Учебный курс

Язык UML в анализе и проектировании программных систем и бизнес-процессов

Лекция 1

Базовые принципы и понятия технологии разработки объектно-ориентированных информационных систем на основе UML 2

Автор:

Леоненков Александр Васильевич

кандидат технических наук,
старший научный сотрудник

Причины неудачных проектов

- Недостаточно адекватное управление требованиями
- Несогласованность требований, проектных решений и реализации
- Жесткая архитектура ПО
- Нарастающая сложность ПО
- Неточная и противоречивая коммуникация
- Недостаточное тестирование
- Субъективное отношение к приоритетам отдельных артефактов проекта
- Игнорирование рисков и отсутствие процедур управления рисками
- Бесконтрольное внесение изменений в артефакты проекта
- Недостаточное использование CASE-средств и средств поддержки отдельных этапов проекта

Отсутствие моделей при разработке ПО

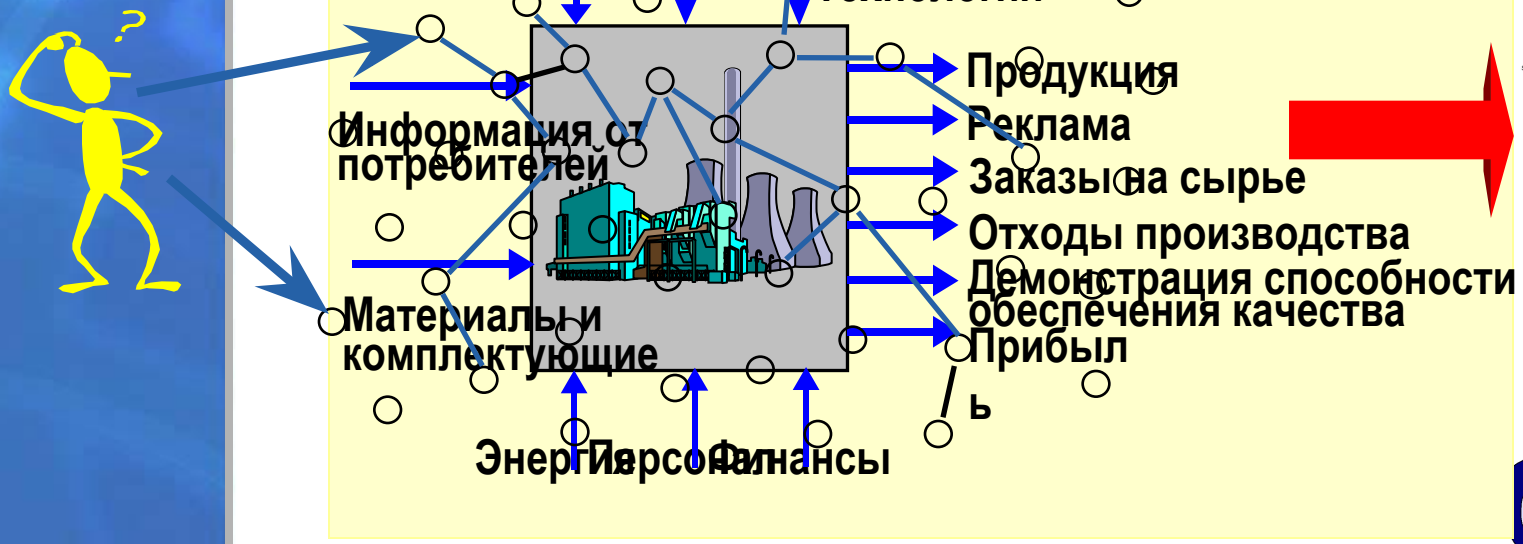
- Не позволяет справиться с растущей сложностью разрабатываемых программных систем
- Не позволяет эффективно управлять разработкой в условиях изменяющихся требований
- Создает барьеры непонимания: аналитик не понимает руководителя проекта, разработчик – аналитика, тестировщик – разработчика и пр.
- Не позволяет обеспечить контроль изменений в процессе выполнения работ
- Не позволяет избежать субъективности в оценке качества разрабатываемых продуктов
- **Модель** (model) — абстракция физической системы, рассматриваемая с определенной точки зрения и представленная на некотором языке или в графической форме

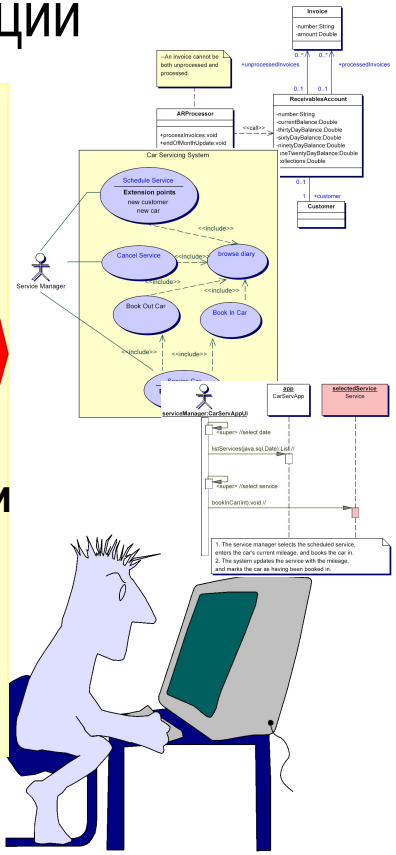
Лучшие практики разработки ПО

- Использование визуальных моделей при разработке ПО
- Итеративная разработка ПО
- Управление требованиями
- Управление изменениями и конфигурацией артефактов ПО
- Использование компонентных архитектур
- Непрерывное тестирование и верификация качества ПО
- Использование паттернов проектирования
- Использование CASE-средств и RAD-средств
- Управление рисками:
  - Технологическими рисками
  - Связанными с требованиями
  - Связанными с квалификацией персонала проекта
  - Политическими рисками

Что такое визуальное моделирование?

Визуальное моделирование есть моделирование с использованием некоторой графической нотации





На входе – Неструктурированная информация

На выходе – Модели ПО и бизнес-процессов

Основные понятия визуального моделирования

- **Нотация** – система условных обозначений для графического представления визуальных моделей
- **Семантика** – система правил и соглашений, определяющая смысл и интерпретацию конструкций некоторого языка
- **Методология** – совокупность принципов моделирования и подходов к логической организации методов и средств разработки моделей
- **CASE (Computer Aided Software Engineering) – м**етодология разработка программного обеспечения, основанная на комплексном использовании компьютеров не только для написания исходного кода, но и для анализа и моделирования соответствующей предметной области
- **CASE-средства** (CASE-tools) – программное обеспечение, которое предназначено для разработки визуальных моделей программных систем и генерации исходного кода или схемы базы данных на некотором языке

CASE-средства

Разработка визуальных моделей сложных систем, в виду значительного объема решаемых задач, должно опираться на специальные средства программной поддержки

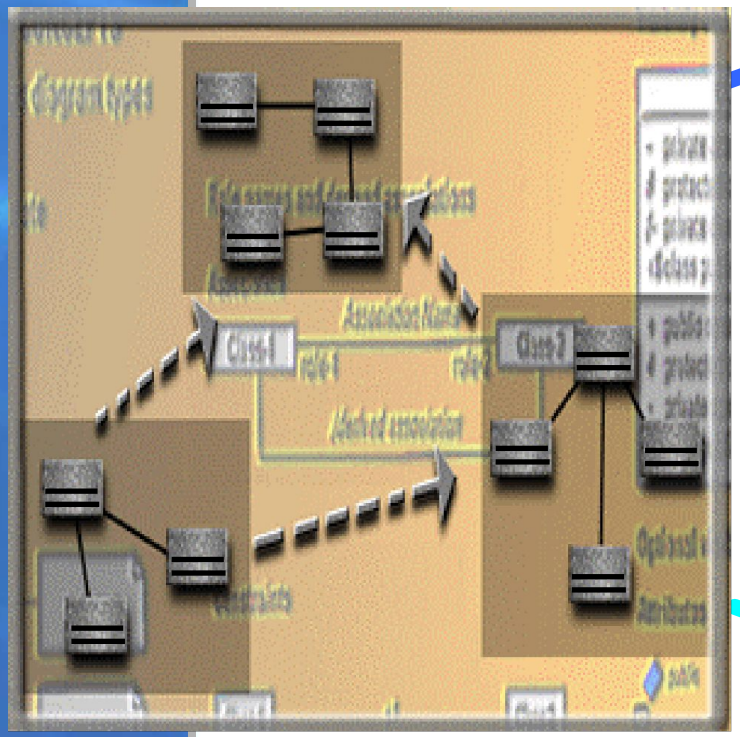
Oracle Designer

BPwin, ERwin

Rational Rose

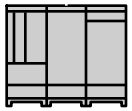
- 1-е поколение: генерация схем БД (Oracle Designer 2000, ERwin)
- 2-е поколение: генерация программного кода (Borland Together Designer 2005)
- 3-е поколение: прямая и обратная кодогенерация (IBM Rational Rose 2002/2003, Borland Together Developer 2005, Sparx Enterprise Architect)
- 4-е поколение: синхронизация программного кода и моделей (IBM Rational Software Architect 6/7, Borland Together Architect 2006, Borland Development Studio 2006)

Визуальные модели представляют архитектуру программных систем



Бизнес-логика
(C++, Java)

Интерфейсы
пользователя
(Delphi,
Visual Basic,
Java)



Базы данных
(SQL)

Визуальная модель системы не должна зависеть от языка ее реализации!

Визуальные модели являются средством коммуникации

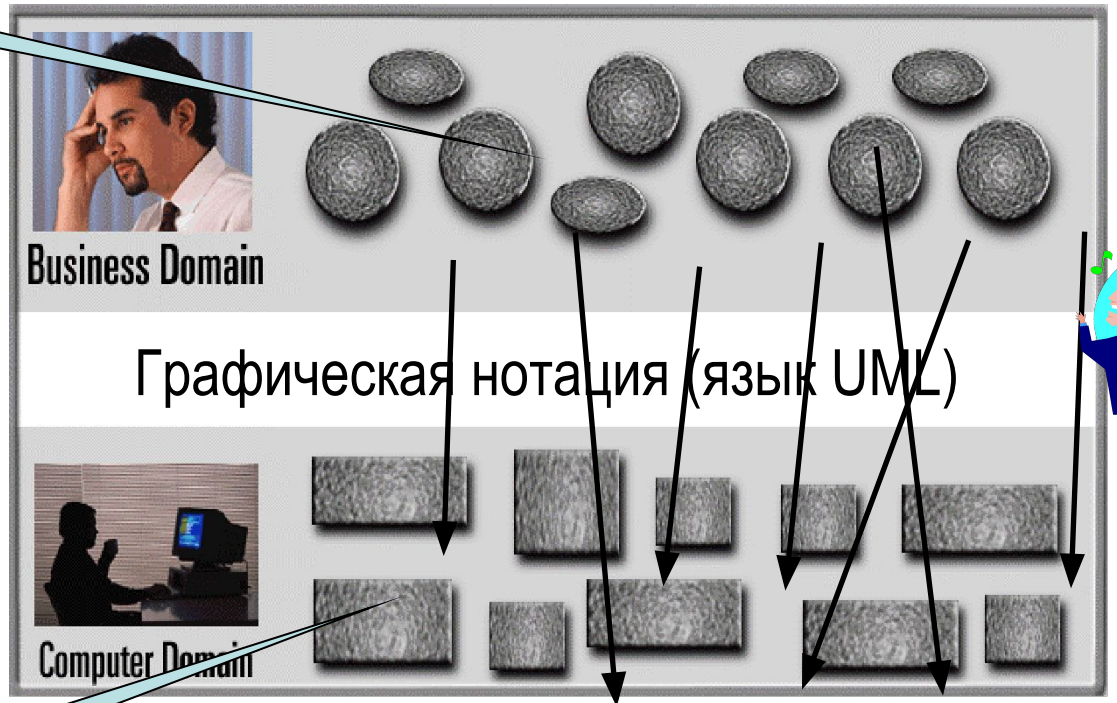
Артефакты БП

Визуальные модели описывают бизнес-процессы

Визуальные модели используются для проектирования и разработки программных систем

Артефакты ПО

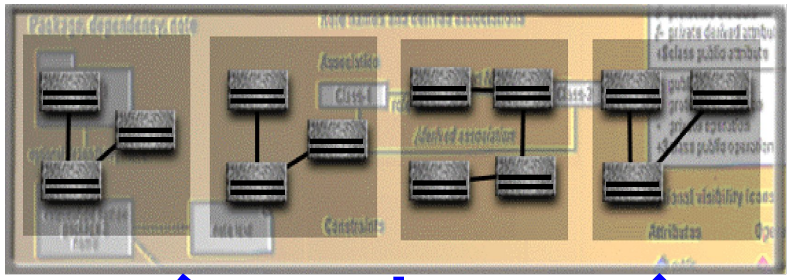
Бизнес-аналитики, системные аналитики, архитекторы, CIO, MIS, CPO



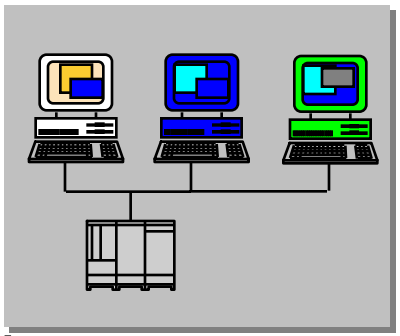
Программисты, тестировщики, менеджеры проектов

Визуальные модели – основа многократного использования кода

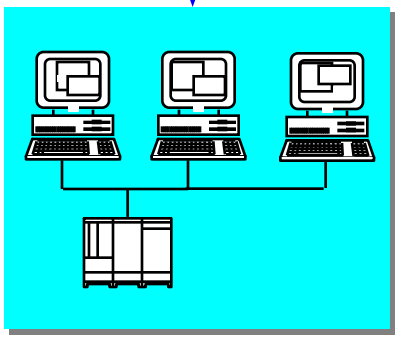
Моделирование охватывает существенные (основные, релевантные) аспекты структуры и поведения системы



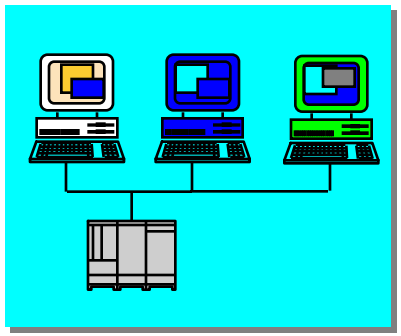
Многократно используемые компоненты (Reusable Components)



Интернет порталы



ERP Системы



Базы данных

ООП – основные понятия

- ***Объектно-ориентированное программирование*** (Object-Oriented Programming) — совокупность принципов, технологии и инструментальных средств для создания программных систем, в основу которых закладывается архитектура взаимодействия объектов
- ***Абстракция*** — характеристика сущности, которая отличает ее от других сущностей
- ***Наследование*** — принцип, в соответствии с которым знание о более общей категории разрешается применять для более частной категории
- ***Инкапсуляция*** — сокрытие отдельных деталей внутреннего устройства классов от внешних по отношению к нему объектов или пользователей
- ***Полиморфизм*** — свойство элементов модели с одинаковыми именами иметь различное поведение

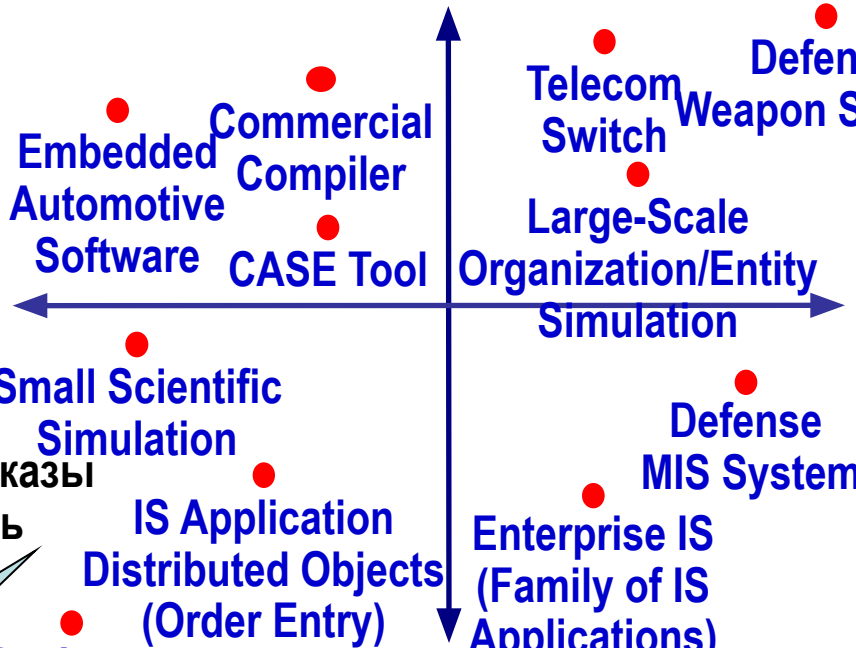
ООАП – основные понятия

- ***Объектно-ориентированный анализ и проектирование*** (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
- ***Диаграмма*** (diagram) — графическое представление совокупности элементов модели в форме связного графа, вершинам и ребрам (дугам) которого приписывается определенная семантика
- Нотация канонических диаграмм является основным средством разработки моделей на языке UML

Классификация проектов по сложности

Высокая техническая сложность
- **Встроенные системы реального времени**
- **Распределенные высоконадежные системы**
- **Высокопроизводительные системы**

Использование языка UML обязательно!



Низкая сложность управления
- **Малый масштаб**
- **Неформальные заказы**
- **Один пользователь**
- **“Продукты”**

Высокая сложность управления
- **Большой масштаб**
- **Контрактные заказы**
- **Много пользователей**
- **«Проекты»**

Использование языка UML не обязательно

Низкая техническая сложность
- **Использование макроязыков или 4GL**
- **Реинжиниринг приложений баз данных**
- **Разработка учетно-расчетных приложений**

Классификация проектов по типу приложений

Использование языка UML обязательно!

Использование языка UML в проектах по отраслевой принадлежности

- **Банки и инвестиционные фонды**
- **Связь и телекоммуникации**
- **Нефтегазовая промышленность**
- Страховые фонды
- Энергетика
- Машиностроение
- Торговля
- Фармацевтическая промышленность
- Оборонная промышленность
- Федеральная таможенная служба
- Учебные заведения

Средний проект по разработке ПО:

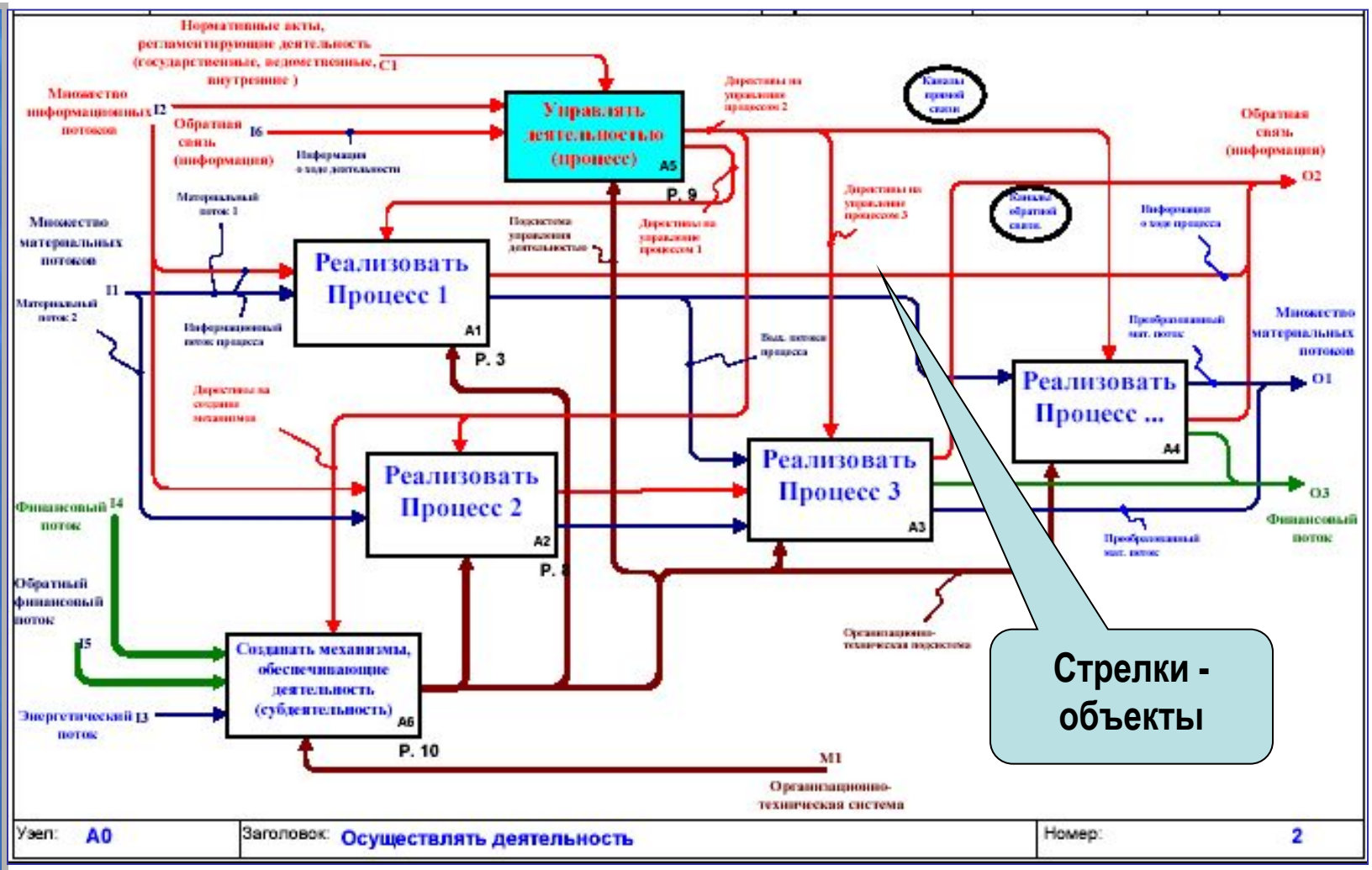
- 5-10 человек
- 10-15 месяцев
- 10-15 внешних интерфейсов
- Незначительная неопределенность и риски

Взаимосвязь нотации, методологии и инструментальных средств

Графические нотации моделирования, используемые в России

- **UML** (Unified Modeling Language) – отраслевой стандарт OMG, поддерживают более 50 CASE-средств, основной инструмент IBM Rational Rose/ IBM RSA (IBM Rational Software)
- **IDEF** – семейство нотаций, стандарт МО США, рекомендован Правительством РФ для применения в государственных учреждениях, основной инструмент AllFusion Pricess Modeller (Computer Associations)
- **ARIS** (*ARchitecture of Integrated Information Systems*) – методология и нотация для профессионального моделирования бизнес-процессов, инструмент ARIS Toolset (IDS Scheer AG)

Пример визуальной модели в нотации IDEF



IDEF не объектно-ориентированная нотация!

Взаимосвязь нотации UML, методологии и инструментальных средств

Нотация – UML 1.x

RUP®
Rational Unified Process®

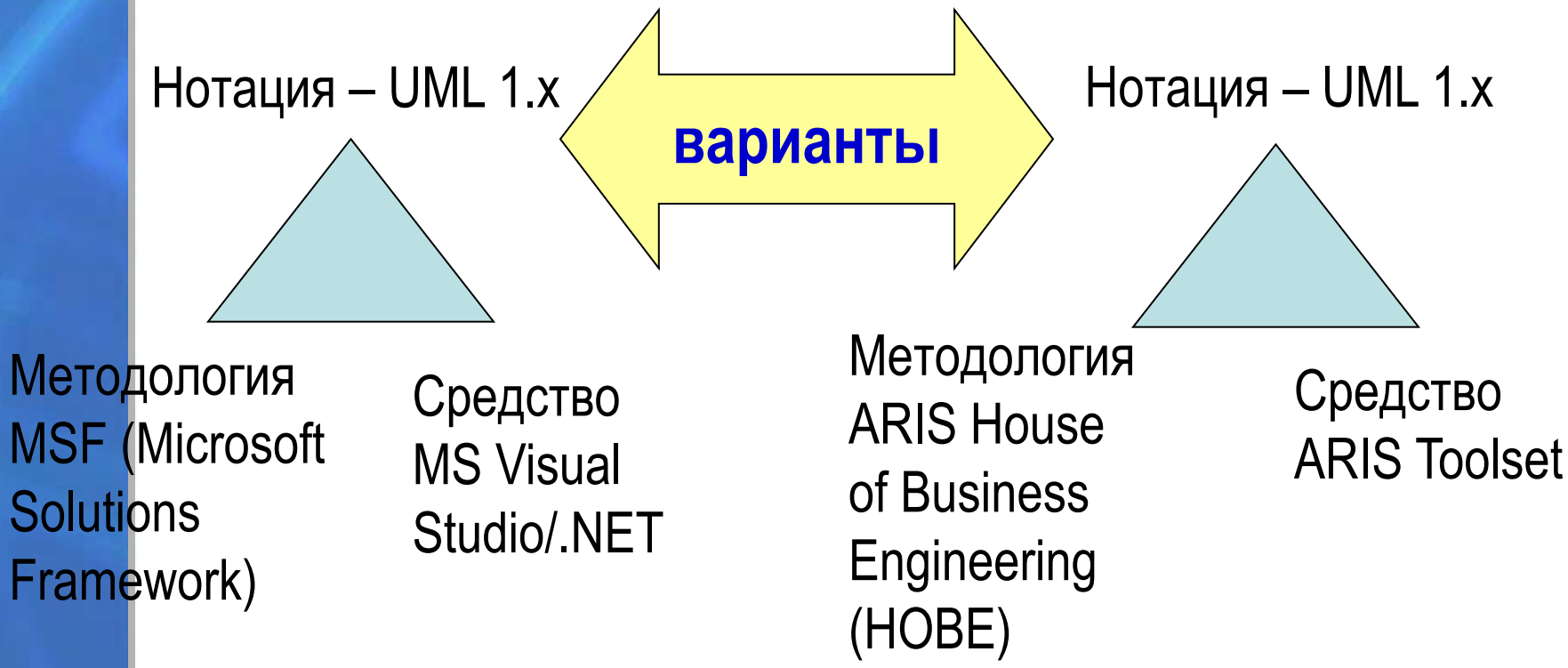
Rational
the software development company

Методология - RUP

Средство – IBM Rational Rose

+ дополнительная интеграция с линейкой продуктов IBM Rational

Взаимосвязь нотации UML, методологии и инструментальных средств



Взаимосвязь нотации UML, методологии и инструментальных средств

Нотация – UML 2.x

Нотация - UML 2.x

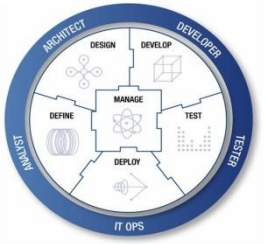
варианты

Методология RUP

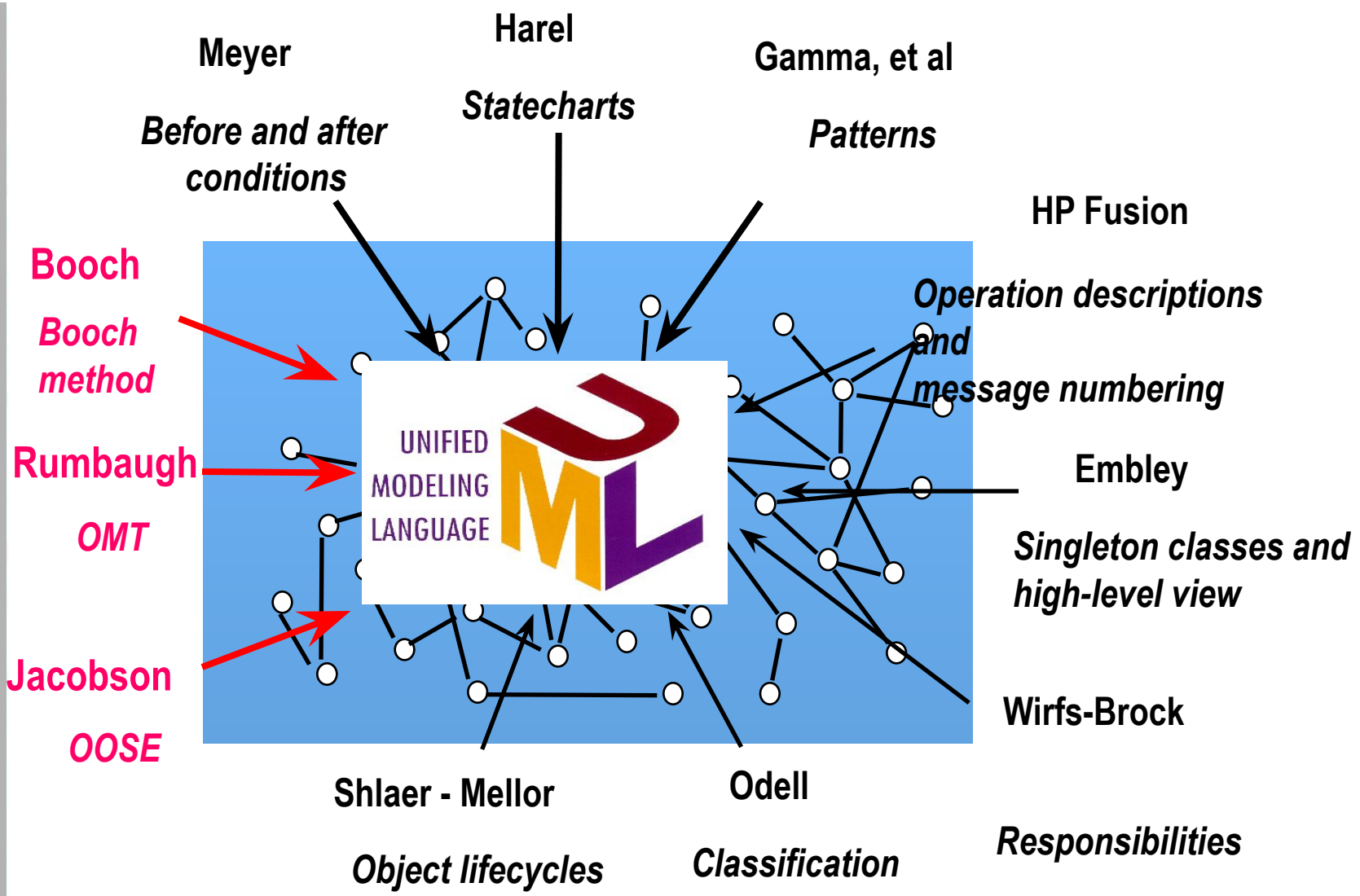
Средство IBM Rational Software Architect

Методология ALM (Application Lifecycle Management)

Средство Borland Together Architect 2006



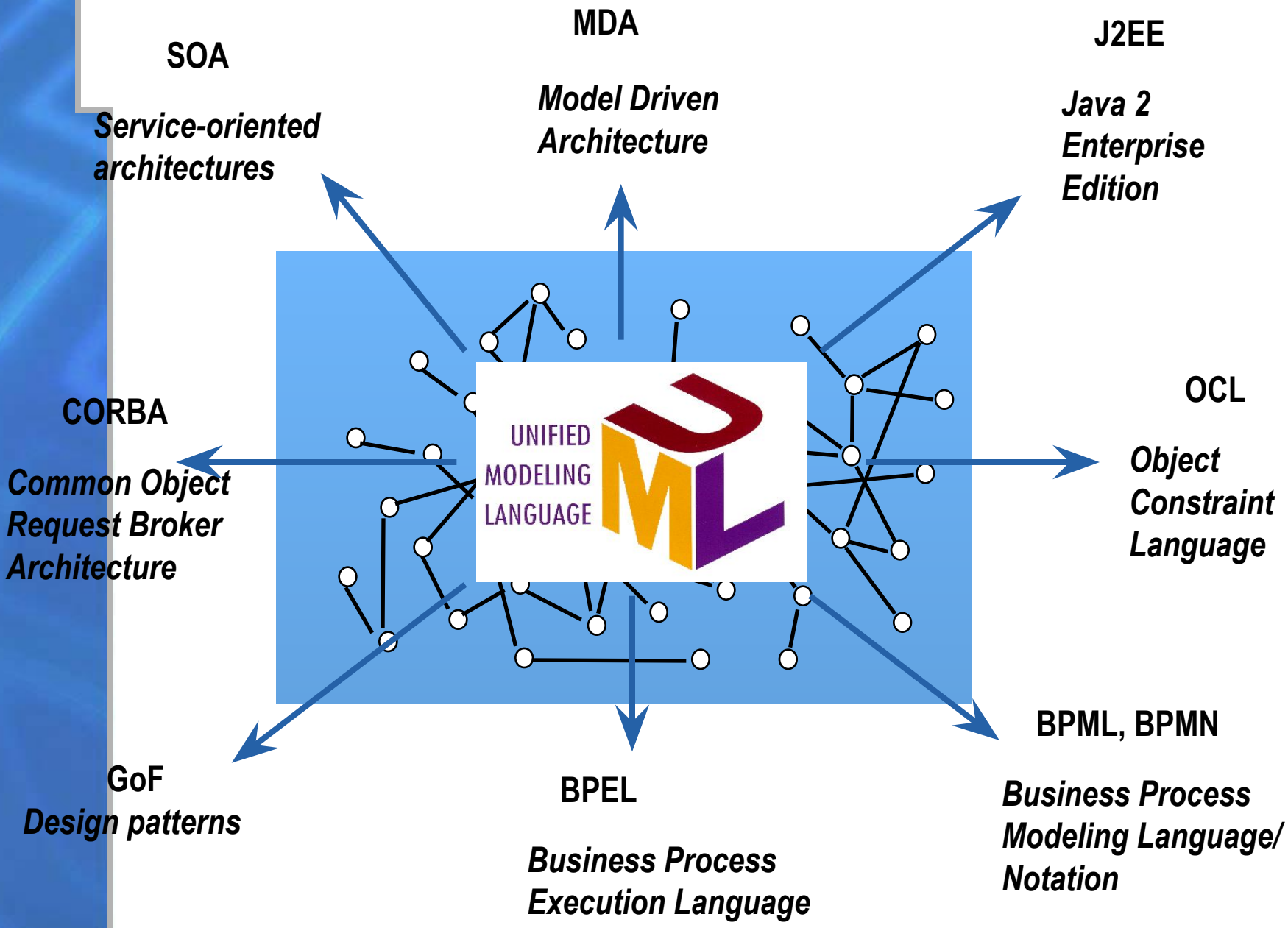
«Война методов» конца 1980 гг.



Популярные графические нотации визуального моделирования (конец 80-х гг.)

- **ERD** (Entity-Relationship Diagrams) – диаграммы «сущность-связь»
- **DFD** (Data Flow Diagrams) – диаграммы потоков данных, обеспечивающих анализ требований и функциональное проектирование информационных систем
- **STD** (State Transition Diagram) – диаграммы перехода состояний для проектирования систем реального времени
- **SADT** (Structured Analysis and Design Technique) – технология структурного анализа и проектирования
- **ICAM** (Integrated Computer Aided Manufacturing) – интегрированное компьютерное производство
- **FDD** (Functional Decomposition Diagrams) – диаграммы функциональной декомпозиции
- **Структурные карты** Джексона и Константайна – проектирование межмодульных взаимодействий и внутренней структуры объектов

Язык UML и современные технологии



Основные разработчики языка UML (Three amigos)

Grady Booch
Гради Буч

Dr. James Rumbaugh
Джеймс Рамбо
(Джим Румбах)

Dr. Ivar Jacobson
Айвар Джекобсон
(Ивар Якобсон)

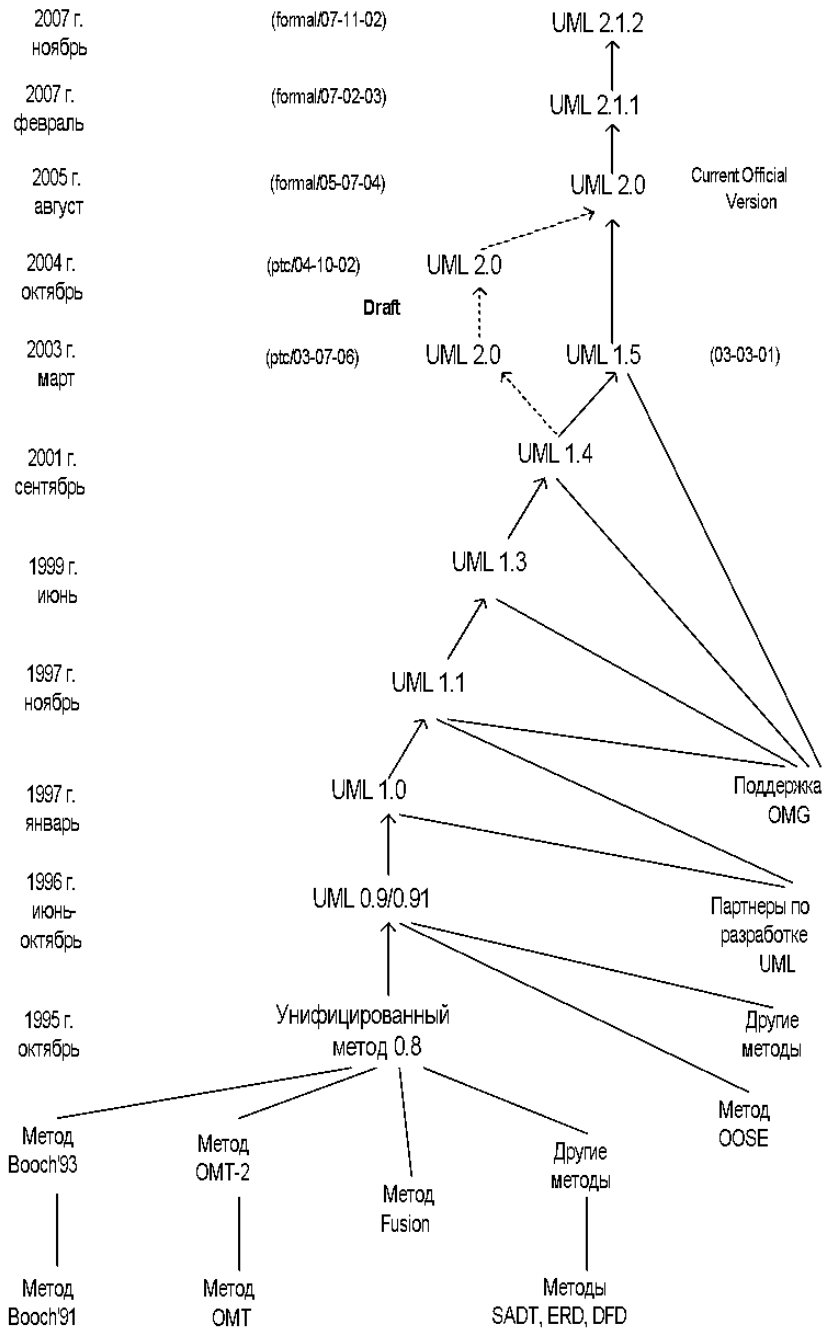
- **OMG** (Object Management Group) — название консорциума, созданного в 1989 году для разработки индустриальных стандартов с их последующим использованием в процессе создания масштабируемых неоднородных распределенных объектных сред.
- В настоящее время входит более 800 софтверных компаний
- Официальный сайт: www.omg.org

Rational
the software development company

История развития языка UML

Спецификация языка UML 2.1.2:

- Суперструктура: 07-11-02.pdf – 736 стр.
- Инфраструктура: 07-02-04.pdf – 218 стр.
- Object Constrain Language v.2.0: 2005-06-06.pdf – 185 стр.
- Diagram Interchange: 03-07-03.pdf – 34 стр.
- Model Driven Architecture 03-06-01.pdf – 62 стр.



Основные разработчики языка UML 2

- Don Baisley
- Morgan Bjorkander
- Conrad Bock
- Steve Cook
- Philippe Desfray
- Nathan Dykman
- Anders Ek
- David Frankel
- Eran Gery
- Oystein Haugen
- Sridhar Iyengar
- Cris Kobryn
- Birger Moller-Pedersen
- James Odell
- Gunnar Overgaard
- Karin Palmkvist
- Guus Ramackers
- **Jim Rumbaugh**
- Bran Selic
- Thomas Weigert
- Larry Williams

Определение языка UML

***Unified Modeling Language* —** унифицированный язык моделирования для описания, визуализации и документирования объектно-ориентированных систем в процессе их анализа и проектирования

Язык UML предоставляет стандартный способ написания проектной документации на системы, включая концептуальные аспекты, такие как бизнес процессы и функции системы, а также конкретные аспекты, такие как выражения языков программирования, схемы баз данных и повторно используемые компоненты ПО

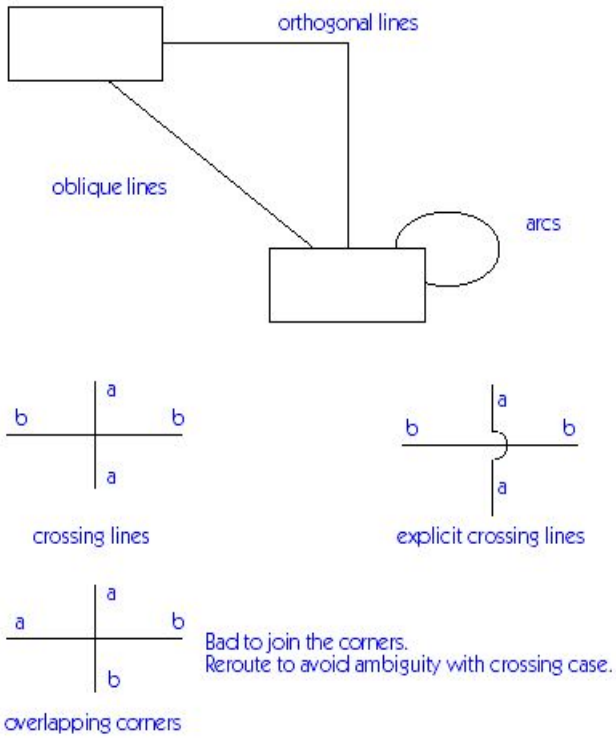
- Язык UML не является методологией
- Язык UML не является процессом
- Язык UML не является языком программирования
- Язык UML не является формальным языком

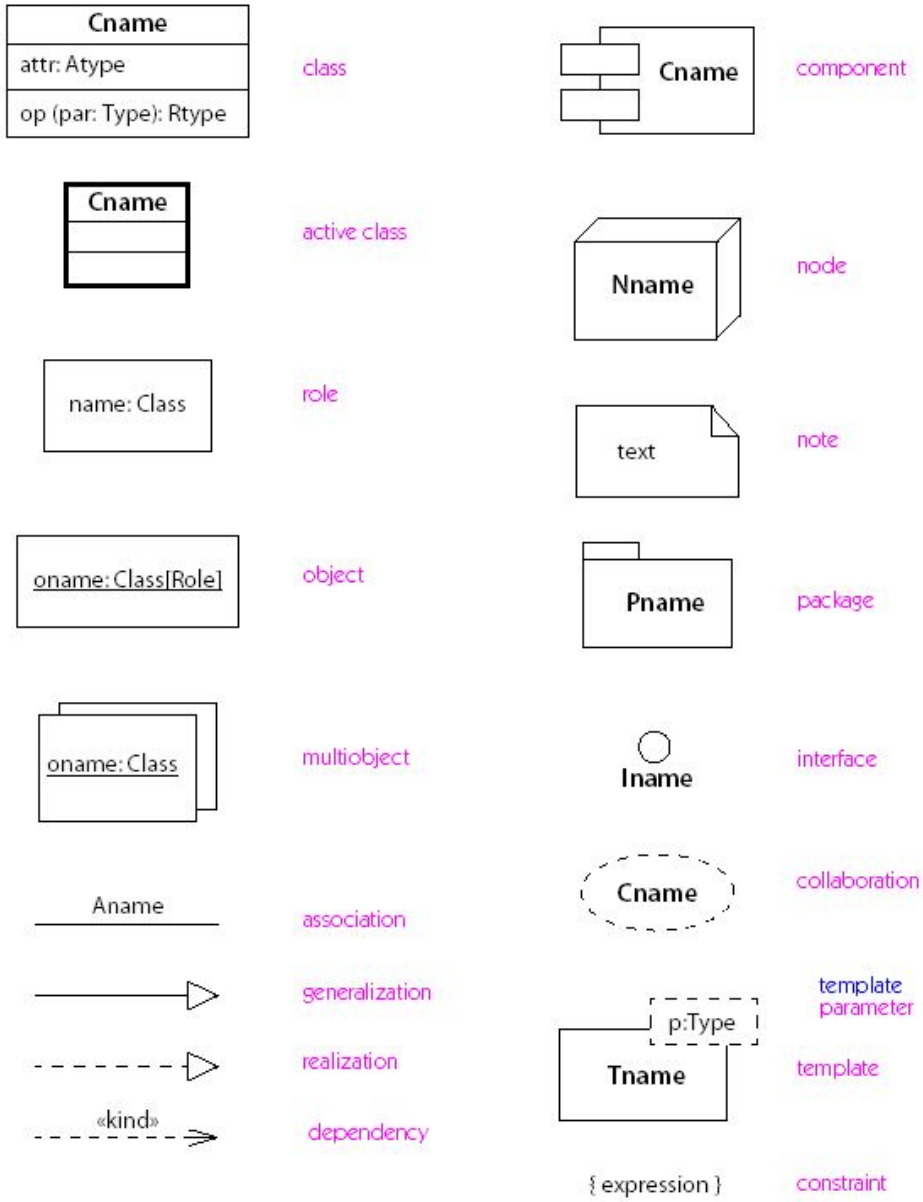
UML = нотация + семантика !

Назначение языка UML

- Предоставить разработчикам легко воспринимаемый и выразительный язык визуального моделирования, специально предназначенный для разработки и документирования моделей сложных систем различного целевого назначения
- Снабдить исходные понятия языка UML возможностью расширения и специализации для более точного представления моделей систем в конкретной предметной области
- Графическое представление моделей в нотации UML не должно зависеть от конкретных языков программирования и инструментальных средств проектирования
- Описание языка UML должно включать в себя семантический базис для понимания общих особенностей ООАП
- Способствовать распространению объектных технологий и поощрять развитие рынка программных инструментальных средств
- Интегрировать в себя новейшие и наилучшие достижения практики ООАП

Особенности изображения графического элементов диаграмм языка UML





Особенности изображения диаграмм в нотации UML

- **Графические узлы** на плоскости, которые изображаются с помощью геометрических фигур и могут иметь различную высоту и ширину с целью размещения внутри этих фигур других конструкций языка UML
- **Пути**, которые представляют собой последовательности из отрезков линий, соединяющих отдельные графические узлы
- **Значки или пиктограммы**. Значок представляет собой графическую фигуру фиксированного размера и формы, которая не может увеличивать свои размеры, чтобы разместить внутри себя дополнительные символы.
- **Строки текста**. Служат для представления различных видов информации в некоторой грамматической форме.

Общие рекомендации по изображению диаграмм в нотации языка UML

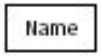
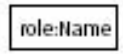
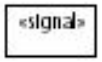
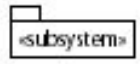
- Каждая диаграмма должна служить законченным представлением соответствующего фрагмента моделируемой предметной области
- Все сущности на диаграмме модели должны быть одного концептуального уровня
- Вся информация о сущностях должна быть явно представлена на диаграммах
- Диаграммы не должны содержать противоречивой информации
- Диаграммы не следует перегружать текстовой информацией
- Каждая диаграмма должна быть само достаточной для правильной интерпретации всех ее элементов и понимания семантики всех используемых графических символов

Противоречивость и адекватность моделей в нотации UML

- Модель, соответствующая правилам нотации или семантики языка UML называется *непротиворечивой* (well-formed model)
- Модель, нарушающая правила нотации или семантики языка UML называется *противоречивой* (ill-formed model)
- Здесь могут быть использованы формальные критерии – соответствие спецификации языка UML!
- Модель, достаточно полно и правильно отражающая предметную область или решаемую проблему называется *адекватной*
- Модель, не достаточно полно или неправильно отражающая предметную область или решаемую проблему называется *не адекватной*
- Здесь могут быть использованы только неформальные критерии – субъективное мнение экспертов!
- *Моя модель – это не ваша модель, а ваша модель – не моя…*

Классификаторы – основные элементы языка UML

Прямоугольник – основной символ для графического изображения классификатора

Classifier	*Function*	*Notation*
actor	An outside user of a system	
class	A concept from the modeled system	
class-in-state	A class restricted to being in a given state	
classifier role	A classifier restricted to a particular usage in a collaboration	
component	A physical piece of a system	
data type	A descriptor of a set of primitive values that lack identity	Name
interface	A named set of operations that characterize behavior	
node	A computational resource	
signal	An asynchronous communication among objects	
subsystem	A package that is treated as a unit with a specification, implementation, and identity	
use case	A specification of the behavior of an entity in its interaction with outside agents