость средств структурного и объектно-ориентированного анализа позволяют разработчикам и будущим пользователям системы с самого начала неформально участвовать в ее создании, обсуждать и закреплять понимание основных технических решений.

# CASE-СРЕДСТВА

К концу 80-х гг. **назрела необходимость в CASE-технологиях и CASE-средствах** и **возникли предпосылки для их появления**: было проведено много исследований в области программирования (разработка и внедрение языков высокого уровня, методов структурного и модульного программирования, языков проектирования и средств их поддержки, формальных и неформальных языков описания системных требований и спецификаций и т. д.)

Первоначально значение термина **CASE (Computer Aided Software Engineering)** ограничивалось вопросами автоматизации разработки только программного обеспечения, а в настоящее время оно охватывает процесс разработки сложных информационных систем в целом.

# CASE-ТЕХНОЛОГИЯ

**CASE-технология представляет собой совокупность методов проектирования информационных систем, а также набор инструментальных средств**, позволяющих:

- в наглядной форме моделировать предметную область;
- анализировать модель на всех стадиях разработки и сопровождения систем;
- разрабатывать приложения в соответствии с информационными потребностями пользователей.

Большинство существующих CASE-средств основано на методах структурного или объектно-ориентированного анализа и проектирования и используют спецификации в виде диаграмм или текстов для описания внешних требований, связей между моделями системы, динамики поведения системы и архитектуры программных средств.

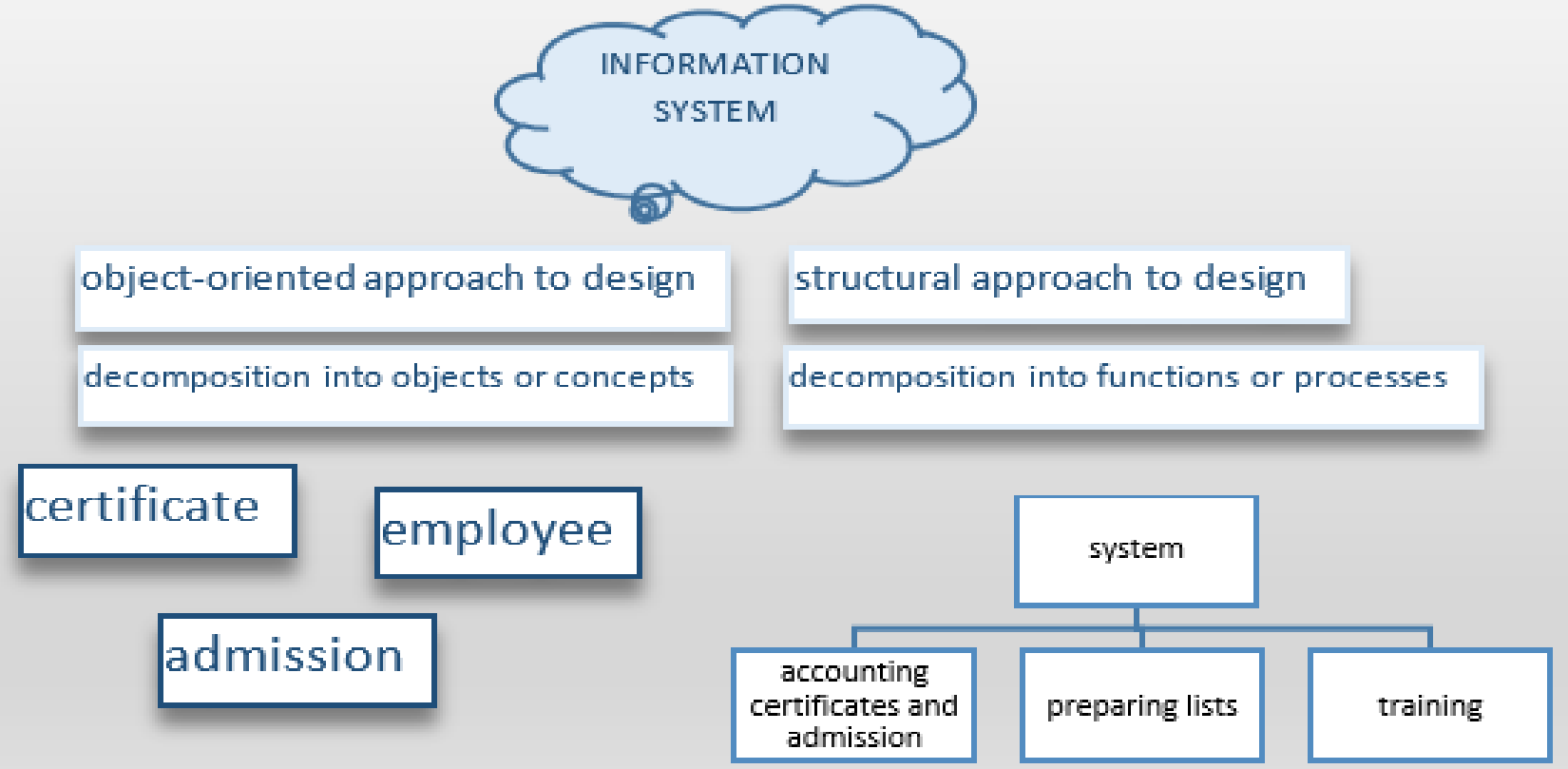
# ОТЛИЧИЕ СТРУКТУРНОГО И ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПОДХОДА К ПРОЕКТИРОВАНИЮ

Принципиальное отличие заключается в **способе декомпозиции системы**.

**Объектно-ориентированный подход** использует **объектную декомпозицию**, при этом **статическая структура системы описывается в терминах объектов и связей между ними**, а **поведение системы** описывается **в терминах обмена сообщениями между объектами.**

Каждый объект системы обладает своим собственным поведением, моделирующим поведение объекта реального мира**.**

# СРАВНЕНИЕ ПОДХОДОВ



ОСНОВНЫЕ ПОЛОЖЕНИЯ ОБЪЕКТНОЙ МОДЕЛИ ЙОНЕСАВА И ТОКОРО:

*«ТЕРМИН "ОБЪЕКТ" ПОЯВИЛСЯ ПРАКТИЧЕСКИ НЕЗАВИСИМО В РАЗЛИЧНЫХ ОБЛАСТЯХ, СВЯЗАННЫХ С КОМПЬЮТЕРАМИ, И ПОЧТИ ОДНОВРЕМЕННО В НАЧАЛЕ 70-Х ГОДОВ ДЛЯ ОБОЗНАЧЕНИЯ ТОГО, ЧТО МОЖЕТ ИМЕТЬ РАЗЛИЧНЫЕ ПРОЯВЛЕНИЯ, ОСТАВАЯСЬ ЦЕЛОСТНЫМ. ЧТОБЫ УМЕНЬШИТЬ СЛОЖНОСТЬ ПРОГРАММНЫХ СИСТЕМ, ОБЪЕКТАМИ НАЗЫВАЛИСЬ КОМПОНЕНТЫ СИСТЕМЫ ИЛИ ФРАГМЕНТЫ ПРЕДСТАВЛЯЕМЫХ ЗНАНИИ».*

ОО ПОДХОД БЫЛ СВЯЗАН С СОБЫТИЯМИ [ЛЕВИ]:

- ПРОГРЕСС В ОБЛАСТИ АРХИТЕКТУРЫ ЭВМ;
- РАЗВИТИЕ ЯП: SIMULA, SMALLTALK, CLU, ADA;
- РАЗВИТИЕ МЕТОДОЛОГИИ ПРОГРАММИРОВАНИЯ, ВКЛЮЧАЯ ПРИНЦИПЫ МОДУЛЬНОСТИ И СКРЫТИЯ ДАННЫХ.

НА СТАНОВЛЕНИЕ ОБЪЕКТНОГО ПОДХОДА ОКАЗАЛИ ВЛИЯНИЕ:

- РАЗВИТИЕ ТЕОРИИ БАЗ ДАННЫХ;
- ИССЛЕДОВАНИЯ В ОБЛАСТИ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА;
- ДОСТИЖЕНИЯ ФИЛОСОФИИ И ТЕОРИИ ПОЗНАНИЯ.

ПОНЯТИЕ "ОБЪЕКТ" ВПЕРВЫЕ БЫЛО ИСПОЛЬЗОВАНО ПРИ КОНСТРУИРОВАНИИ КОМПЬЮТЕРОВ С DESCRIPTOR-BASED И CAPABILITY-BASED АРХИТЕКТУРАМИ.

В ЭТИХ РАБОТАХ - ПОПЫТКИ ОТОЙТИ ОТ АРХИТЕКТУРЫ ФОН НЕЙМАНА И ПРЕОДОЛЕТЬ БАРЬЕР МЕЖДУ ВЫСОКИМ УРОВНЕМ ПРОГРАММНОЙ АБСТРАКЦИИ И НИЗКИМ УРОВНЕМ ЭВМ.

ПО МНЕНИЮ СТОРОННИКОВ ЭТИХ ПОДХОДОВ, БЫЛИ СОЗДАНЫ БОЛЕЕ КАЧЕСТВЕННЫЕ СРЕДСТВА, ОБЕСПЕЧИВАЮЩИЕ: ЛУЧШЕЕ ВЫЯВЛЕНИЕ ОШИБОК, БОЛЬШУЮ ЭФФЕКТИВНОСТЬ РЕАЛИЗАЦИИ ПРОГРАММ, СОКРАЩЕНИЕ НАБОРА ИНСТРУКЦИЙ, УПРОЩЕНИЕ КОМПИЛЯЦИИ, СНИЖЕНИЕ ОБЪЕМА ТРЕБУЕМОЙ ПАМЯТИ.

КОМПЬЮТЕРЫ С ОО АРХИТЕКТУРОЙ: BURROUGHS 5000, PLESSEY 250, CAMBRIDGE CAP, SWARD, INTEL 432, CALTECH'S COM, IBM SYSTEM/38, RATIONAL R1000, BIIN 40 И 60.

С ОО АРХИТЕКТУРОЙ СВЯЗАНЫ ОО ОС.

ДЕЙКСТРА, РАБОТАЯ НАД МУЛЬТИПРОГРАММНОЙ СИСТЕМОЙ THE, ВПЕРВЫЕ ВВЕЛ ПОНЯТИЕ МАШИНЫ С УРОВНЯМИ СОСТОЯНИЯ В КАЧЕСТВЕ СРЕДСТВА ПОСТРОЕНИЯ СИСТЕМЫ.

ВКЛАД В ОБЪЕКТНЫЙ ПОДХОД ВНЕСЕН ОБЪЕКТНЫМИ И ОО ЯП.

ВПЕРВЫЕ ПОНЯТИЯ КЛАССОВ И ОБЪЕКТОВ ВВЕДЕНЫ В ЯЗЫКЕ SIMULA 67.

СИСТЕМА FLEX И ДИАЛЕКТЫ SMALLTALK-72, -74, -76, -80, ВЗЯВ ЗА ОСНОВУ МЕТОДЫ SIMULA, ДОВЕЛИ ИХ ДО ЛОГИЧЕСКОГО ЗАВЕРШЕНИЯ, ВЫПОЛНЯЯ ВСЕ ДЕЙСТВИЯ НА ОСНОВЕ КЛАССОВ.

В 1970-Х ГГ. - ЯП, РЕАЛИЗУЮЩИХ ИДЕЮ АБСТРАКЦИИ ДАННЫХ: ALPHARD, CLU, EUCLID, GYPSY, MESA И MODULA.

МЕТОДЫ, ИСПОЛЬЗУЕМЫЕ В ЯЗЫКАХ SIMULA И SMALLTALK, БЫЛИ ИСПОЛЬЗОВАНЫ В ТРАДИЦИОННЫХ ЯП ВЫСОКОГО УРОВНЯ.

ВНЕСЕНИЕ ОО ПОДХОДА В С ПРИВЕЛО К ВОЗНИКНОВЕНИЮ ЯЗЫКОВ C++ И OBJECTIVE C.

НА ОСНОВЕ ЯП PASCAL ВОЗНИКЛИ OBJECT PASCAL, EIFFEL И ADA.

ПОЯВИЛИСЬ ДИАЛЕКТЫ LISP: FLAVORS, LOOPS И CLOS (COMMON LISP OBJECT SYSTEM), С ВОЗМОЖНОСТЯМИ ЯЗЫКОВ SIMULA И SMALLTALK.

ДЕЙКСТРА УКАЗАЛ НА НЕОБХОДИМОСТЬ ПОСТРОЕНИЯ СИСТЕМ В ВИДЕ СТРУКТУРИРОВАННЫХ АБСТРАКЦИЙ.

ПАРНАС ВВЕЛ ИДЕЮ СКРЫТИЯ ИНФОРМАЦИИ.

В 70-Х ГГ. ЛИСКОВ И ЖИЛЬ, ГУТТАГ И ШОУ РАЗРАБОТАЛИ МЕХАНИЗМЫ АБСТРАКТНЫХ ТИПОВ ДАННЫХ.

ХОАР ДОПОЛНИЛ ЭТИ ПОДХОДЫ ТЕОРИЕЙ ТИПОВ И ПОДКЛАССОВ.

НА ОБЪЕКТНЫЙ ПОДХОД ОКАЗАЛИ ВЛИЯНИЕ ТЕХНОЛОГИИ ПОСТРОЕНИЯ БД, БЛАГОДАРЯ "СУЩНОСТЬ-ОТНОШЕНИЕ" (ER, ENTITY-RELATIONSHIP). В МОДЕЛЯХ ER (ЧЕН), МОДЕЛИРОВАНИЕ - В ТЕРМИНАХ СУЩНОСТЕЙ, ИХ АТРИБУТОВ И ВЗАИМООТНОШЕНИЙ.

В ПОНИМАНИЕ ОО АБСТРАКЦИЙ - ВКЛАД РАЗРАБОТЧИКИ СПОСОБОВ ПРЕДСТАВЛЕНИЯ ДАННЫХ В ОБЛАСТИ ИИ:

В 1975 Г. МИНСКИ - ТЕОРИЯ ФРЕЙМОВ ДЛЯ ПРЕДСТАВЛЕНИЯ РЕАЛЬНЫХ ОБЪЕКТОВ В СИСТЕМАХ РАСПОЗНАВАНИЯ ОБРАЗОВ И ЕСТЕСТВЕННЫХ ЯЗЫКОВ. ФРЕЙМЫ СТАЛИ АРХИТЕКТУРНОЙ ОСНОВОЙ В ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМАХ

# ОБЪЕКТНЫЙ ПОДХОД ИЗВЕСТЕН ИЗДАВНА:

ГРЕКИ - ИДЕЯ О ТОМ, ЧТО МИР МОЖНО РАССМАТРИВАТЬ В ТЕРМИНАХ КАК ОБЪЕКТОВ, ТАК И СОБЫТИЙ.

В XVII ВЕКЕ ДЕКАРТ: ЛЮДИ ИМЕЮТ ОО ВЗГЛЯД НА МИР.

В XX ВЕКЕ ЭТУ ТЕМУ РАЗВИВАЛА РЭНД В СВОЕЙ ФИЛОСОФИИ ОБЪЕКТИВИСТСКОЙ ЭПИСТЕМОЛОГИИ.

МИНСКИ ПРЕДЛОЖИЛ МОДЕЛЬ ЧЕЛОВЕЧЕСКОГО МЫШЛЕНИЯ, В КОТОРОЙ РАЗУМ ЧЕЛОВЕКА РАССМАТРИВАЕТСЯ КАК ОБЩНОСТЬ РАЗЛИЧНО МЫСЛЯЩИХ АГЕНТОВ. ОН ДОКАЗЫВАЕТ, ЧТО ТОЛЬКО СОВМЕСТНОЕ ДЕЙСТВИЕ ТАКИХ АГЕНТОВ ПРИВОДИТ К ОСМЫСЛЕННОМУ ПОВЕДЕНИЮ ЧЕЛОВЕКА.

# ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ

## *(OOP , OBJECT-ORIENTED PROGRAMMING) –*

*МЕТОДОЛОГИЯ ПРОГРАММИРОВАНИЯ, ОСНОВАННАЯ НА ПРЕДСТАВЛЕНИИ ПРОГРАММЫ В ВИДЕ СОВОКУПНОСТИ ОБЪЕКТОВ, КАЖДЫЙ ИЗ КОТОРЫХ ЯВЛЯЕТСЯ ЭКЗЕМПЛЯРОМ ОПРЕДЕЛЕННОГО КЛАССА, А КЛАССЫ ОБРАЗУЮТ ИЕРАРХИЮ НАСЛЕДОВАНИЯ.*

3 ЧАСТИ:

1) ООР ИСПОЛЬЗУЕТ В КАЧЕСТВЕ БАЗОВЫХ ЭЛЕМЕНТОВ *ОБЪЕКТЫ*, А НЕ АЛГОРИТМЫ (ИЕРАРХИЯ "БЫТЬ ЧАСТЬЮ»);
2) КАЖДЫЙ ОБЪЕКТ ЯВЛЯЕТСЯ *ЭКЗЕМПЛЯРОМ* КАКОГО-ЛИБО ОПРЕДЕЛЕННОГО *КЛАССА*;
3) КЛАССЫ ОРГАНИЗОВАНЫ *ИЕРАРХИЧЕСКИ* (ИЕРАРХИЯ «IS A».

ПРОГРАММА БУДЕТ ОО ТОЛЬКО ПРИ СОБЛЮДЕНИИ ВСЕХ ТРЕХ ТРЕБОВАНИЙ.

ПРОГРАММИРОВАНИЕ, НЕ ОСНОВАННОЕ НА ИЕРАРХИЧЕСКИХ ОТНОШЕНИЯХ, НЕ ОТНОСИТСЯ К ООР, А НАЗЫВАЕТСЯ *ПРОГРАММИРОВАНИЕМ НА ОСНОВЕ АБСТРАКТНЫХ ТИПОВ ДАННЫХ.*

НЕ ВСЕ ЯП ОБЪЕКТНО-ОРИЕНТИРОВАННЫЕ.

СТРАУСТРУП: «*ЕСЛИ ТЕРМИН ОО ЯЗЫК ВООБЩЕ ЧТО-ЛИБО ОЗНАЧАЕТ, ТО ОН ДОЛЖЕН ОЗНАЧАТЬ ЯЗЫК, ИМЕЮЩИЙ СРЕДСТВА ХОРОШЕЙ ПОДДЕРЖКИ ОО СТИЛЯ ПРОГРАММИРОВАНИЯ… ОБЕСПЕЧЕНИЕ ТАКОГО СТИЛЯ ОЗНАЧАЕТ, ЧТО В ЯЗЫКЕ УДОБНО ПОЛЬЗОВАТЬСЯ ЭТИМ СТИЛЕМ. ЕСЛИ НАПИСАНИЕ ПРОГРАММ В СТИЛЕ ООР ТРЕБУЕТ СПЕЦИАЛЬНЫХ УСИЛИЙ ИЛИ ОНО НЕВОЗМОЖНО СОВСЕМ, ТО ЭТОТ ЯЗЫК НЕ ОТВЕЧАЕТ ТРЕБОВАНИЯМ ООР*».

ТЕОРЕТИЧЕСКИ ВОЗМОЖНА ИМИТАЦИЯ ОО ПРОГРАММИРОВАНИЯ НА ОБЫЧНЫХ ЯЗЫКАХ (PASCAL, COBOL ИЛИ АССЕМБЛЕР), НО ЭТО ЗАТРУДНИТЕЛЬНО.

КАРДЕЛЛИ И ВЕГНЕР:

«*ЯП ЯВЛЯЕТСЯ ОО ТОГДА И ТОЛЬКО ТОГДА, КОГДА ВЫПОЛНЯЮТСЯ УСЛОВИЯ:*

- *ПОДДЕРЖИВАЮТСЯ ОБЪЕКТЫ, ТО ЕСТЬ АБСТРАКЦИИ ДАННЫХ, ИМЕЮЩИЕ ИНТЕРФЕЙС В ВИДЕ ИМЕНОВАННЫХ ОПЕРАЦИЙ И СОБСТВЕННЫЕ ДАННЫЕ, С ОГРАНИЧЕНИЕМ ДОСТУПА К НИМ.*
- *ОБЪЕКТЫ ОТНОСЯТСЯ К СООТВЕТСТВУЮЩИМ ТИПАМ (КЛАССАМ).*
- *ТИПЫ (КЛАССЫ) МОГУТ НАСЛЕДОВАТЬ АТРИБУТЫ СУПЕРТИПОВ (СУПЕРКЛАССОВ)*".

ПОДДЕРЖКА НАСЛЕДОВАНИЯ В ТАКИХ ЯП ОЗНАЧАЕТ ВОЗМОЖНОСТЬ УСТАНОВЛЕНИЯ ОТНОШЕНИЯ "IS-A" («ЭТО ЕСТЬ»), НАПРИМЕР: КРАСНАЯ РОЗА - ЭТО ЦВЕТОК, А ЦВЕТОК - ЭТО РАСТЕНИЕ.

ЯЗЫКИ, НЕ ИМЕЮЩИЕ ТАКИХ МЕХАНИЗМОВ, НЕЛЬЗЯ ОТНЕСТИ К ОО. ТАКИЕ ЯП – *ОБЪЕКТНЫЕ (*КАРДЕЛЛИ И ВЕГНЕР).

ОО ЯП: SMALLTALK, OBJECT PASCAL, C++ И CLOS;

ADA - ОБЪЕКТНЫЙ ЯЗЫК.

ТАК КАК ОБЪЕКТЫ И КЛАССЫ ЯВЛЯЮТСЯ ЭЛЕМЕНТАМИ ОБЕИХ ГРУПП ЯЗЫКОВ, ЖЕЛАТЕЛЬНО ИСПОЛЬЗОВАТЬ И В ТЕХ, И В ДРУГИХ МЕТОДЫ ОО ПРОЕКТИРОВАНИЯ.

# ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОЕКТИРОВАНИЕ

*(OOD, OBJECT-ORIENTED DESIGN) –*

*МЕТОДОЛОГИЯ ПРОЕКТИРОВАНИЯ, СОЕДИНЯЮЩАЯ ПРОЦЕСС ОБЪЕКТНОЙ ДЕКОМПОЗИЦИИ И ПРИЕМЫ ПРЕДСТАВЛЕНИЯ ЛОГИЧЕСКОЙ И ФИЗИЧЕСКОЙ, СТАТИЧЕСКОЙ И ДИНАМИЧЕСКОЙ МОДЕЛЕЙ ПРОЕКТИРУЕМОЙ СИСТЕМЫ*

2 ЧАСТИ: OOD

1) ОСНОВЫВАЕТСЯ НА ОО ДЕКОМПОЗИЦИИ;

2) ИСПОЛЬЗУЕТ МНОГООБРАЗИЕ ПРИЕМОВ ПРЕДСТАВЛЕНИЯ МОДЕЛЕЙ, ОТРАЖАЮЩИХ ЛОГИЧЕСКУЮ (КЛАССЫ И ОБЪЕКТЫ) И ФИЗИЧЕСКУЮ (МОДУЛИ И ПРОЦЕССЫ) СТРУКТУРУ СИСТЕМЫ, А ТАКЖЕ ЕЕ СТАТИЧЕСКИЕ И ДИНАМИЧЕСКИЕ АСПЕКТЫ.

ПРОГРАММИРОВАНИЕ ПОДРАЗУМЕВАЕТ ПРАВИЛЬНОЕ И ЭФФЕКТИВНОЕ ИСПОЛЬЗОВАНИЕ МЕХАНИЗМОВ КОНКРЕТНЫХ ЯП.

ПРОЕКТИРОВАНИЕ ОСНОВНОЕ ВНИМАНИЕ УДЕЛЯЕТ ПРАВИЛЬНОМУ И ЭФФЕКТИВНОМУ СТРУКТУРИРОВАНИЮ СЛОЖНЫХ СИСТЕМ.

# ОБЪЕКТНО-ОРИЕНТИРОВАННЫЙ АНАЛИЗ

*(OOA, OBJECT-ORIENTED ANALYSIS) –*

*МЕТОДОЛОГИЯ, ПРИ КОТОРОЙ ТРЕБОВАНИЯ К СИСТЕМЕ ВОСПРИНИМАЮТСЯ С Т.ЗР. КЛАССОВ И ОБЪЕКТОВ, ВЫЯВЛЕННЫХ В ПРЕДМЕТНОЙ ОБЛАСТИ.*

НА ОБЪЕКТНУЮ МОДЕЛЬ ПОВЛИЯЛА БОЛЕЕ РАННЯЯ МОДЕЛЬ ЖЦ ПО. ТРАДИЦИОННАЯ ТЕХНИКА СТРУКТУРНОГО АНАЛИЗА (ДЕ МАРК, ИОРДАН, ГЕЙН И САРСОН, С УТОЧНЕНИЯМИ ДЛЯ РЕЖИМОВ РЕАЛЬНОГО ВРЕМЕНИ - ВАРДА И МЕЛЛОРА, ХОТЛИ И ПИРБХАЯ) ОСНОВАНА НА ПОТОКАХ ДАННЫХ В СИСТЕМЕ.

ОО АНАЛИЗ НАПРАВЛЕН НА СОЗДАНИЕ МОДЕЛЕЙ РЕАЛЬНОЙ ДЕЙСТВИТЕЛЬНОСТИ НА ОСНОВЕ ОО МИРОВОЗЗРЕНИЯ.

КАК СООТНОСЯТСЯ OOA, OOD И OOP?

НА РЕЗУЛЬТАТАХ OOA ФОРМИРУЮТСЯ МОДЕЛИ, НА КОТОРЫХ ОСНОВЫВАЕТСЯ OOD;

OOD СОЗДАЕТ ФУНДАМЕНТ ДЛЯ ОКОНЧАТЕЛЬНОЙ РЕАЛИЗАЦИИ СИСТЕМЫ С ИСПОЛЬЗОВАНИЕМ МЕТОДОЛОГИИ OOP.

# СОСТАВНЫЕ ЧАСТИ ОБЪЕКТНОГО ПОДХОДА

# ПАРАДИГМЫ ПРОГРАММИРОВАНИЯ

ДЖЕНКИНС И ГЛАЗГО: *«В БОЛЬШИНСТВЕ СВОЕМ ПРОГРАММИСТЫ ИСПОЛЬЗУЮТ В РАБОТЕ ОДИН ЯП И СЛЕДУЮТ ОДНОМУ СТИЛЮ. ОНИ ПРОГРАММИРУЮТ В ПАРАДИГМЕ, НАВЯЗАННОЙ ИСПОЛЬЗУЕМЫМ ИМИ ЯЗЫКОМ. ЧАСТО ОНИ ОСТАВЛЯЮТ В СТОРОНЕ АЛЬТЕРНАТИВНЫЕ ПОДХОДЫ К ЦЕЛИ, А СЛЕДОВАТЕЛЬНО, ИМ ТРУДНО УВИДЕТЬ ПРЕИМУЩЕСТВА СТИЛЯ, БОЛЕЕ СООТВЕТСТВУЮЩЕГО РЕШАЕМОЙ ЗАДАЧЕ».*

БОБРОВ И СТЕФИК ОПРЕДЕЛИЛИ ПОНЯТИЕ СТИЛЯ ПРОГРАММИРОВАНИЯ: *«ЭТО СПОСОБ ПОСТРОЕНИЯ ПРОГРАММ, ОСНОВАННЫЙ НА ОПРЕДЕЛЕННЫХ ПРИНЦИПАХ ПРОГРАММИРОВАНИЯ, И ВЫБОР ПОДХОДЯЩЕГО ЯЗЫКА, КОТОРЫЙ ДЕЛАЕТ ПОНЯТНЫМИ ПРОГРАММЫ, НАПИСАННЫЕ В ЭТОМ СТИЛЕ».*

# 5 ОСНОВНЫХ РАЗНОВИДНОСТЕЙ СТИЛЕЙ ПРОГРАММИРОВАНИЯ С ПРИСУЩИМИ ИМ ВИДАМИ АБСТРАКЦИЙ:

| | |
|---|---|
| · ПРОЦЕДУРНО-ОРИЕНТИРОВАННЫЙ | АЛГОРИТМЫ |
| · ОБЪЕКТНО-ОРИЕНТИРОВАННЫЙ | КЛАССЫ И ОБЪЕКТЫ |
| · ЛОГИКО-ОРИЕНТИРОВАННЫЙ | ЦЕЛИ, В ТЕРМИНАХ ИСЧИСЛЕНИЯ ПРЕДИКАТОВ |
| · ОРИЕНТИРОВАННЫЙ НА ПРАВИЛА | ПРАВИЛА «ЕСЛИ-ТО» |
| · ОРИЕНТИРОВАННЫЙ НА ОГРАНИЧЕНИЯ | ИНВАРИАНТНЫЕ СООТНОШЕНИЯ |

НЕВОЗМОЖНО ПРИЗНАТЬ КАКОЙ-ЛИБО СТИЛЬ ПРОГРАММИРОВАНИЯ ЛУЧШИМ ВО ВСЕХ ОБЛАСТЯХ ПРАКТИЧЕСКОГО ПРИМЕНЕНИЯ.

ДЛЯ ПРОЕКТИРОВАНИЯ БЗ БОЛЕЕ ПРИГОДЕН СТИЛЬ, ОРИЕНТИРОВАННЫЙ НА ПРАВИЛА, ДЛЯ ВЫЧИСЛИТЕЛЬНЫХ ЗАДАЧ - ПРОЦЕДУРНО-ОРИЕНТИРОВАННЫЙ.

ОО СТИЛЬ НАИБОЛЕЕ ПРИЕМЛЕМ ДЛЯ ШИРОКОГО КРУГА ПРИЛОЖЕНИЙ.

КАЖДЫЙ СТИЛЬ ПРОГРАММИРОВАНИЯ ИМЕЕТ СВОЮ КОНЦЕПТУАЛЬНУЮ БАЗУ. ДЛЯ ОО СТИЛЯ КОНЦЕПТУАЛЬНАЯ БАЗА - ЭТО *ОБЪЕКТНАЯ МОДЕЛЬ*.

# ТРИ МЕТОДА ОРГАНИЗАЦИИ МЫШЛЕНИЯ

В разделе «Теория классификации» Британской энциклопедии сказано следующее: «В постижении реального мира люди пользуются тремя методами, организующими их мышление:

(1) **разделение окружающей действительности на конкретные объекты и их атрибуты** (например, когда явно различаются понятия дерева и его высоты или пространственного расположения по отношению к другим объектам);

(2) **различие между целыми объектами и их составными частями** (например, ветви являются составными частями дерева);

(3) **формирование и выделение различий между различными классами объектов** (например, между классом всех деревьев и классом всех камней.)»

Объектная модель является наиболее естественным способом представления реального мира

# ОБЪЕКТНАЯ МОДЕЛЬ

К основным понятиям объектно-ориентированного подхода относятся:

- **объект;**
- **класс;**
- **атрибут;**
- **операция;**
- **полиморфизм (интерфейс);**
- **компонент;**
- **связи.**

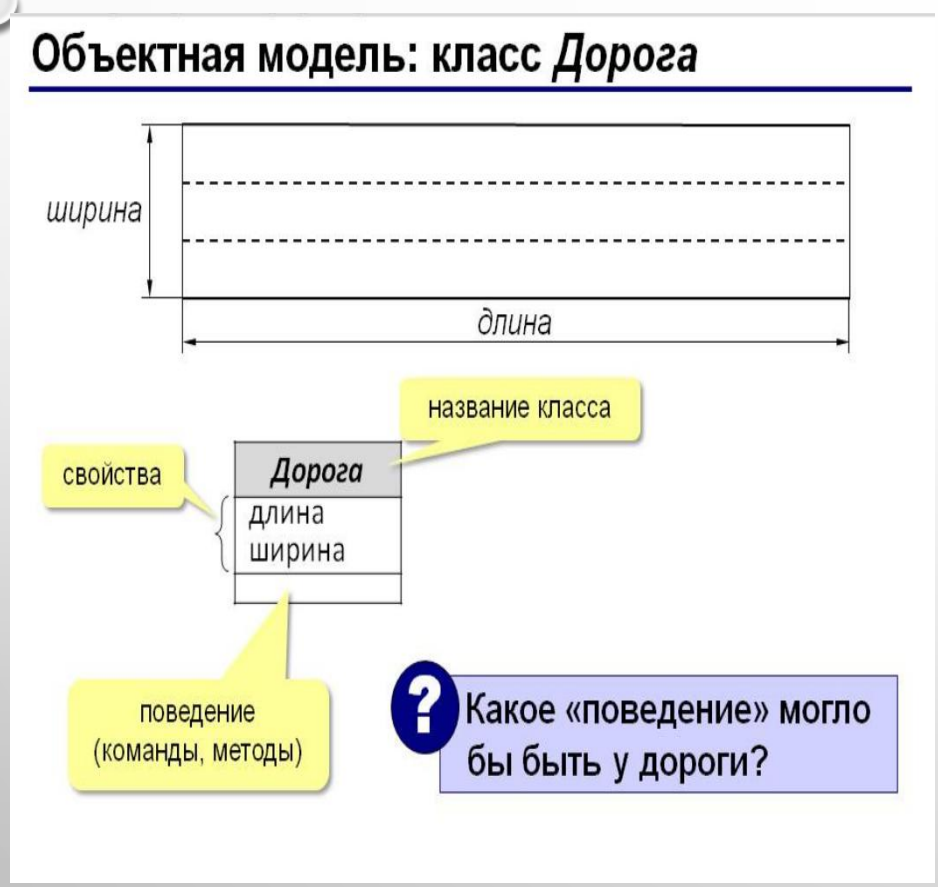
Основными ее элементами являются:

- **абстрагирование (abstraction);**
- **инкапсуляция (encapsulation);**
- **модульность (modularity);**
- **иерархия (hierarchy).**

Три дополнительных элемента, не являющихся в отличие от основных строго обязательными:

- **типизация (typing);**
- **параллелизм (concurrency);**
- **устойчивость (persistence).**

# ОБЪЕКТНАЯ МОДЕЛЬ



# АБСТРАГИРОВАНИЕ

**Абстрагирование — это выделение существенных характеристик некоторого объекта, которые отличают его от всех других видов объектов** и, таким образом, четко определяют его концептуальные границы относительно дальнейшего рассмотрения и анализа.

Абстрагирование концентрирует внимание **на внешних особенностях объекта** и позволяет отделить самые существенные особенности его поведения от деталей их реализации. Выбор правильного набора абстракций для заданной предметной области представляет собой главную задачу объектно-ориентированного проектирования.

# ИНКАПСУЛЯЦИЯ

**Инкапсуляция** - это **процесс отделения друг от друга отдельных элементов объекта, определяющих его устройство и поведение**.

**Инкапсуляция служит** для того, чтобы **изолировать интерфейс объекта, отражающий его внешнее поведение, от внутренней реализации объекта**. Объектный подход предполагает, что собственные ресурсы, которыми могут манипулировать только методы самого класса, скрыты от внешней среды.

Абстрагирование и инкапсуляция являются взаимодополняющими операциями:

- абстрагирование фокусирует внимание на внешних особенностях объекта,
- инкапсуляция (или, иначе, ограничение доступа) не позволяет объектам-пользователям различать внутреннее устройство объекта.

# МОДУЛЬНОСТЬ И ИЕРАРХИЯ

**Модульность** - это свойство системы, связанное с возможностью ее декомпозиции на ряд внутренне связных, но слабо связанных между собой модулей. Инкапсуляция и модульность создают барьеры между абстракциями.

**Иерархия** — это ранжированная или упорядоченная система абстракций, расположение их по уровням.

**Основными видами иерархических структур** применительно к сложным системам являются **структура классов** (иерархия по номенклатуре) и **структура объектов** (иерархия по составу).

Примерами **иерархии классов являются простое и множественное наследование** (один класс использует структурную или функциональную часть соответственно одного или нескольких других классов).

Пример **иерархии объектов — агрегация**.

# ТИПИЗАЦИЯ, ПАРАЛЛЕЛИЗМ,УСТОЙЧИВОСТЬ

**Типизация** — это ограничение, накладываемое на класс объектов и препятствующее взаимозаменяемости различных классов (или сильно сужающее ее возможность). Типизация позволяет защититься от использования объектов одного класса вместо другого или по крайней мере управлять таким использованием.

**Параллелизм** — свойство объектов находиться в активном или пассивном состоянии и различать активные и пассивные объекты между собой.

**Устойчивость** — свойство объекта существовать во времени (вне зависимости от процесса, породившего данный объект) и/или в пространстве (при перемещении объекта из адресного пространства, в котором он был создан).

# ОБЪЕКТ

**Объект** определяется как **осязаемая реальность** (tangible entity) — предмет или явление, **имеющие четко определяемое поведение.**

Объект обладает **состоянием, поведением и индивидуальностью**. **Структура и поведение схожих объектов** определяют общий для них **класс**. Термины "экземпляр класса" и "объект" являются эквивалентными.

**Состояние** объекта характеризуется перечнем всех возможных (статических) свойств данного объекта и текущими значениями (динамическими) каждого из этих свойств.

**Поведение** характеризует воздействие объекта на другие объекты и полностью определяется его действиями.

Каждый класс обладает уникальной индивидуальностью. **Индивидуальность** — это свойства объекта, отличающие его от всех других объектов.

# КЛАСС, АТРИБУТ

**Класс** — это **множество объектов, связанных общностью структуры и поведения.** Любой объект является экземпляром класса. Определение классов и объектов — одна из самых сложных задач объектно-ориентированного проектирования.

**Атрибут — поименованное свойство класса, определяющее диапазон допустимых значений, которые могут принимать экземпляры данного свойства.** Атрибут — это элемент информации, связанный **с** классом.

**Например**, у класса Company (Компания) могут быть атрибуты Name (Название), Address (Адрес) и NumberOfEmployees (Число служащих).

# ОПЕРАЦИИ (МЕТОДЫ)

**Определенное воздействие одного объекта на другой с целью вызвать соответствующую реакцию называется операцией.**

Операция — это реализация услуги, которую можно запросить у любого объекта данного класса.

Операции отражают поведение объекта.

**Операция-запрос** не изменяет состояния объекта.

**Операция-команда** может изменить состояние объекта.

Результат операции зависит от текущего состояния объекта. Как правило, в объектных и объектно-ориентированных языках программирования операции, выполняемые над данным объектом, называются **методами** и являются составной частью определения класса.

# НАСЛЕДОВАНИЕ И ПОЛИМОРФИЗМ

Понятие **полиморфизма** можно объяснить как **способность класса принадлежать более чем одному типу.**

**Наследование** означает **построение новых классов на основе существующих** с возможностью **добавления или переопределения данных и методов**.

**Наследование и полиморфизм обеспечивают возможность определения новой функциональности классов** с помощью создания производных классов — **потомков базовых классов**. Потомки наследуют характеристики родительских классов без изменения их первоначального описания и добавляют при необходимости собственные структуры данных и методы. Определение производных классов, при котором задаются только различия или уточнения, в огромной степени экономит время и усилия при производстве и использовании спецификаций и программного кода.

# КОМПОНЕНТЫ, СВЯЗИ

**Компонент** — относительно независимая и замещаемая часть системы, выполняющая четко определенную функцию в контексте заданной архитектуры. Компонент представляет собой физическую реализацию проектной абстракции и может быть:

- компонентом исходного кода;
- компонентом времени выполнения (run time);
- исполняемым компонентом.

Компонент обеспечивает физическую реализацию набора интерфейсов.

Между элементами объектной модели существуют различные виды связей.

К основным типам связей относятся **связи ассоциации, зависимости и обобщения.**

# ВАЖНЫЕ КАЧЕСТВА ОБЪЕКТНОГО ПОДХОДА

Важным качеством объектного подхода является **согласованность моделей деятельности организации и моделей проектируемой системы от стадии формирования требований до стадии реализации**.

Требование согласованности моделей выполняется благодаря возможности применения абстрагирования, модульности, полиморфизма на всех стадиях разработки.

Модели ранних стадий могут быть непосредственно подвергнуты сравнению с моделями реализации. По объектным моделям может быть прослежено отображение реальных сущностей моделируемой предметной области (организации) в объекты и классы информационной системы.

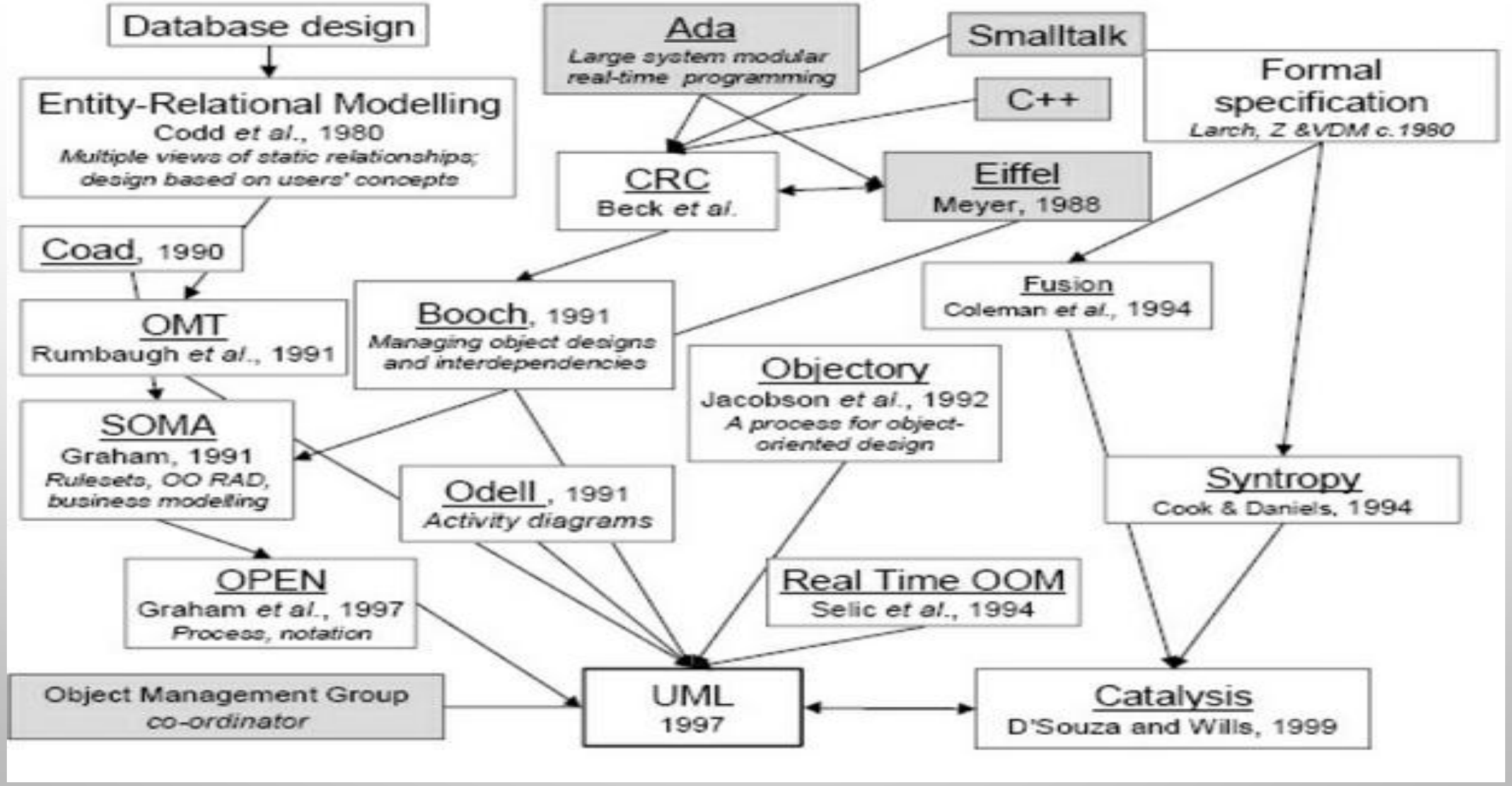
В 80-Е ГОДЫ - МНОЖЕСТВО РАЗЛИЧНЫХ МЕТОДОЛОГИЙ МОДЕЛИРОВАНИЯ.

КАЖДАЯ ИМЕЛА СВОИ ДОСТОИНСТВА И НЕДОСТАТКИ, СВОЮ НОТАЦИЮ.

ТО СМУТНОЕ ВРЕМЯ ПОЛУЧИЛО НАЗВАНИЕ "ВОЙНЫ МЕТОДОВ".

ПРОБЛЕМА: РАЗНЫЕ ЛЮДИ ИСПОЛЬЗОВАЛИ РАЗНЫЕ НОТАЦИИ. ОДИН И ТОТ ЖЕ СИМВОЛ ОЗНАЧАЛ В РАЗНЫХ НОТАЦИЯХ АБСОЛЮТНО РАЗНЫЕ ВЕЩИ.

UML - ЧЕРТЫ НОТАЦИЙ ГРАДИ БУЧА (GRADY BOOCH), ДЖИМА РУМБАХА (JIM RUMBAUGH), АЙВАРА ЯКОБСОНА (IVAR JACOBSON) И ДРУГИХ.




Часть методов, которые существовали в то время и повлияли на UML

ПОЯВЛЕНИЕ ООП ТРЕБОВАЛО УДОБНОГО ИНСТРУМЕНТА ДЛЯ МОДЕЛИРОВАНИЯ, ЕДИНОЙ НОТАЦИИ ДЛЯ ОПИСАНИЯ СЛОЖНЫХ ПРОГРАММНЫХ СИСТЕМ.

ТРИ СПЕЦИАЛИСТА, ТРИ АВТОРА НАИБОЛЕЕ ПОПУЛЯРНЫХ МЕТОДОВ РЕШИЛИ ОБЪЕДИНИТЬ СВОИ РАЗРАБОТКИ.

В 1991-М КАЖДЫЙ ИЗ НИХ НАПИСАЛ КНИГУ, В КОТОРОЙ ИЗЛОЖИЛ СВОЙ МЕТОД ООАП. КАЖДАЯ МЕТОДОЛОГИЯ БЫЛА ПО-СВОЕМУ ХОРОША, НО ИМЕЛА И НЕДОСТАТКИ.

МЕТОД **БУЧА -** ХОРОШ В ПРОЕКТИРОВАНИИ, НО СЛАБОВАТ В АНАЛИЗЕ.

*OMT* **РУМБАХА -** ОТЛИЧНОЕ СРЕДСТВО АНАЛИЗА, НО ПЛОХ В ПРОЕКТИРОВАНИИ.

OBJECTORY **ЯКОБСОНА -** ХОРОШ С ТОЧКИ ЗРЕНИЯ *USER EXPERIENCE*, НА КОТОРЫЙ НИ МЕТОД БУЧА, НИ *OMT* НЕ ОБРАЩАЛИ ОСОБОГО ВНИМАНИЯ.

ОСНОВНАЯ ИДЕЯ OBJECTORY - АНАЛИЗ ДОЛЖЕН НАЧИНАТЬСЯ С ПРЕЦЕДЕНТОВ, А НЕ С ДИАГРАММЫ КЛАССОВ, КОТОРЫЕ ДОЛЖНЫ БЫТЬ ПРОИЗВОДНЫМИ ОТ НИХ.

В 1994 Г. СУЩЕСТВОВАЛО 72 МЕТОДА ИЛИ ЧАСТНЫЕ МЕТОДИКИ. МНОГИЕ ИЗ НИХ «ПЕРЕКРЫВАЛИСЬ» - ИСПОЛЬЗОВАЛИ ПОХОЖИЕ ИДЕИ, НОТАЦИИ.

ЧУВСТВОВАЛАСЬ ОСТРАЯ ПОТРЕБНОСТЬ, "СОЦИАЛЬНЫЙ ЗАКАЗ" - ЗАКОНЧИТЬ "ВОЙНУ МЕТОДОВ" И ОБЪЕДИНИТЬ В ОДНОМ УНИФИЦИРОВАННОМ СРЕДСТВЕ ВСЕ ЛУЧШЕЕ, ЧТО БЫЛО СОЗДАНО В ОБЛАСТИ МОДЕЛИРОВАНИЯ.

СЕЙЧАС THE UML ЖИВЕТ И РАЗВИВАЕТСЯ. ИМЕЕМ ДЕСЯТКИ CASE-СРЕДСТВ, ПОДДЕРЖИВАЮЩИХ UML.

RATIONAL НЕ ВЛАДЕЕТ UML, НО ПРОДОЛЖАЕТ РАБОТАТЬ НАД НИМ. UML ПРИНАДЛЕЖИТ OMG, А САМА RATIONAL ЯВЛЯЕТСЯ ОДНИМ ИЗ ПОДРАЗДЕЛЕНИЙ IBM И ФИГУРИРУЕТ ВО ВСЕХ ДОКУМЕНТАХ КАК IBM RATIONAL.

UML ПОЛУЧИЛ МНОЖЕСТВО ПАКЕТОВ РАСШИРЕНИЙ – ***ПРОФАЙЛОВ***, ПОЗВОЛЯЮЩИХ ИСПОЛЬЗОВАТЬ ЕГО ДЛЯ МОДЕЛИРОВАНИЯ СИСТЕМ ИЗ СПЕЦИФИЧЕСКИХ ПРЕДМЕТНЫХ ОБЛАСТЕЙ.

# ЯЗЫК МОДЕЛИРОВАНИЯ И ПРОЦЕСС МОДЕЛИРОВАНИЯ

Большинство существующих методов объектно-ориентированного анализа и проектирования (ООАП) включают как язык моделирования, так и описание процесса моделирования.

**Язык моделирования — это нотация (в основном графическая), которая используется методом для описания проектов**.

**Нотация** представляет собой совокупность графических объектов, которые используются в моделях; она является синтаксисом языка моделирования. Например, нотация диаграммы классов определяет, каким образом представляются такие элементы и понятия, как класс, ассоциация и множественность.

**Процесс** — это описание шагов, которые необходимо выполнить при разработке проекта.

# УНИФИЦИРОВАННЫЙ ЯЗЫК МОДЕЛИРОВАНИЯ UML (UNIFIED MODELING LANGUAGE)

***Унифицированный язык моделирования UML (Unified Modeling** Language)* представляет собой язык для определения, представления, проектирования и документирования программных систем, организационно-экономических систем, технических систем и других систем различной природы. UML содержит стандартный набор диаграмм и нотаций самых разнообразных видов.

Создание UML фактически началось в конце 1994 г Гради Бучем и Джеймсом Рамбо. К концу 1995 г. они создали первую спецификацию объединенного метода, названного ими Unified Method, версия 0.8. К ним при-соединился создатель метода OOSE (Object-Oriented Software Engineering) Ивар Якобсон. Таким образом, UML является прямым объединением и унификацией методов Буча, Рамбо и Якобсона, однако дополняет их новыми возможностями.

# ГЛАВНЫЕ ЦЕЛИ В РАЗРАБОТКЕ UML

Главными в разработке UML были следующие цели:

- предоставить пользователям готовый к использованию выразительный язык визуального моделирования, позволяющий разра-батывать осмысленные модели и обмениваться ими;
- предусмотреть механизмы расширяемости и специализации для расширения базовых концепций;
- обеспечить независимость от конкретных языков программирования и процессов разработки;
- обеспечить формальную основу для понимания этого языка моделирования (язык должен быть одновременно точным и доступным для понимания, без лишнего формализма);
- стимулировать рост рынка объектно-ориентированных инструментальных средств;
- интегрировать лучший практический опыт.

UML В ПЕРВУЮ ОЧЕРЕДЬ - ЭТО СПЕЦИФИКАЦИИ.

**СПЕЦИФИКАЦИЯ** - ПОДРОБНОЕ ОПИСАНИЕ СИСТЕМЫ, ПОЛНОСТЬЮ ОПРЕДЕЛЯЕТ ЕЕ ЦЕЛЬ И ФУНКЦИОНАЛЬНЫЕ ВОЗМОЖНОСТИ.

ЗАКАЗЧИК И РАЗРАБОТЧИК ИМЕЮТ ОБЫЧНО РАЗНОЕ ПОНИМАНИЕ СМЫСЛА ЭТОГО ПОНЯТИЯ. ЕСТЬ ЕЩЕ АНАЛИТИКИ, МЕНЕДЖЕРЫ, БИЗНЕС-КОНСУЛЬТАНТЫ... КАЖДЫЙ ИЗ НИХ НАЗЫВАЕТ СПЕЦИФИКАЦИИ ПО-СВОЕМУ: ПОСТАНОВКА ЗАДАЧИ, ТРЕБОВАНИЯ ПОЛЬЗОВАТЕЛЯ, ТЕХНИЧЕСКОЕ ЗАДАНИЕ, ФУНКЦИОНАЛЬНАЯ СПЕЦИФИКАЦИЯ, АРХИТЕКТУРА СИСТЕМЫ...

ОНИ СПЕЦИАЛИСТЫ В РАЗНЫХ ПРЕДМЕТНЫХ ОБЛАСТЯХ, ГОВОРЯТ НА СВОИХ ЯЗЫКЕ И НЕ ПОНИМАЮТ ДРУГ ДРУГА.

ВОЗНИКАЕТ ПРОБЛЕМА - НА РИСУНКЕ, ЕЕ МОЖЕТ РЕШИТЬ НАЛИЧИЕ ЕДИНОГО УНИФИЦИРОВАННОГО СРЕДСТВА СОЗДАНИЯ СПЕЦИФИКАЦИЙ, ПРОСТОГО И ПОНЯТНОГО ДЛЯ ВСЕХ ЗАИНТЕРЕСОВАННЫХ ЛИЦ.

СПЕЦИФИКАЦИЯ РАЗРАБАТЫВАЕМОГО ПО ПРИ ИСПОЛЬЗОВАНИИ UML ОБЪЕДИНЯЕТ МОДЕЛИ: ИСПОЛЬЗОВАНИЯ, ЛОГИЧЕСКУЮ, РЕАЛИЗАЦИИ, ПРОЦЕССОВ, РАЗВЕРТЫВАНИЯ.

- ***МОДЕЛЬ ИСПОЛЬЗОВАНИЯ*** - ОПИСАНИЕ ФУНКЦИОНАЛЬНОСТИ ПО С Т.ЗР. ПОЛЬЗОВАТЕЛЯ.
- ***ЛОГИЧЕСКАЯ МОДЕЛЬ*** - ОПИСЫВАЕТ КЛЮЧЕВЫЕ АБСТРАКЦИИ ПО (КЛАССЫ, ИНТЕРФЕЙСЫ), Т.Е. СРЕДСТВА, ОБЕСПЕЧИВАЮЩИЕ ТРЕБУЕМУЮ ФУНКЦИОНАЛЬНОСТЬ.
- ***МОДЕЛЬ РЕАЛИЗАЦИИ*** - ОПРЕДЕЛЯЕТ РЕАЛЬНУЮ ОРГАНИЗАЦИЮ ПРОГРАММНЫХ МОДУЛЕЙ В СРЕДЕ РАЗРАБОТКИ.
- ***МОДЕЛЬ ПРОЦЕССОВ*** - ОТОБРАЖАЕТ ОРГАНИЗАЦИЮ ВЫЧИСЛЕНИЙ, ОПЕРИРУЕТ ПОНЯТИЯМИ «ПРОЦЕССЫ» И «НИТИ». ОНА ПОЗВОЛЯЕТ ОЦЕНИТЬ ПРОИЗВОДИТЕЛЬНОСТЬ, МАСШТАБИРУЕМОСТЬ И НАДЕЖНОСТЬ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ.
- ***МОДЕЛЬ РАЗВЕРТЫВАНИЯ*** - ПОКАЗЫВАЕТ ОСОБЕННОСТИ РАЗМЕЩЕНИЯ

КАЖДАЯ МОДЕЛЬ ХАРАКТЕРИЗУЕТ ОПРЕДЕЛЕННЫЙ АСПЕКТ ПРОЕКТИРУЕМОЙ СИСТЕМЫ, А ВМЕСТЕ ОНИ СОСТАВЛЯЮТ ОТНОСИТЕЛЬНО ПОЛНУЮ МОДЕЛЬ РАЗРАБАТЫВАЕМОГО ПО

UML - 9 ДОПОЛНЯЮЩИХ ДРУГ ДРУГА ДИАГРАММ, ВХОДЯЩИХ В РАЗЛИЧНЫЕ МОДЕЛИ:

- ДИАГРАММЫ ВАРИАНТОВ ИСПОЛЬЗОВАНИЯ;
- ДИАГРАММЫ КЛАССОВ;
- ДИАГРАММЫ ПАКЕТОВ;
- ДИАГРАММЫ ПОСЛЕДОВАТЕЛЬНОСТЕЙ ДЕЙСТВИЙ;
- ДИАГРАММЫ КООПЕРАЦИИ;
- ДИАГРАММЫ ДЕЯТЕЛЬНОСТЕЙ;
- ДИАГРАММЫ СОСТОЯНИЙ ОБЪЕКТОВ;

ВСЕ ДИАГРАММЫ ПО ВОЗМОЖНОСТИ ИСПОЛЬЗУЮТ ЕДИНУЮ ГРАФИЧЕСКУЮ НОТАЦИЮ, ЧТО ОБЛЕГЧАЕТ ИХ ПОНИМАНИЕ.

ПОМИМО УКАЗАННЫХ ДИАГРАММ, КАК И ПРИ СТРУКТУРНОМ ПОДХОДЕ, СПЕЦИФИКАЦИЯ ОБЯЗАТЕЛЬНО ВКЛЮЧАЕТ СЛОВАРЬ ТЕРМИНОВ, РАЗЛИЧНОГО РОДА ОПИСАНИЯ И ТЕКСТОВЫЕ СПЕЦИФИКАЦИИ.

КОНКРЕТНЫЙ НАБОР ДОКУМЕНТАЦИИ ОПРЕДЕЛЯЕТСЯ РАЗРАБОТЧИКОМ.

UML И RATIONAL UNIFIED PROCESS ПОДДЕРЖИВАЮТСЯ ПАКЕТОМ RATIONAL ROSE ФИРМЫ RATIONAL SOFTWARE CORPORATION.

РЯД ДИАГРАММ UML МОЖНО ПОСТРОИТЬ СРЕДСТВАМИ MICROSOFT VISUAL MODELER И ДРУГИХ CASE-СРЕДСТВ.