Современные компьютерные технологии

Кравченко Ю.А.

Разделы дисциплины

- Основные этапы, тенденции и риски развития компьютерных технологий
- Проблемы и тенденции развития программного обеспечения и вычислительной техники
- Перспективы развития средств интеллектуального анализа данных и управления знаниями, интегрированными из разных предметных областей
- Проблемы компьютерного моделирования сложных систем
- Развитие технологий проектирования информационных, автоматизированных и автоматических систем
- Направления развития систем поддержки жизненного цикла наукоёмкой продукции

1. Основные этапы, тенденции и риски развития компьютерных технологий

- **Какие основные универсальные величины вам известны?**

Информация

- **Энергия и материя считаются основными универсальными величинами. Тем не менее, понятие информации стало фундаментальным и широко распространенным, подтверждая ее категоризацию как третей фундаментальной величины. Одной из присущих характеристик жизни является информация.**

Информационные ресурсы

- Что является средствами создания и использования информационных ресурсов?

средства создания и использования информационных ресурсов

1. научная методология, используемая в информационной сфере общества;
2. программно-аппаратные средства информатизации;
3. современные информационные технологии.

Развитие глобального процесса информатизации

1

Формирование информационной технологии, как самостоятельной научной дисциплины о методах создания высокоэффективных информационных технологии

Что является **объектом** и **предметом** исследования информационных технологий, как самостоятельной науки?

Объект исследования и предметная область

1

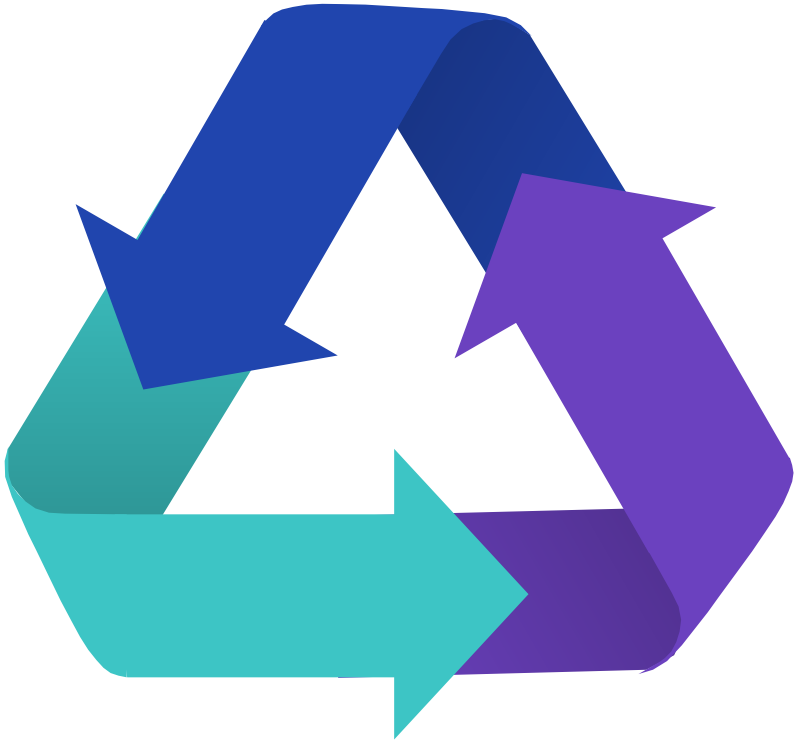
Объектом исследований **информационной технологии, как научной дисциплины, являются способы рациональной организации информационных процессов**

2

Предметом **исследований должны стать теоретические основы и методы создания способов организации, а также их проектирование и эффективная реализация**

Цикл формирования этого нового научного направления

классификация различных видов информационных технологий



разработка критериев для их сравнительного анализа и количественной оценки эффективности

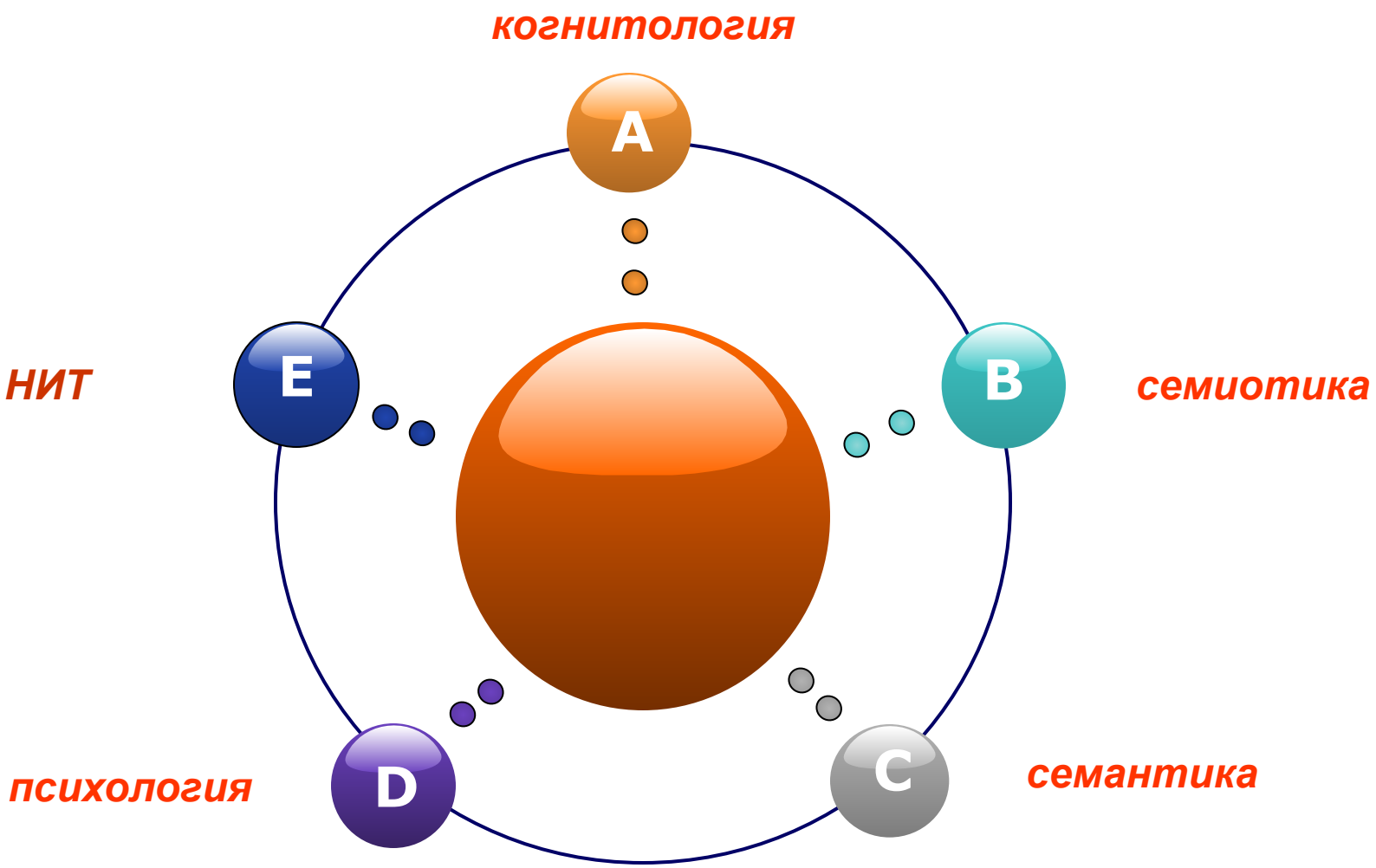
создание методов синтеза высоко-эффективных технологий

Предметная область НИТ

ЗАДАЧИ

- разработка методов структуризации и классификации Информационных технологий различного вида и назначения по их характерным признакам;
- разработка критериев эффективности информационных технологий, методов их оптимизации и сравнительной количественной оценки;
- определение перспективных направлений развития информационных технологий, а также научных методов, которые должны лежать в их основе;
- определение принципов построения перспективных средств для реализации высокоэффективных информационных технологий нового поколения.

Креативные технологии



Новая информационная технология

Технология при переводе с греческого означает **искусство, мастерство, умение**, а это процессы. Под *процессом* следует понимать определенную **совокупность действий, направленных на достижение поставленной цели**.

Новая информационная технология

Информационная технология (ИТ) - **процесс**, использующий совокупность средств и методов сбора, обработки и передачи данных (первичной информации) для получения информации нового качества о состоянии объекта, процесса или явления (информационного продукта).

Новая информационная технология

Информационная система (ИС) - взаимосвязанная совокупность средств, методов и персонала, используемых для хранения, обработки и выдачи информации в интересах достижения поставленной цели.

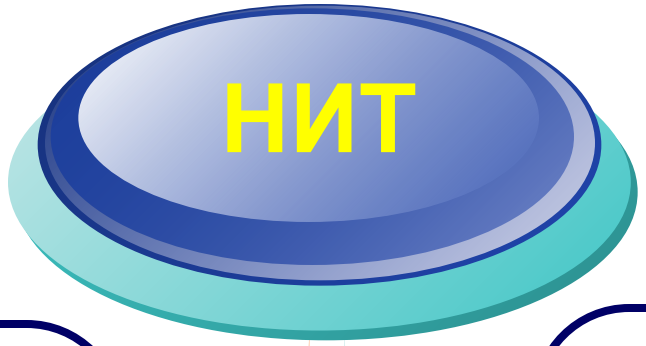
Новая информационная технология

В основу концепции **новой ИТ**, базирующейся на широком применении компьютерной техники, положены три основных принципа: **интегрированность, гибкость, интерактивность**.

Характеристики НИТ

- работа пользователя в режиме манипулирования данными
- сквозная информационная поддержка на всех этапах прохождения информации на основе интегрированной базы данных
- Автоматическая структуризация и классификация информации
- интерактивный (диалоговый) режим решения задачи с широкими возможностями для пользователя
- возможность коллективного исполнения документов на основе группы ПЭВМ, объединенных средствами коммуникации
- возможность адаптивной перестройки форм и способа представления информации в процессе решения задачи

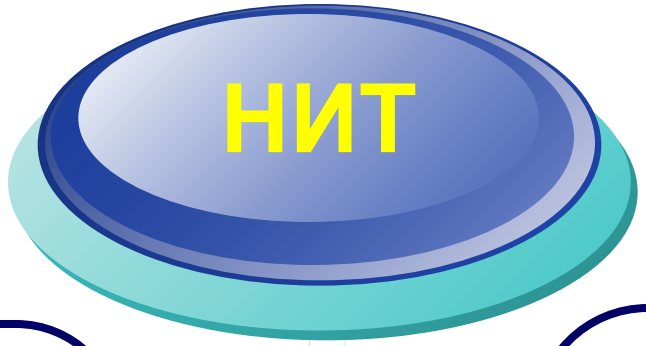
Два способа внедрения НИТ



ориентирован на существующую структуру учреждения

Ориентирован на будущую оптимизиро-ванную структуру

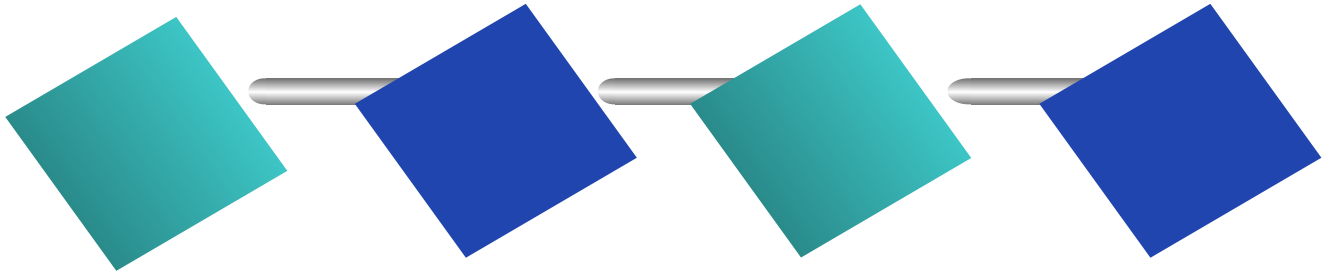
Области применения НИТ



автоматизация проектирования оперативного планирования и управления промышленным производством: системы САПР, АСУ, АСНИ

автоматизация организационного управления (учрежденческой деятельности в самых различных ее аспектах): текстовые системы, электронная почта, речевая почта, система ведения баз данных

Особенности НИТ



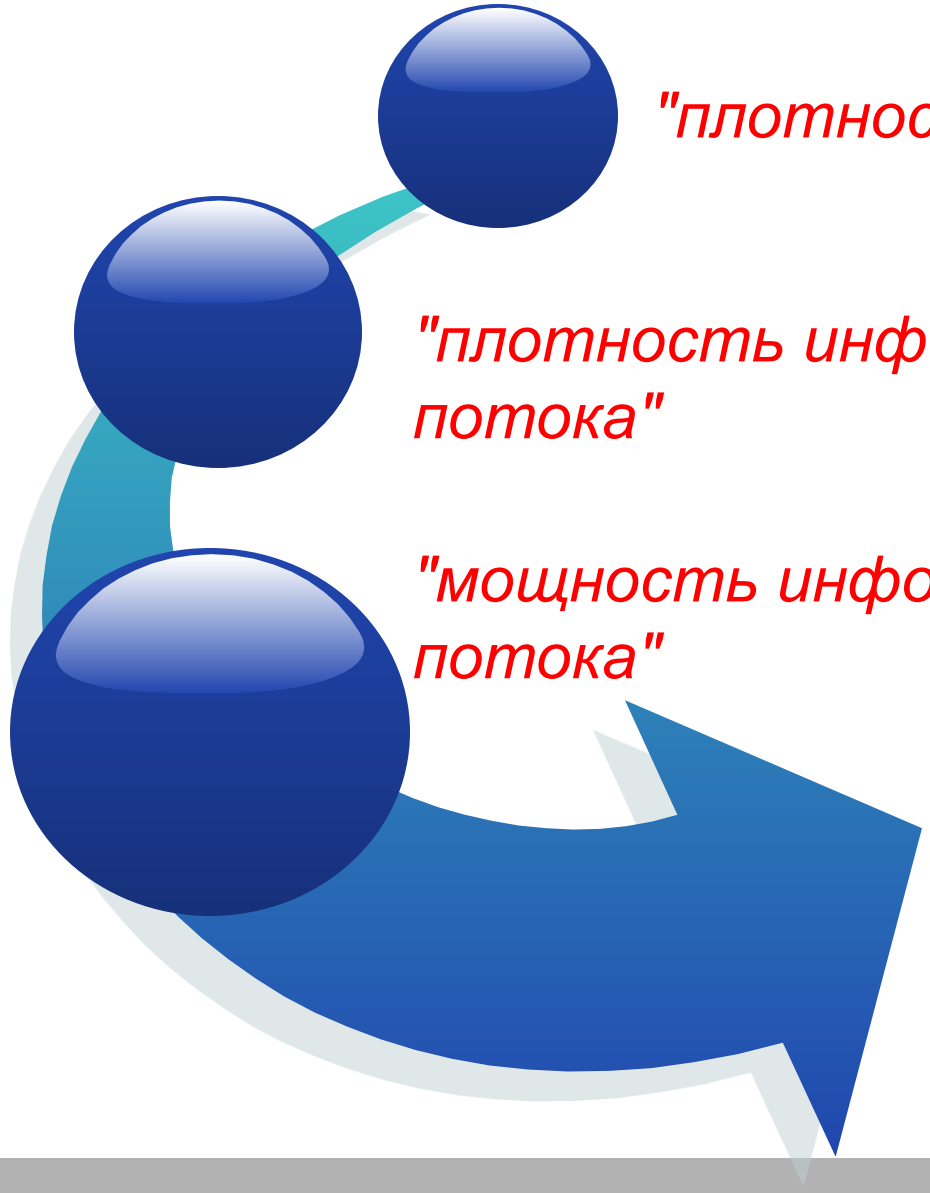
динамичность (технология, поколения технических и программных средств изменяются дважды за 5 лет)

регулярное повышение квалификации разработчиков и пользователей информационных систем

глубокое и долговременное влияние на развитие производительных сил и производственных отношений

высокая степень потенциальной эффективности в условиях: стандартизации, масштабности внедрения, своевременного внедрения новых средств и методов НИТ

Новые понятия ИТ



"плотность информации"

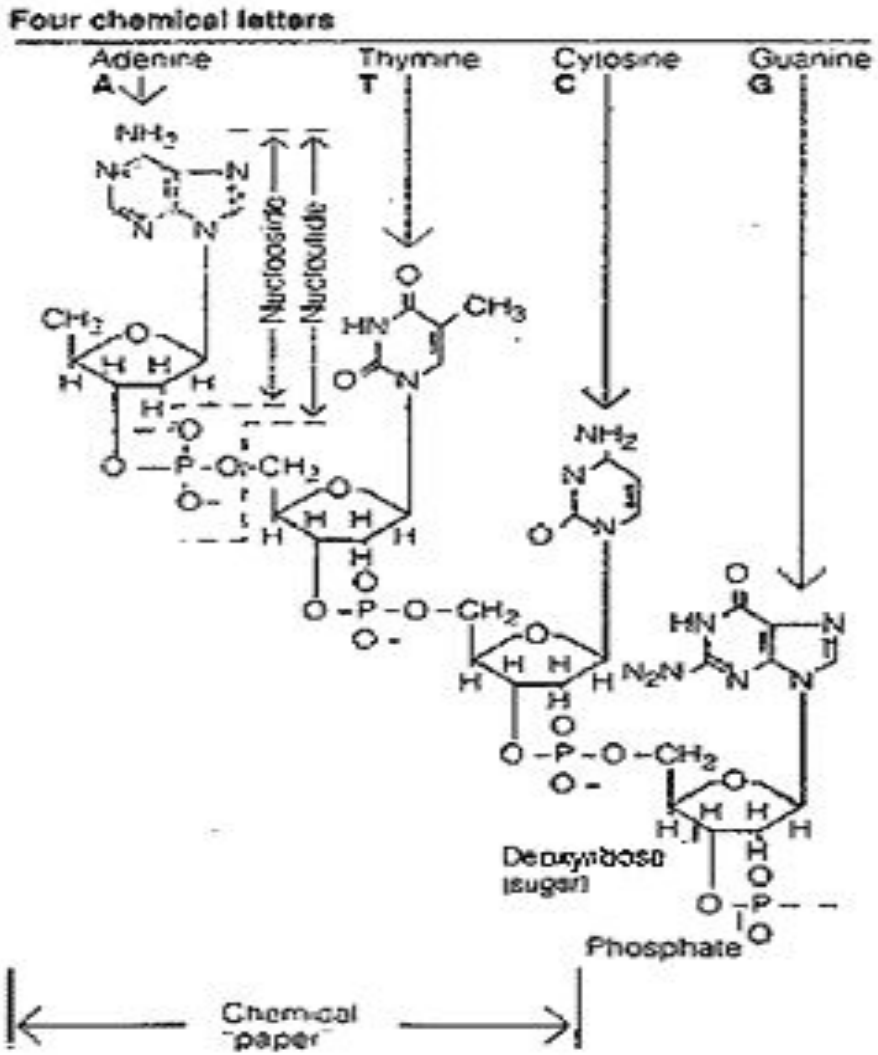
"плотность информационного потока"

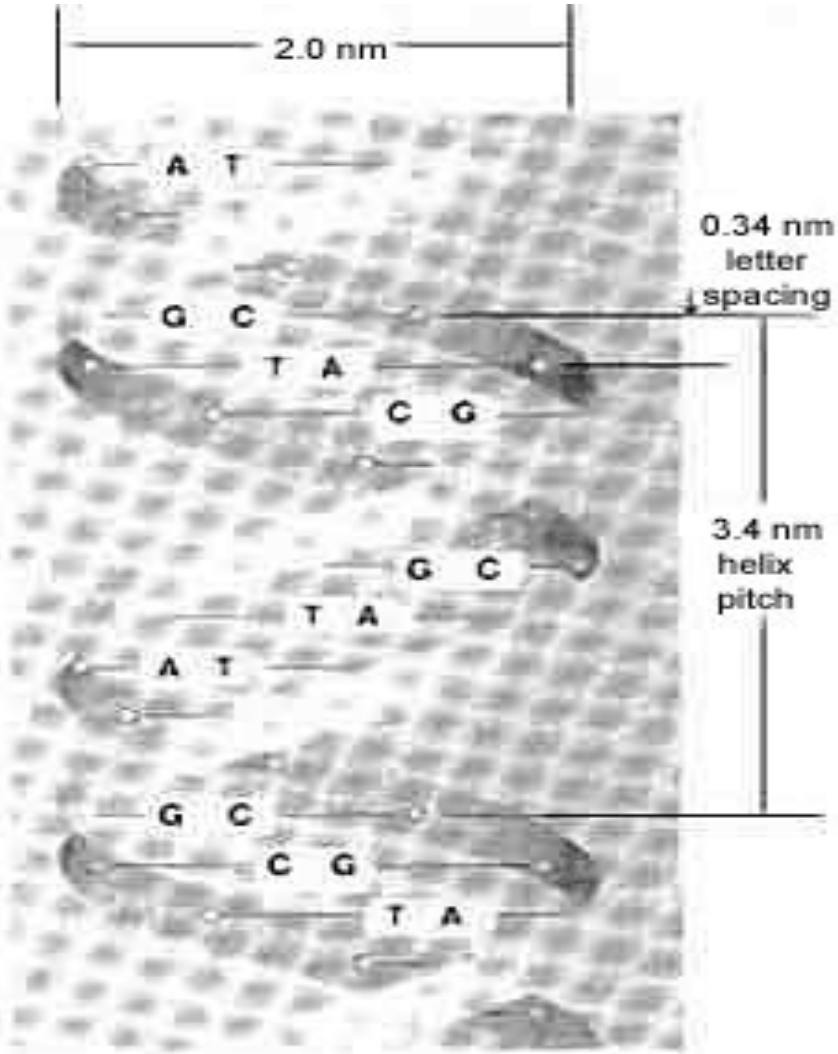
"мощность информационного потока"

Плотность информации

Где присутствует наиболее высокая известная плотность информации?

Плотность информации

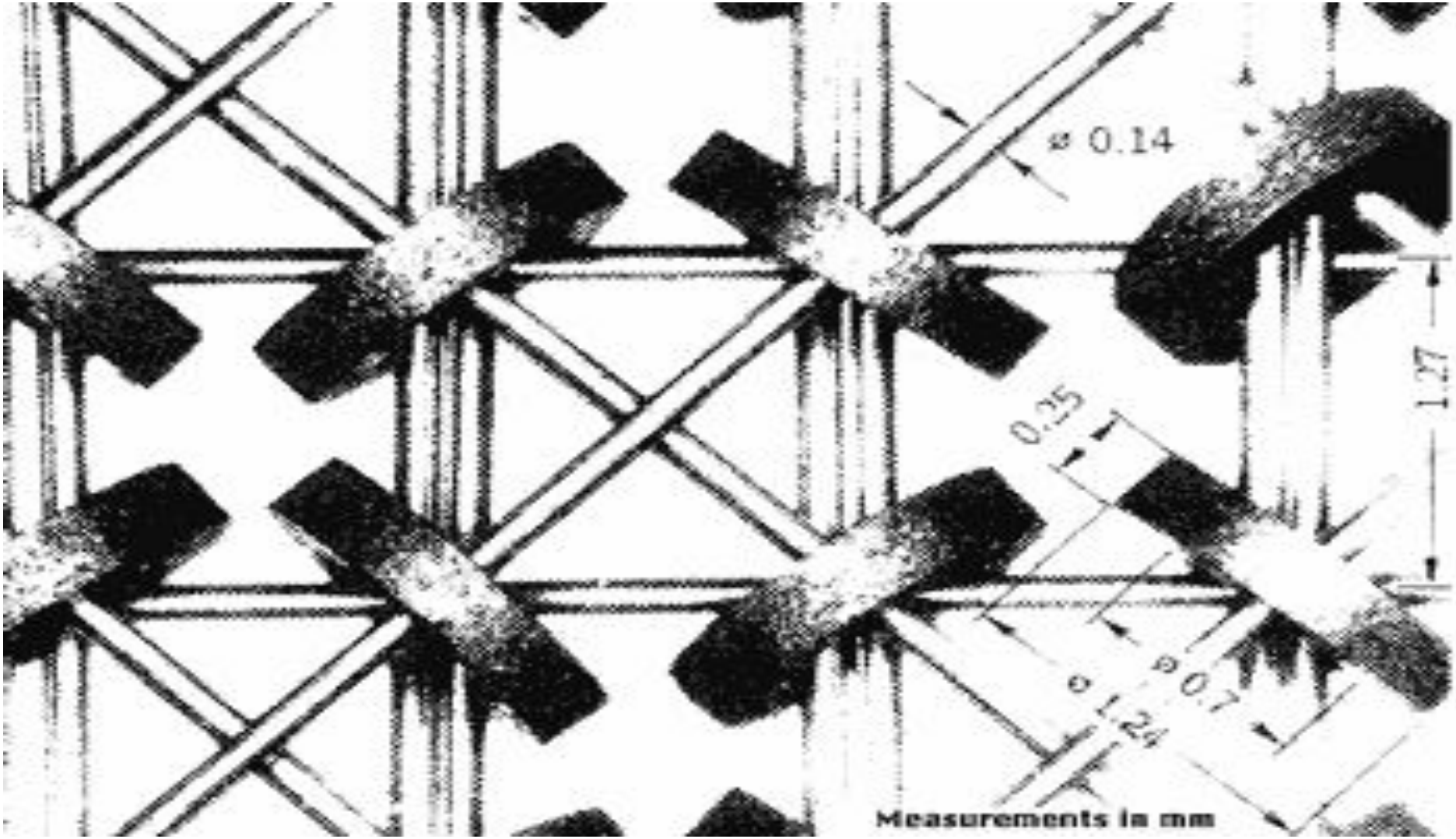




Плотность информации

Наиболее высокая известная плотность информации находится в ДНК (дезоксирибонуклеиновая кислота) молекулах живых клеток. Этот химический носитель данных всего 2 нм в диаметре и с шагом спирали 3.4 нм. Это обеспечивает объем 10.68×10^{-21} $см^3$ на виток. Каждый виток содержит 10 химических букв (нуклеотидов), что обеспечивает объемную плотность информации *0.94×10^{21} букв/$см^3$*. **В генетическом алфавите молекулы ДНК содержат только 4 нуклеотидные основы, т.е. аденин, тимин, гуанин и цитозин. Информационное содержание такой буквы составляет 2 бит/нуклеотид.**
Итак, статистическая информационная плотность - 1.88×10^{21} бит/$см^3$.

Плотность информации



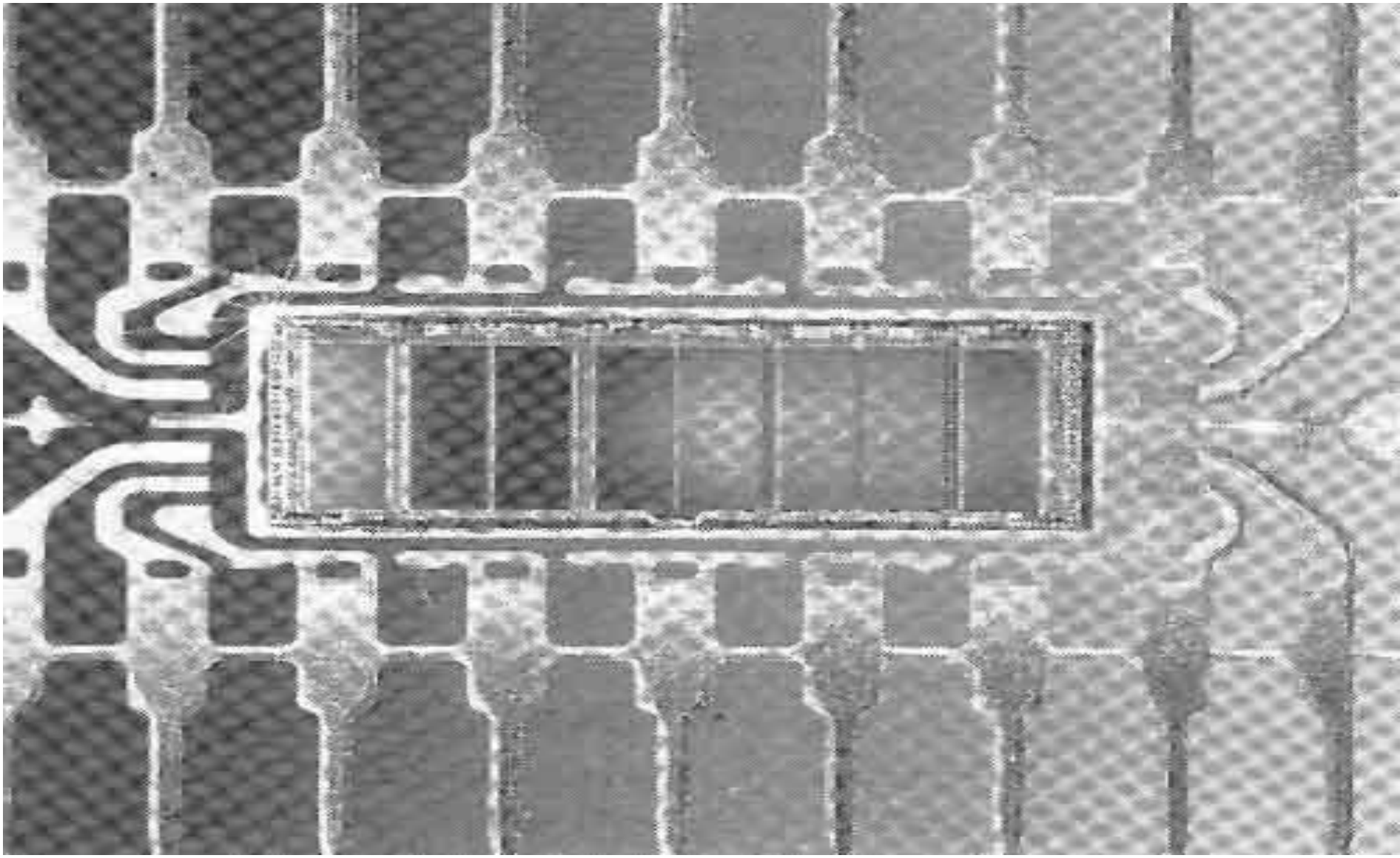
Плотность информации

Оперативная память: **раньше оперативная память могла хранить *4,096 битов в области 6,400 мм^2*.**

Это соответствует области хранения памяти *0.64 бит/мм^2*

С оперативной памятью диаметром 1.24 мм (объем хранения 7,936 мм^3), получается плостность объемного хранения *0.52 бит/мм^3*

Плотность информации



Плотность информации

1-Mbit DRAM (динамическое ОЗУ): прогрессивный прыжок от запоминающего устройства на магнитных сердечниках к полупродниковой памяти выражается в удивительных цифрах плотности хранения, 1-Mbit DRAMs позволяет хранить **1 048 576 бит в области примерно 50 кв.мм**, **что *соответствует плотности площади хранения памяти* в 21 000 бит/мм^2**.

С толщиной примерно 0.5 мм мы получаем плотность объемного хранения 42000 бит/мм^3. В плотности площади хранения информации чип превзошел ЗУ на магнитных сердечниках в 32800 раз, а в показатели объемной плотности - в 81 000 раз.

Плотность потока информации

Поток информации рассматривается как совокупность двух понятий — **схемы и элементов потока**.
Элементами потока могут быть документы, элементы документов (показатели, реквизиты), операторы (люди, устройства, подразделения). Операторы могут быть источниками и потребителями.
В потоке информации (ПИ) определяются два основных параметра — **направление и плотность потока**.
Направление потока задается местом его входа (наименование или шифр) подразделения. **Плотность (значение) потока** — объем информации (бит, количество документов, строк, знаков и т. д.), передаваемый в единицу времени (длительность передачи, приема или обработки).

Мощность потока информации

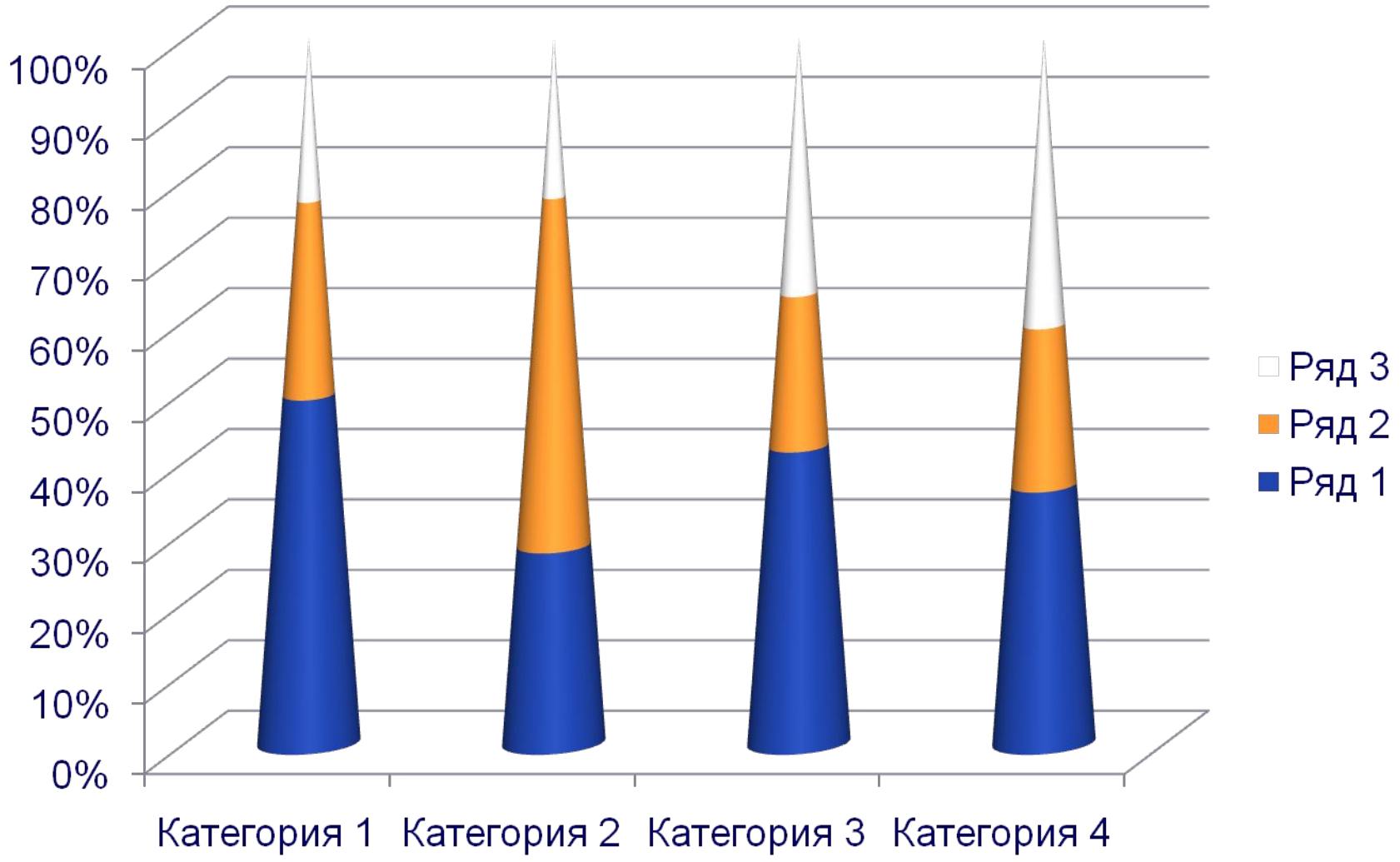
Таким образом, МОЩНОСТЬ можно трактовать как пропускную способность потока, если рассматривать её как внутреннюю характеристику потока, или как проникающую («пробивную») способность потока, если рассматривать мощность как внешнюю характеристику. Чем больше мощность потока – тем легче прохождение, и наоборот. Если опять обратиться к электрической аналогии, то нулевое сопротивление продвижению оказывает поток с бесконечной мощностью, т.е. поток с эквивалентным «бесконечным» поперечным сечением. Огромное сопротивление продвижению даёт поток с минимальной мощностью, т.е. поток с бесконечно малым эквивалентным поперечным сечением.

Мощность – это движущая, проникающая способность потока.

Проблема семантического сжатия информации

- Целевая функция
- Сегмент 1
- Сегмент 1n

Информационные конусы



Методологический аппарат науки как информационная технология

Формирование информационной технологии как самостоятельного научного направления может оказаться весьма полезным для развития и самой науки в части дальнейшего совершенствования ее методологического аппарата.

Каковы основные этапы эволюции информационных технологий?

Эволюция информационных технологий

1. 1951 год – программируемый микропроцессор;
2. 1969 год Intel Corporation – универсальный многоцелевой процессор;
3. 1973 год – Bob Metcalfe создал Ethernet.

Эволюция информационных технологий

4

- 1975 год – первый серийный персональный компьютер;

5

- 1976 год Wang Laboratories – текстовый редактор Word Processing;

6

- 1978 год – VisiCalc первая программа для работы с таблицами.

Эволюция информационных технологий

7

- **1979 год – первый текстовый редактор для ПК WordStar;**

8

- **1979 год – Oracle;**

9

- **1982 год – TCP/IP.**

Эволюция информационных технологий

10

- 1984 год – Macintosh и первый настольный лазерный принтер;

11

- 1989 год – Электронная почта;

12

- 1990 год – Tim Berners-Lee (World Wide Web).

Каков итог эволюции информационных технологий?

Эволюция информационных технологий

Итог: Информационные технологии стали основой экономики развитых стран

Растет ли стратегическая значимость информационных технологий в последние 2 десятилетия?

Переворот в сознаний (IT – commodity input)

Ежегодные затраты компаний во всем мире на аппаратные средства, ПО, коммуникации и обслуживание – более 2 трлн. долл (более 50%).

Ошибочное мнение – **по мере развития и роста доступности ИТ их стратегическая значимость также возрастает!?**

На самом деле – **вы имеете преимущество перед конкурентами только в том случае, если обладаете чем-то, чего у них нет, или делаете что-то, чего они сделать не могут!!!**

Стратегический подход

Дифференциация – главная цель и показатель эффективности любого стратегического планирования.

Функциональная и структурная дифференциации могут дать возможность компаниям в сфере IT уйти от губительных последствий ценовой конкуренции.

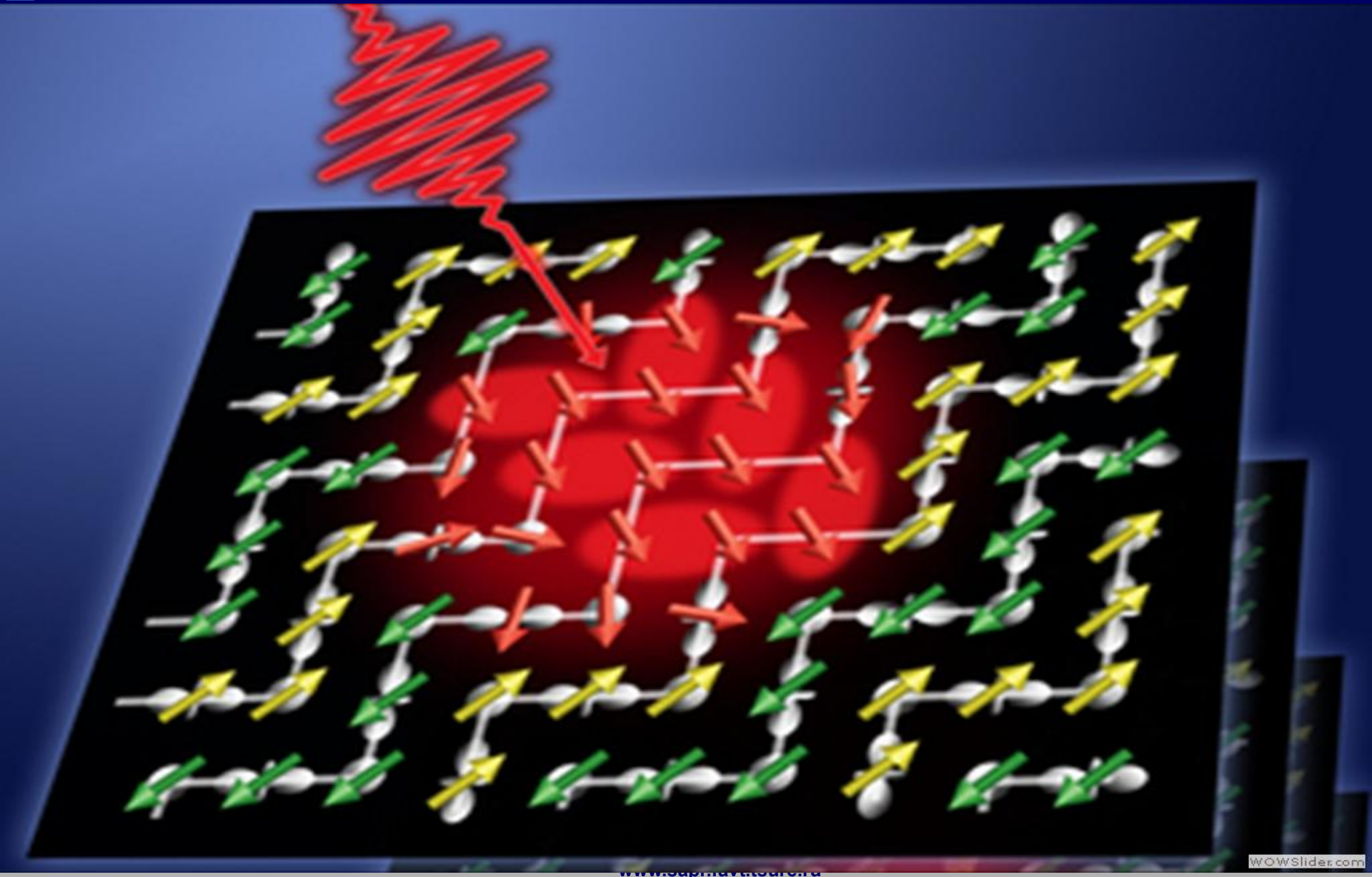
Принципиально важно отличать товарные общедоступные ресурсы (commodity input) от ресурсов, способных создать преимущества

Взгляд в будущее и прошлое

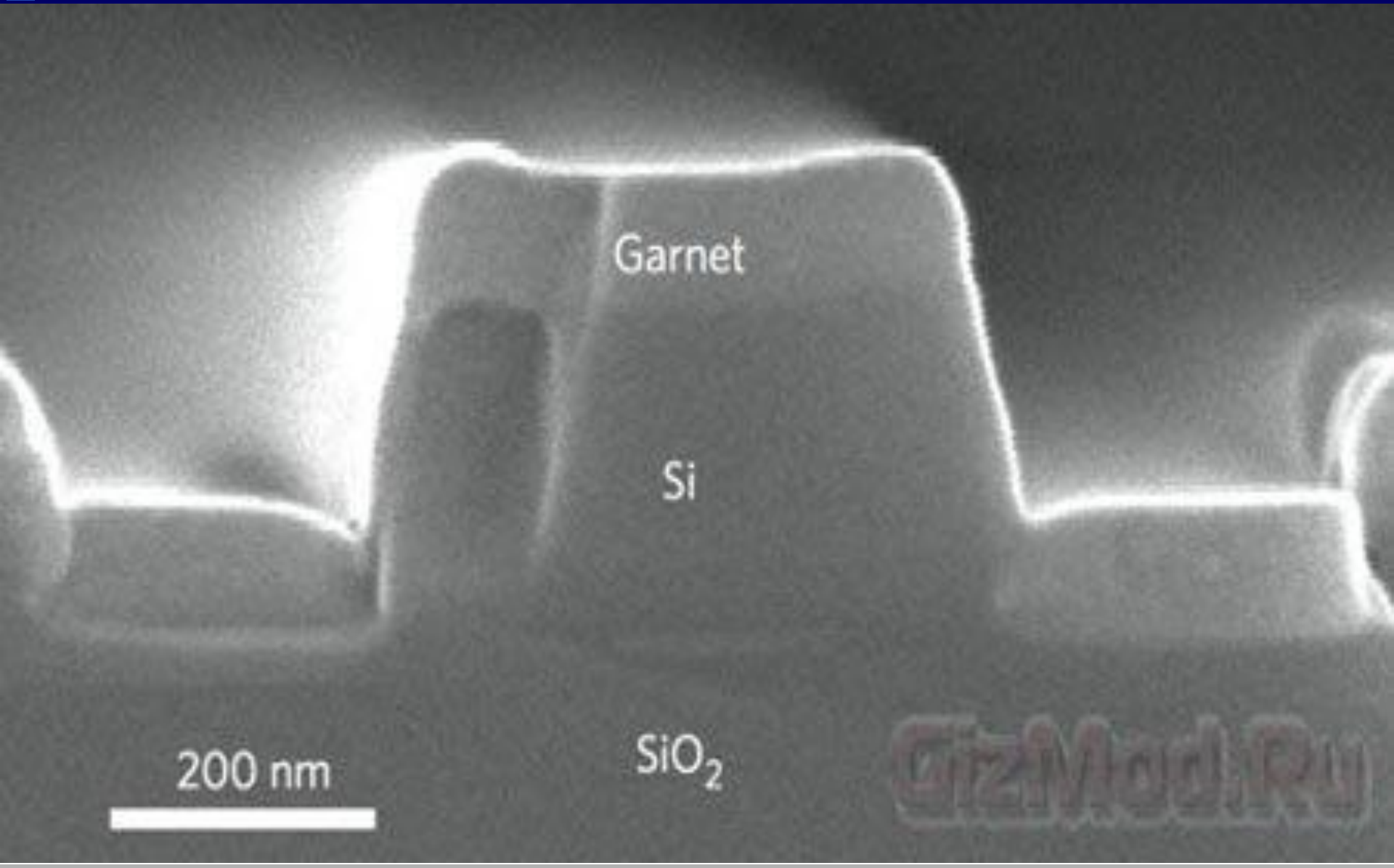
Превращение IT из источника конкурентных преимуществ в рядовую статью затрат ставит перед руководителями компаний ряд проблем.
Снижение рисков становится важнее инноваций, а сокращение затрат – важнее новых инвестиций!
IT проще понять, если рассматривать их в качестве «последнего звена» в ряду технологий, получивших широкое распространение и изменивших экономику за последние два столетия, которые по мере распространения превращались в обычный товарный ресурс.

Развитие каких технологий скорее всего приведет к «революции» в области IT?

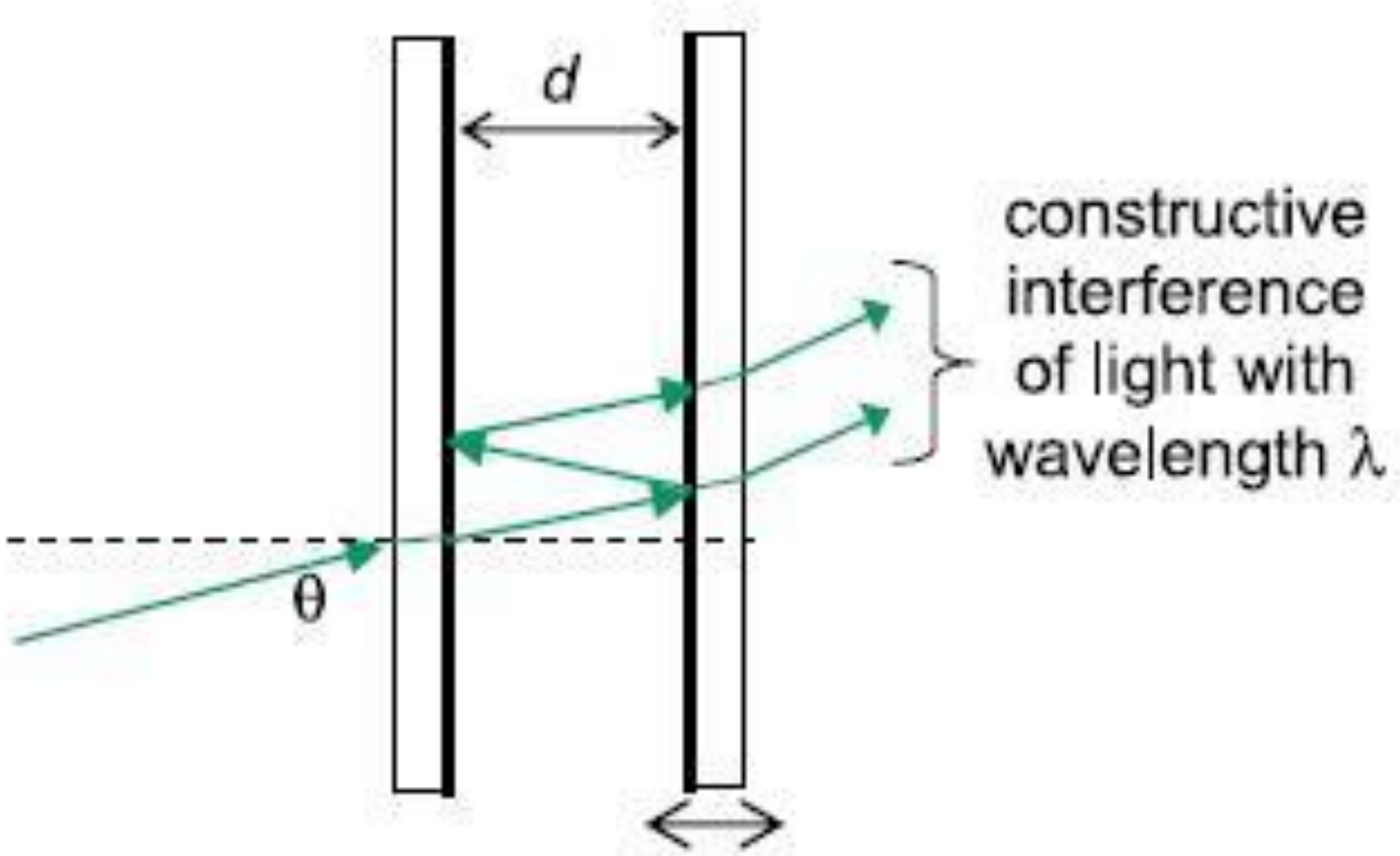
Квантовые компьютеры



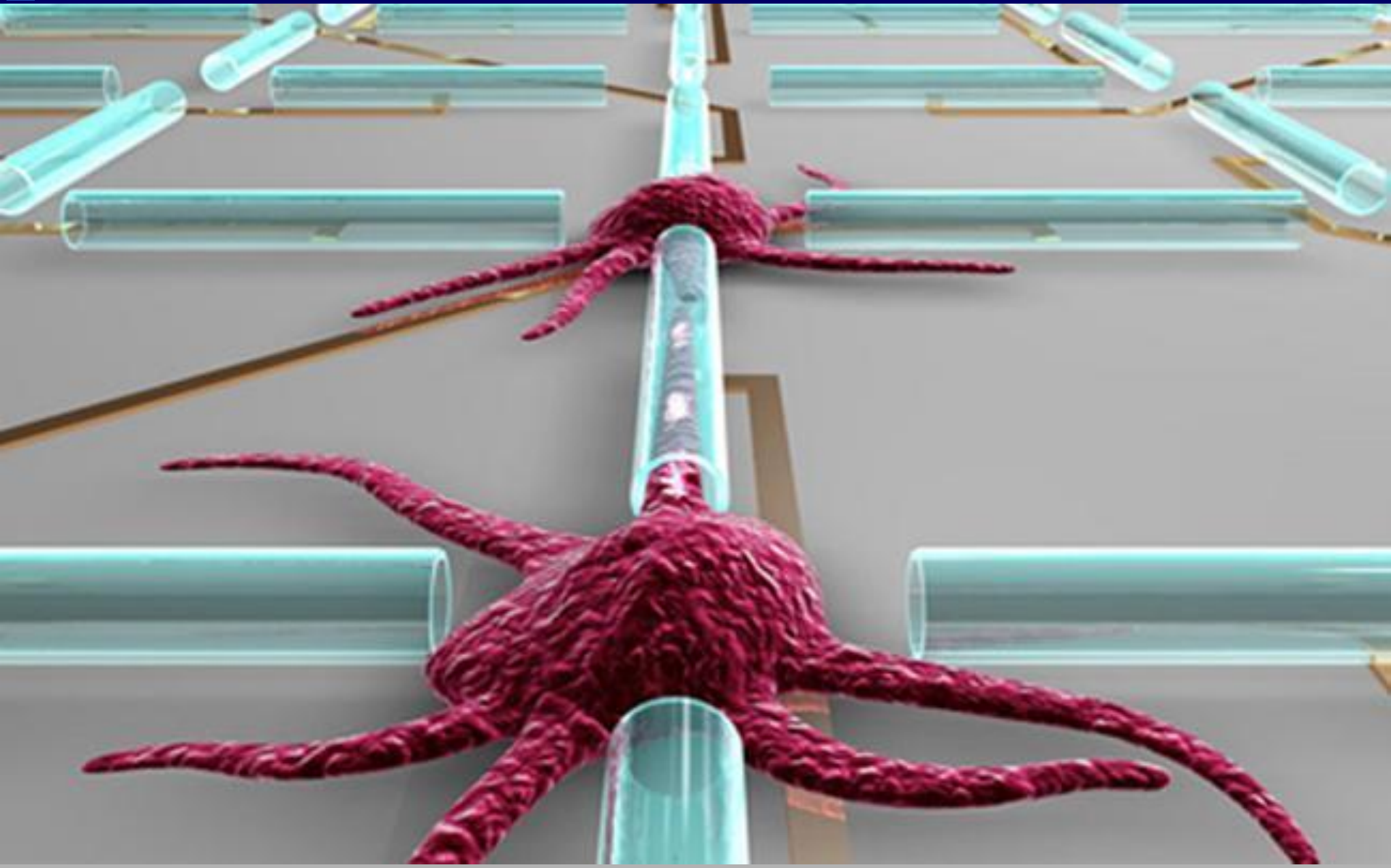
«Диод для света»



Нелинейные резонаторы



WetWare



Анализ современного состояния информационных систем (ИС)

Тенденции развития ИС

1

Появление у систем так называемых интеллектуальных (семантических) свойств

2

Способность систем к самоорганизации, саморазвитию (аналогия с организмами живой природы)

Какой основной недостаток технологии поддержки «всемирной паутины» WWW?

Развитие поисковых систем

В современном состоянии Всемирной паутины (WWW) веб-страницы являются пассивными, а «мыслящей частью» являются именной поисковые средства. Они не только индексируют страницы Сети, но и всячески анализируют их, организуя собственные базы знаний. **Контроль поисковых систем приведет в будущем к коммерциализации сервисов обработки информации.**

Развитие поисковых систем

Правила вывода, задаваемые в онтологиях, предоставляют дополнительные возможности. Поисковая система не «понимает» в полном смысле этого слова ничего из содержащийся в них информации, но теперь уже может манипулировать терминами гораздо более эффективно.

Компьютер способен разобраться со структурой документа, но не может обрабатывать его семантику.

Word Wide Web Consortium (W3C) стандарты

Какие стандарты устанавливают механизм структурирования информации в сети Интернет таким образом, что её восприятие становится доступным программам, способным выполнять логический анализ и генерировать необходимые выводы?

Word Wide Web Consortium (W3C) стандарты

Extensible Markup Language (XML)
Язык построения структурированных документов

Resource Description Framework (RDF)
Система описания ресурсов

Ontology Web Language (OWL)
Язык описания онтологий

Основные стандарты организации интеллектуальной сети (структурирования) информации

Развитие поисковых систем

Группы разработчиков (IBM, HP,W3C) провели формирование онтологий для широкого ряда предметных областей, которые являются одним из ключевых элементов технологии и позволяют описывать отношения объектов реального мира с помощью языков OWL и DAML (DARPA Agent Markup Language, DARPA – Defence Advanced Research Projects Agency)

Что в ближайшее время станет развитием технологии поддержки «всемирной паутины» www?

Сеть GGG (IntellectNet)

Информационная эпоха завершится созданием и всемирным использованием **децентрализованной** глобальной информационной сети GRAPH – GLOBAL GNOSEOLOGY GRAPH (GGG).

Сеть GGG логически дополняет имеющиеся глобальные информационные сети и продолжает взаимосвязанную цепочку NET-WEB-GRAPH. Где NET – осуществляет глобальную физическую связь различных компьютеризированных устройств, в ней нам всем мобильно и оперативно. WEB – формирует глобальную логическую сеть участников взаимодействия пакетами данных, в ней нам свободно и доступно. GRAPH – будет обеспечивать коллективное создание глобального сознания общества и гармонизацию реализации общей деятельности цивилизации.

Сеть GGG (IntellectNet)

NET-WEB-GRAPH

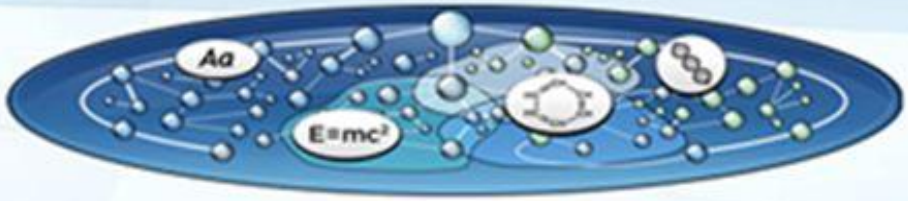
GGG (G3) – Глобальный Гносеологический Граф (Global Gnoseology Graph).

Глобальная информационная сеть нового поколения

Сетецентрические принципы создания

3. GRAPH – GGG (G3)

СЕТЬ ЗНАНИЙ
СРЕДА ВЗАИМОДЕЙСТВИЯ



2. WEB - WWW

СЕТЬ ДОКУМЕНТОВ
СРЕДА ВЗАИМОСВЯЗИ

1. NET

СЕТЬ КОМПЬЮТЕРОВ
СРЕДА КОММУТАЦИИ

Свойства сети GGG

Виртуальность – удобство описания реальностей мира информацией о нем

Доступность – фактическое отсутствие барьеров

Универсальность – унификация языка, знаний, культуры

Автоматическая генерация программного продукта

Способность отражать всю сложность реальных явлений

Индивидуальность

Каким образом решать трансвычислительные задачи?

Трансвычислительная задача

(*Transcomputational problem*) — в **теории сложности вычислений** задача, для решения которой требуется обработка более чем $\mathbf{10^{93}}$ бит информации. Число 10^{93}, называемое «**пределом Бремерманна**», согласно Гансу-Иоахиму Бремерманну, представляет собой общее число бит, обрабатываемых гипотетическим компьютером размером с Землю за период времени, равный общему времени существования Земли.

Задача коммивояжёра

Задача коммивояжёра заключается в поиске пути обхода заданного списка городов, имеющего минимальную стоимость. Путь обхода должен посещать все города ровно по одному разу и возвращаться в исходный город. Если в списке ***n*** городов, то число возможных путей обхода равно ***n!***. Поскольку **66!** примерно равно $\mathbf{5{,}443449391 \times 10^{92}}$, а $\mathbf{67! \approx 3{,}647111092 \times 10^{94}}$, задача проверки всех возможных путей становится трансвычислительной для $\boldsymbol{n > 66}$.

Тестирование интегральных схем

Полное тестирование всех комбинаций интегральной схемы с **308** входами и **1** выходом требует проверки $\mathbf{2^{308}}$ комбинаций входных данных. Поскольку число $\mathbf{2^{308}}$ является трансвычислительным, задача тестирования такой системы интегральных схем является трансвычислительной проблемой. Это означает, что отсутствует способ проверки схемы для всех входных данных методом грубой силы.

Проблема анализа систем

Система из n переменных, каждая из которых может принимать k возможных состояний, может иметь k^n возможных состояний. Анализ такой системы требует обработки как минимум k^n бит информации. Задача становится трансвычислительной, если $k^n > 10^{93}$. Это происходит при следующих значениях k и n:

$k = 2; n = 308$

$k = 10; n = 93$

Распознавание узоров

Рассмотрим массив размером $q \times q$, представляющий узор, похожий на шахматную доску, в которой каждый квадрат может быть одного из k цветов. Общее число возможных узоров равно k^n, где $n = q^2$. Задача определения наилучшей классификации узоров по какому-либо выбранному критерию может быть решена перебором всех возможных цветовых узоров. Для **2** цветов такой поиск становится трансвычислительным при размере массива **18×18** и более. Для массива **10×10** задача становится трансвычислительной при числе цветов **9** и более.

*Данная задача имеет отношение к изучению физиологии сетчатки. Сетчатка состоит примерно из миллиона светочувствительных клеток. Даже если у клетки имеется всего **2** возможных состояния, обработка состояния сетчатки в целом требует обработки более чем $10^{300\,000}$ бит информации. Это намного превосходит предел Бремерманна.*

Применение эволюционного моделирования для создания ИС

В основе второй тенденции создания глобальной сетевой структуры (саморазвивающейся искусственной интеллектуальной системы) лежит использование концепции эволюционного моделирования (ЭМ).

ЭМ идеально приспособлено для решения задач, требующих оптимизации множества гетерогенных критериев. Тогда как методы математического анализа зачастую не способны найти компромисс между несколькими конкурирующими факторами.

Применение эволюционного моделирования для создания ИС

Два преимущества ЭМ

1

Отсутствие жестко заданной логики принятия решений

2

Широкий спектр анализируемых решений (в отличие от классических подходов)

Применение эволюционного моделирования для создания ИС

Данные преимущества открывают перед исследователем новые перспективы и дают возможность использовать адаптивные самонастраивающиеся алгоритмы, позволяющие «выращивать» решения NP- сложных задач.

Фундаментальная научная проблема

Технологии создания информационных систем применяют дискретный способ проектирования, при этом эволюция систем имеет непрерывный характер. **Данное противоречие является фундаментальной научной проблемой**. Поэтому понятен интерес к исследованиям и разработке новых подходов к созданию ИС, свободных от имеющихся ограничений и условностей. Парадигма, использующая принципы эволюционного моделирования, является весьма перспективной.

Концепция эволюционного развития ИС

- Эволюция информационных систем позволит им иметь структуру подобную живым системам – части организмов, организмы и сообщества организмов (элементы систем, системы и сети), причем, каждый из перечисленных элементов также будет представлять интегрированное целое.

Самоорганизация ИС

- **Самоорганизацию информационных систем будем рассматривать с точки зрения их *саморегулирования*, которое проявляется в сбалансированном функционировании системы за счет отношений между внутренними и внешними элементами. *Согласно классическому определению, процесс самоорганизации приводит к значительному росту сложности внутренней организации системы.***
- ***Самоорганизация не является очевидным свойством, проявляющимся при любых обстоятельствах, даже для интеллектуальных информационных систем. Для неизбежности возникновения процессов самоорганизации необходимо выполнение определенных внешних и внутренних условий, связанных не с увеличением количества информации, а с проявлением ее инструктирующих свойств, повышающих ценность данной информации, непосредственно связанную с ее используемостью.***

Самоорганизация ИС

- *Таким образом, одним из условий возникновения самоорганизации в информационной системе является наличие механизмов реализации поиска и отбора информации, имеющей определенную меру ценности, что является упрощенной моделью творческой деятельности человека.*

- ***Нетривиальность** данного подхода к возникновению самоорганизации заключается в том, что ценная информация накапливается в системе медленно и постепенно, давая на протяжении долгого времени лишь несущественные преимущества. Значительные изменения в поведении системы возможны только при условии постоянного анализа и обработки имеющейся ценной информации с целью определения ранее неизвестных зависимостей и закономерностей, выделяющих среди множества элементов информации упорядоченные последовательности составных частей знания нового качества, использование которого позволит перейти системе на новый уровень собственной эволюции.*

Проблема

- ***Управляемое создание условий для протекания такого процесса самоорганизации усложняется неопределенностью надлежащего момента времени, в котором интеграция определенных разрозненных элементов ценной информации приведет к формированию знания нового качества. Дальнейшее протекание процесса накопления ценной информации может привести к рассеиванию преимуществ интеграции.***

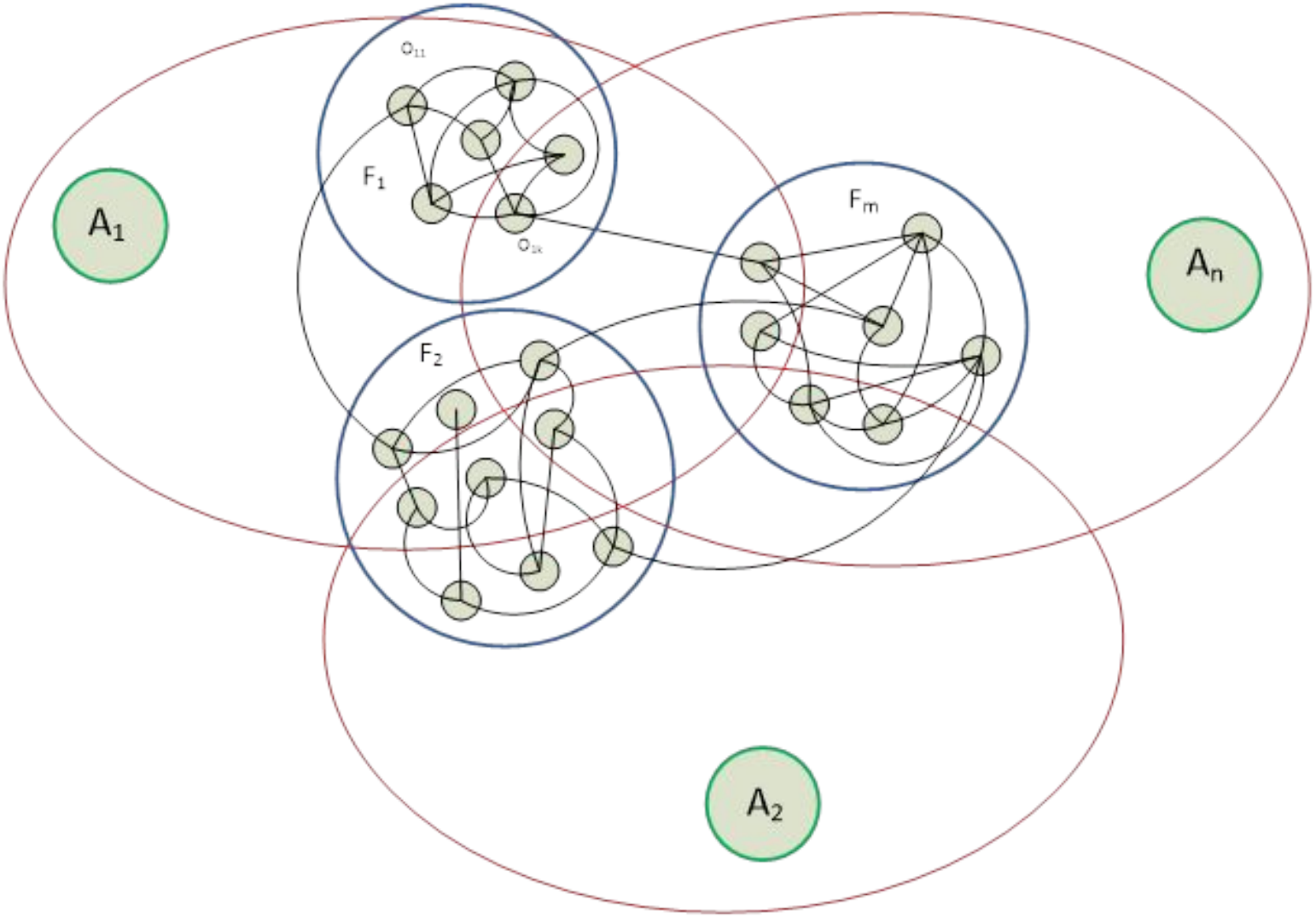
Задача №1

- Разработка принципов получения новых знаний и определения **степени их ценности** на основе моделей процессов самоорганизации и **биоинспирированного поиска**

Среда для решения задач УЗ

- **Реальная среда для решения задачи управления знаниями представляет собой сложный для формализации комплекс взаимосвязанных предметных областей, в котором возникают и актуализируются неструктурированные задачи выявления и идентификации элементов знания. Такая реальная среда характеризуется свойствами частичной наблюдаемости, динамики, непрерывности, эпизодичности, стохастичности. Построение модели подобной среды, пригодной для интерпретации и эффективной обработки неформализованных запросов должно быть основано на концепции агентности, предполагающей множественность, автономность одновременно со связанностью субъектов принятия решений по отношению к этой среде и субъектам (пользователям) формирования запросов.**

МОДЕЛЬ ИНТЕГРИРОВАННОЙ ИНФОРМАЦИОННОЙ СРЕДЫ



Проблема

- **Частичная наблюдаемость среды решения задач управления знаниями обусловлена восприятием автономным агентом лишь некоторых пространственных пределов, обусловленных заданными глубиной поиска и мощностью окрестности, исследуемых междисциплинарных связей. Чаще всего, это незначительная часть среды, характеризующаяся существенной неполнотой информации об отношениях на множестве элементов знания.**

Задача №2

- Разработка модели среды для решения задачи управления знаниями, обладающей свойствами частичной наблюдаемости, динамики, непрерывности, эпизодичности, стохастичности, на основе многоагентного, онтологического и эволюционного моделирования.

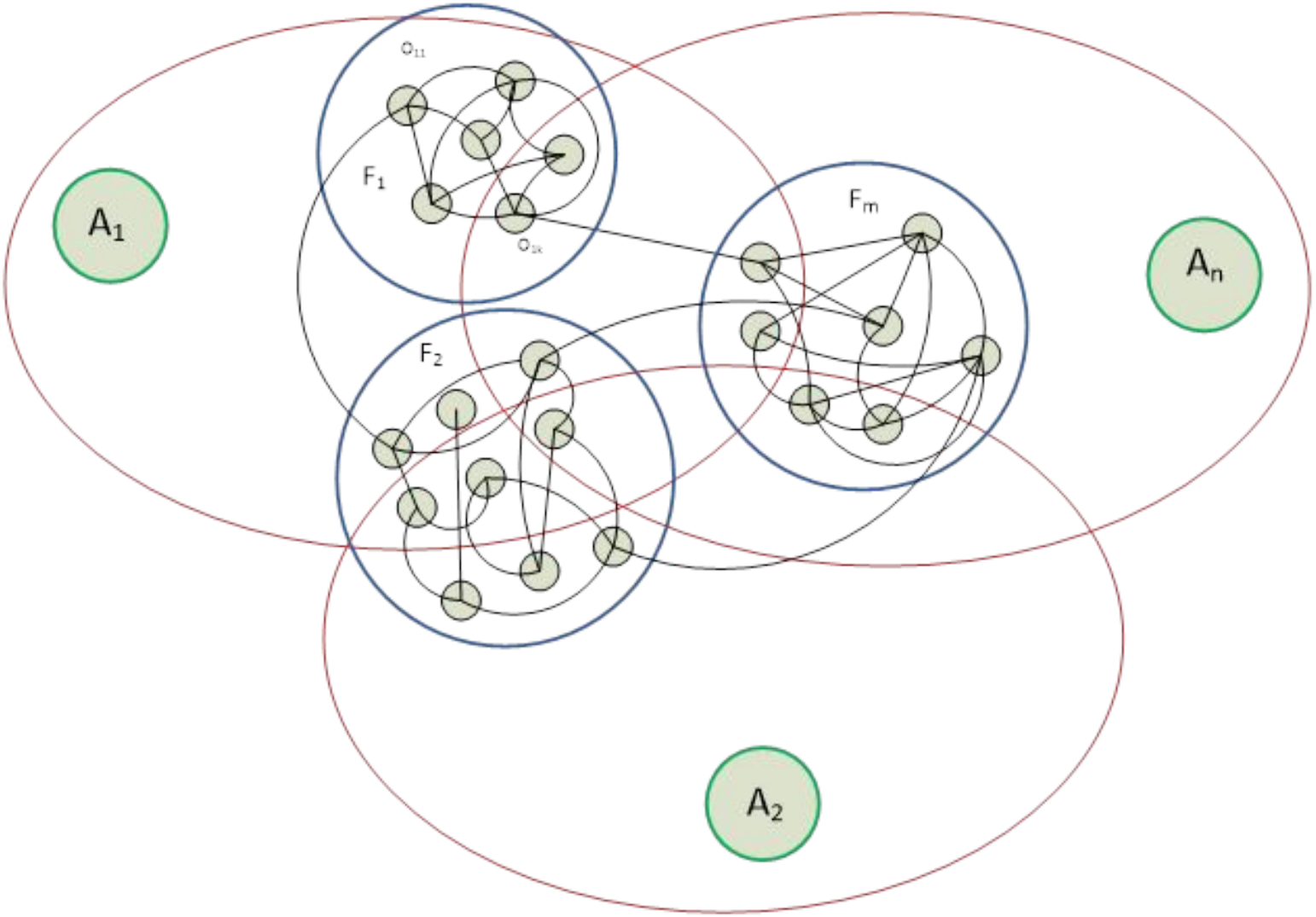
Неструктурированные задачи УЗ

- В методологии системного анализа принято утверждение о том, что для автоматизированного решения неструктурированных задач необходимо сначала свести их к структурированным задачам и что эта процедура осуществляется человеком – экспертом. В этой связи, особый интерес для данного исследования, которое ставит своей целью разработку универсальных моделей интеллектуального накопления и обработки знаний с формализацией семантики поиска и принятия решений при обработке знаний, представляют именно способы и алгоритмы, которые применяются экспертами для сведения неструктурированных задач к структурированным.

Неструктурированные задачи УЗ

- Будем утверждать, что решение проблемы системного анализа неструктурированных задач накопления и обработки знаний из распределенных неоднородных источников требует формализации процессов структуризации знаний, которая обычно выполняется экспертами. Эта задача является **классической задачей искусственного интеллекта, сущность которой состоит в анализе междисциплинарных связей различных предметных областей на основе исследования модели семантических отношений между элементами знания**.

МОДЕЛЬ ИНТЕГРИРОВАННОЙ ИНФОРМАЦИОННОЙ СРЕДЫ



Проблема

- Очевидно, что основную роль в способности эксперта структурировать задачи играет его собственная системная организация и детерминированное ею целенаправленное поведение. Отсюда следует, что для формализации процессов структуризации необходимо воспроизвести эту системную организацию и реализовать соответствующее поведение автономной информационной системы управления знаниями, определяющее сходные функциональные характеристики.

Задача №3

- Решение фундаментальной проблемы разработки алгоритмов моделирования рассуждений, позволяющих **сводить неструктурированные задачи поиска и обработки проблемно-ориентированных знаний к структурированным**, на основе биоинспирированного поиска и онтологии интегрированного пространства знаний.

Оценка семантической близости

- Формальная модель семантической близости позволяет получить значения расстояний и признаков, извлекаемых из предметных областей элементов знания, отношения между которыми, в свою очередь, составят содержание соответствующих абстрактных моделей. **Такие модели**, выраженные в терминах некоторого языка, **представляют собой уже структурированные задачи**, решение которых, в целом, не представляет особых проблем.

Проблема

- Исследование и анализ способов оценки семантической близости на основании иерархических и неиерархических связей.
- Разработка графовых моделей компонентов триплетов онтологий.
- Методы расчета семантической близости.
- Разработка генетического, эволюционного, биоинспирированного алгоритмов, реализующих графовые модели.
- Проведение экспериментальных исследований.

Задача №4

- Решение задачи разработки методов поддержки принятия решений в условиях информационной неопределенности для оптимизации процессов управления знаниями на основе биоинспирированных алгоритмов оценки семантической близости концептов.

Реакция системы на воздействие

- Рассмотрим модель реакции системы на воздействие в контексте процессов циклического сценария управления знаниями. Классическая задача прямой идентификации заключается в определении функциональной зависимости выходного сигнала *y* от входного – *x* (*y* = *F(x)*). Для создания интегрированных моделей априорных знаний методом компараторной идентификации используют алгебру предикатов и предикатных операций. Системы предикатных уравнений решаются с помощью универсальных программ, написанных на языке высокого уровня. Любое уравнение алгебры предикатов может быть представлено в виде переключательной цепи как основы интеллектуального процессора обработки знаний.

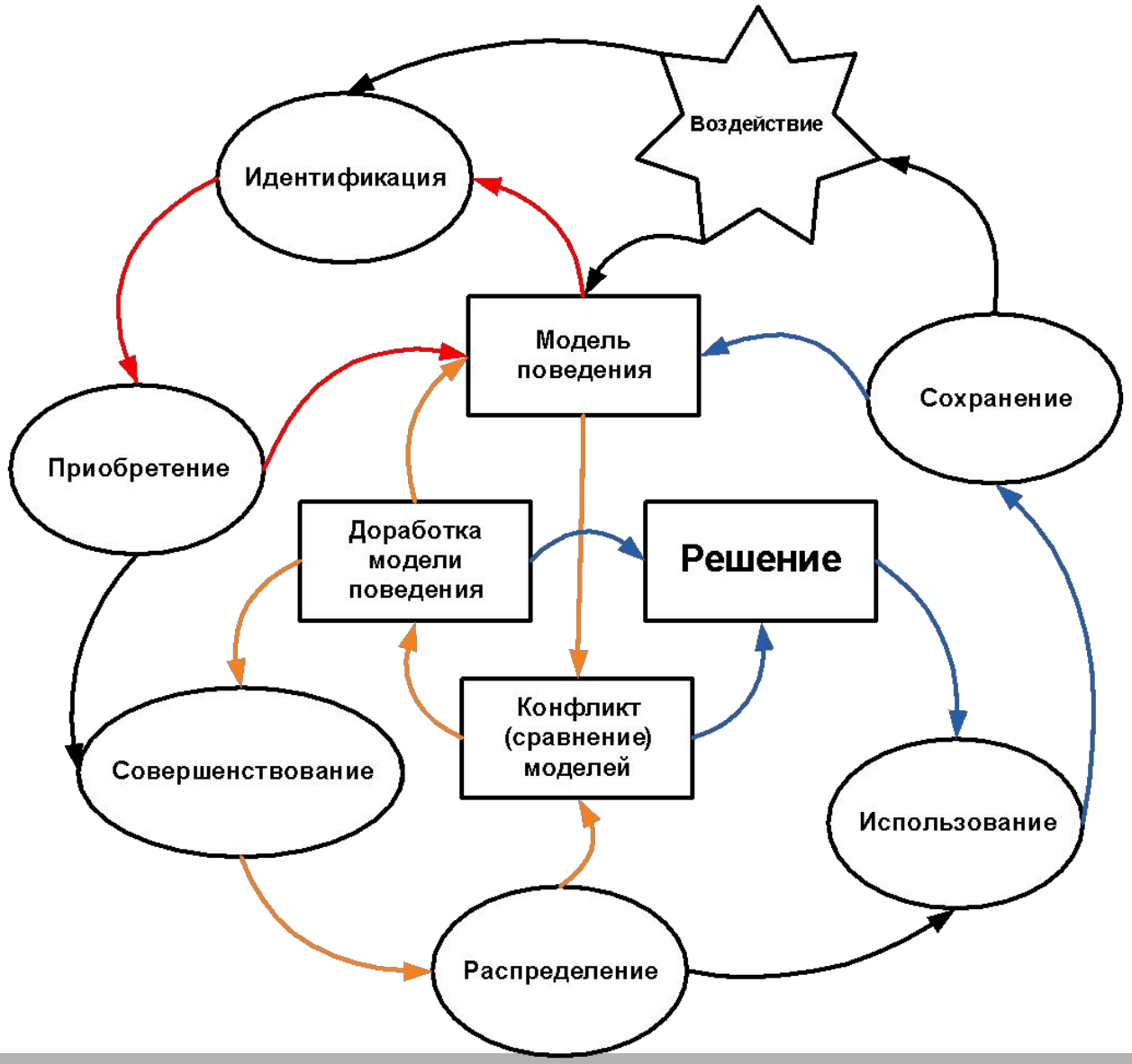
Проблема

- Обратим внимание на то, что **модель реакции системы на внешнее воздействие будет влиять на все, без исключения, стадии циклического сценария управления знаниями**. Учет модели реакции системы на воздействие позволяет увеличить разнообразие возможных циклов управления знаниями в интегрированной модели.

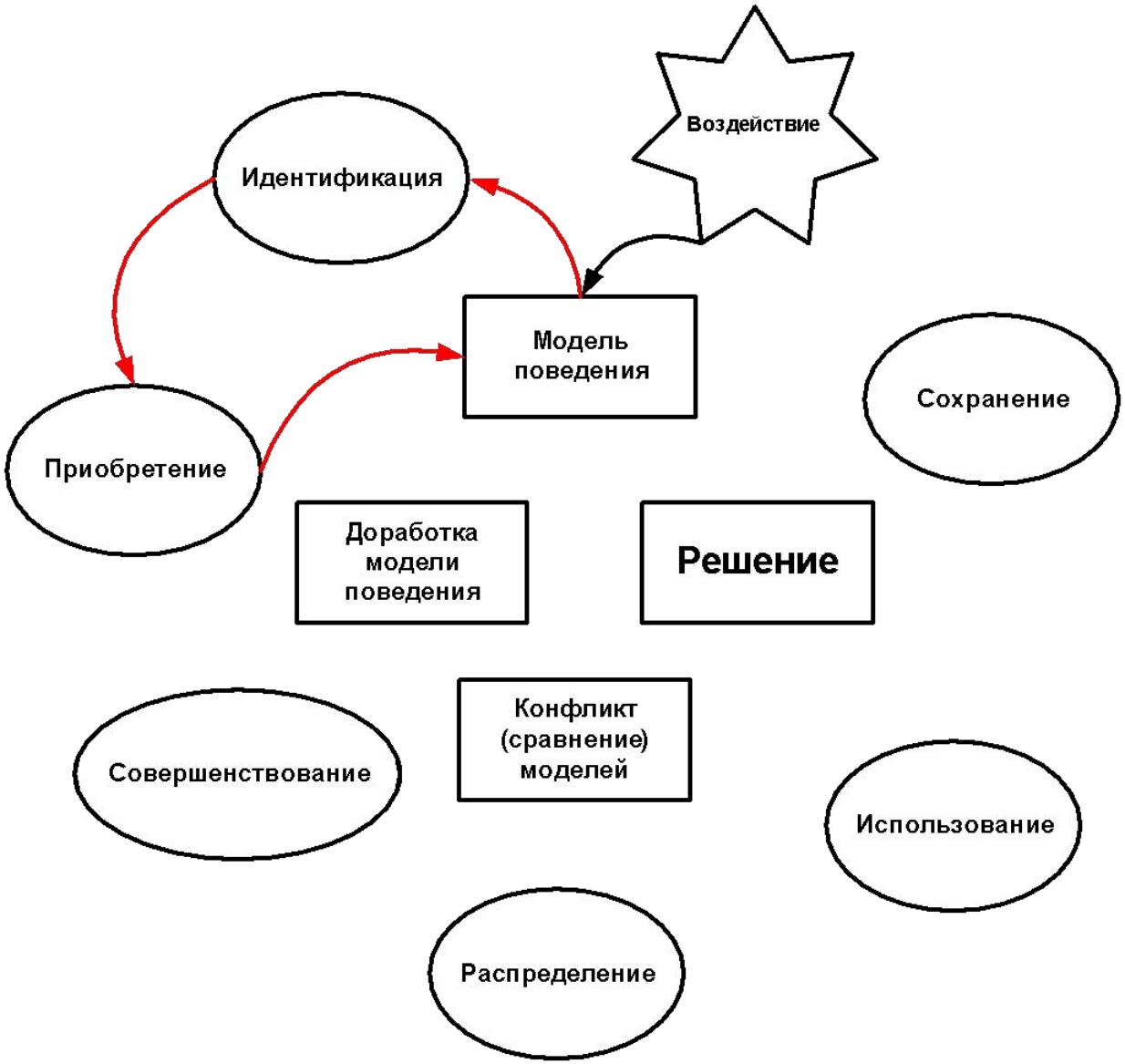
Задача №5

- Разработка **эволюционных алгоритмов поведения** интеллектуальной информационной системы управления знаниями на основе интегрированных моделей реакции на воздействие.

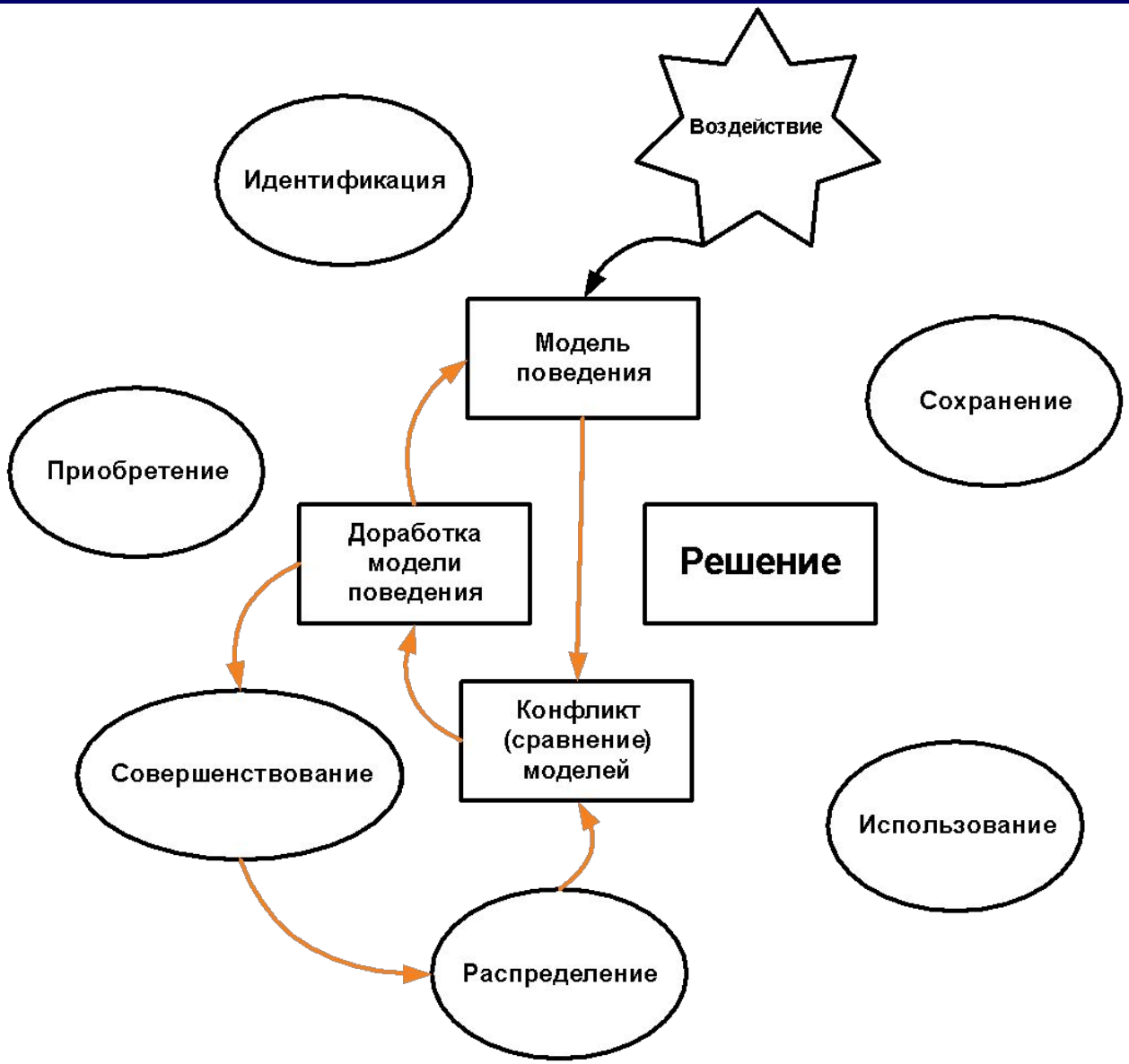
Реакция системы на воздействие



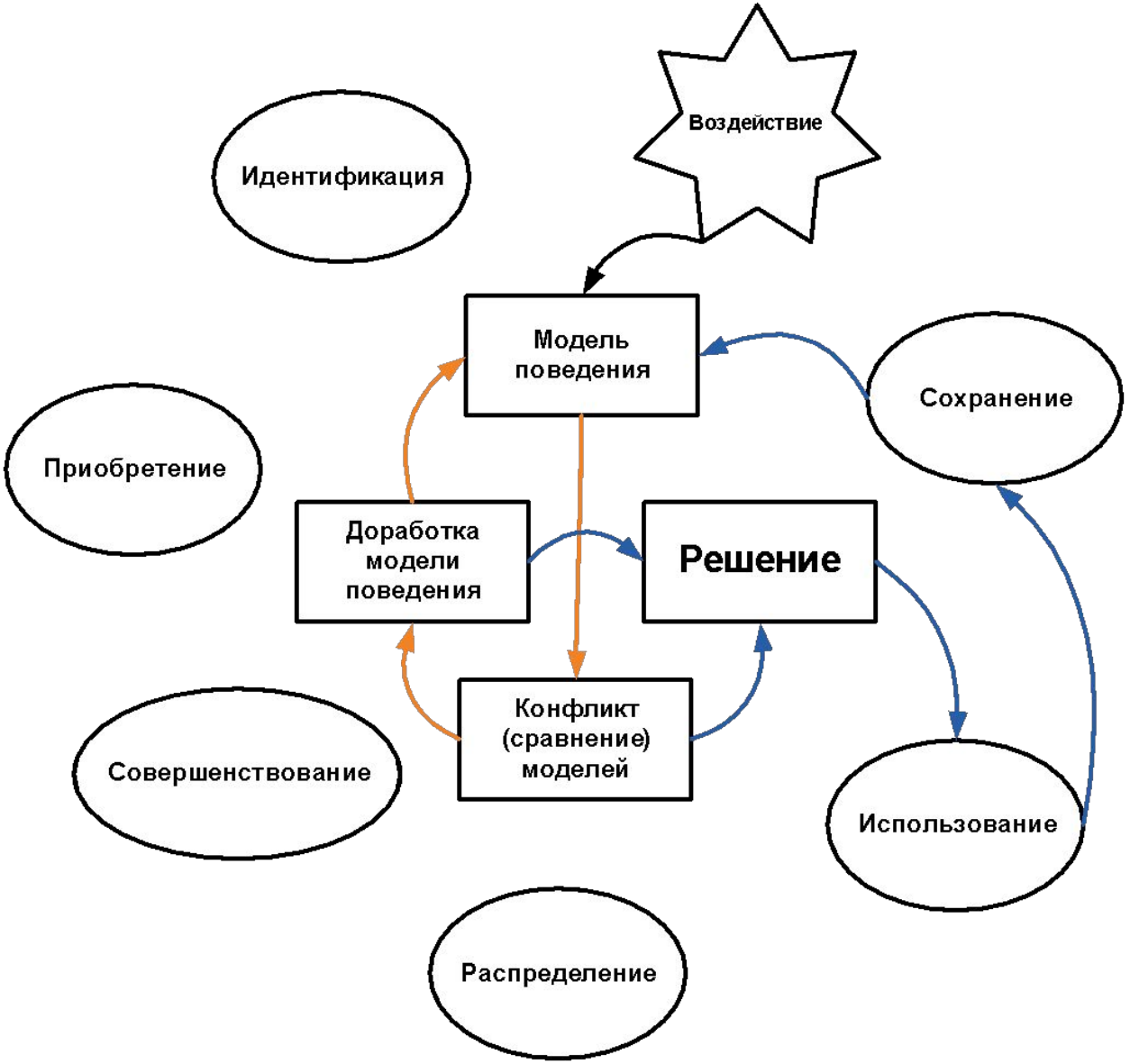
Упрощенный цикл УЗ



Доработка модели поведения



Принятие решений



Поисковые запросы

- Общность семантических моделей предметных областей позволяет создавать нерегламентированные запросы поиска элементов знания на высоком уровне абстракции, что само по себе является препятствием объективности понимания поставленной задачи. Подобные запросы характеризуются значительной степенью обобщения одновременно с пренебрежением к отдельным деталям рассматриваемых концептов. Такая задача может быть решена успешно в случае сходства моделей семантики автора запроса и информационной системы управления знаниями, тогда будет обеспечено необходимое понимание и восстановление упущенных в запросе деталей.

Проблема

- В традиционных механизмах поиска, документ, как правило, извлекается, когда хотя бы одно из ключевых слов в строке запроса находится внутри концепта. Недостижимым в этом случае является получение всех экземпляров концептов, которые связаны с введенным ключевым словом, даже если оно не встречается внутри самого концепта.

Задача №6

- Создание информационно-поисковой среды построенной на новых принципах редактирования и разбиения поисковых запросов с целью **эффективной навигации при поиске и обработке проблемно-ориентированных знаний из гетерогенных источников.**

2. Проблемы и тенденции развития программного обеспечения и вычислительной техники

Уровни языков программирования

Уровни языков программирования

Если язык программирования ориентирован на конкретный тип процессора и учитывает его особенности, то он называется **языком программирования низкого уровня**. Имеется в виду, что операторы языка близки к машинному коду и ориентированы на конкретные команды процессора.

Уровни языков программирования

Язык ассемблера, который представляет каждую команду машинного кода, но не в виде чисел, а с помощью символьных условных обозначений, называемых **мнемониками**. Однозначное преобразование одной машинной инструкции в одну команду ассемблера называется **транслитерацией**.

Уровни языков программирования

Языки программирования **высокого** уровня понятнее человеку. Особенности конкретных компьютерных архитектур в них не учитываются, поэтому создаваемые программы на уровне исходных текстов легко переносимы на другие платформы, для которых создан транслятор этого языка. Разрабатывать программы на языках высокого уровня с помощью понятных и мощных команд значительно проще, а ошибок при создании программ допускается гораздо меньше.

Классификация языков

- **Компилируемый язык –**
- **язык программирования, исходный код которого преобразуется компилятором в машинный код и записывается в файл с особым заголовком и/или расширением для последующей идентификации этого файла, как исполняемого операционной системой (в отличие от интерпретируемых языков программирования, чьи программы выполняются программой-интерпретатором).**

Классификация языков

- **Аспектно-ориентированные языки –**
- **парадигма программирования, основанная на идее разделения функциональности для улучшения разбиения программы на модули. Методология АОП была предложена группой инженеров исследовательского центра Xerox PARC под руководством Грегора Кичалеса (Gregor Kiczales). Ими же было разработано аспектно-ориентированное расширение для языка Java, получившее название AspectJ — (2001 год).**

Классификация языков

- **Структурные языки –**
- **методология разработки программного обеспечения, в основе которой лежит представление программы в виде иерархической структуры блоков. Предложена в 1970-х годах Э. Дейкстрой и др.В соответствии с данной методологией любая программа строится из трёх базовых управляющих структур: последовательность, ветвление, цикл; кроме того, используются подпрограммы. При этом разработка программы ведётся пошагово, методом «сверху вниз». Basic, Pascal, Fortran, Algol…**

Классификация языков

- **Процедурные языки –**
- **программирование на императивном языке, при котором последовательно выполняемые операторы можно собрать в подпрограммы, то есть более крупные целостные единицы кода, с помощью механизмов самого языка. Процедурное программирование является отражением архитектуры традиционных ЭВМ, которая была предложена Фон Нейманом в 1940-х годах. Теоретической моделью процедурного программирования служит абстрактная вычислительная система под названием машина Тьюринга. Си, Си++, Кобол, Ada, Pascal, Fortran, Go…**

Классификация языков

- **Логические языки –**
- парадигма программирования, основанная на автоматическом доказательстве теорем, а также раздел дискретной математики, изучающий принципы логического вывода информации на основе заданных фактов и правил вывода. Логическое программирование основано на теории и аппарате математической логики с использованием математических принципов резолюций.
- Самым известным языком логического программирования является Prolog.

Классификация языков

- **Объектно-ориентированные языки –**
- **методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определенного класса, а классы образуют иерархию наследования. Идеологически ООП — подход к программированию как к моделированию информационных объектов, решающий на новом уровне основную задачу структурного программирования: структурирование информации с точки зрения управляемости, что существенно улучшает управляемость самим процессом моделирования, что в свою очередь особенно важно при реализации крупных проектов.**
- **C++, Object Pascal, Java, Python, Ruby, Objective-C, Visual Basic, Delphi, C#, PHP…**

Классификация языков

- **Функциональные языки –**
- раздел дискретной математики и парадигма программирования, в которой процесс вычисления трактуется как вычисление значений функций в математическом понимании последних (в отличие от функций как подпрограмм в процедурном программировании). Противопоставляется парадигме императивного программирования, которая описывает процесс вычислений как последовательное изменение состояний (в значении, подобном таковому в теории автоматов). При необходимости, в функциональном программировании вся совокупность последовательных состояний вычислительного процесса представляется явным образом, например, как список.
- Lisp, Erlang, Haskell…

Классификация языков

- **Мультипарадигмальные языки –**
- возможности такого языка изначально предполагалось унаследовать от нескольких, чаще всего не родственных языков. Цель разработки мультипарадигмальных языков программирования состоит, как правило, в том, чтобы позволить программистам использовать лучший инструмент для работы, признавая, что никакая парадигма не решает все проблемы самым лёгким или самым эффективным способом.
- Один из наиболее амбициозных примеров — Oz, который является логическим языком, функциональным языком, объектно-ориентированным языком, языком конкурентного (параллельного) программирования и т. д. В качестве одного из наиболее успешных мультипарадигмальных языков программирования часто называют язык C++.

Классификация языков

- **Динамические языки –**
- **язык программирования, который позволяет определять типы данных и осуществлять синтаксический анализ и компиляцию «на лету», на этапе выполнения программы. Динамические языки удобны для быстрой разработки приложений.**
- **К динамическим языкам относятся: Perl, Tcl, Python, PHP, Ruby, Smalltalk, JavaScript. Некоторыми динамическими чертами обладает также Visual Basic.**

Классификация языков

- **Графические языки** (Visual DataFlex, FBD)
- **Для промышленной автоматизации** (IL,SCL)
- **Стековые** (Forth, PostScript)
- **Параллельные** (C#, Java, Erlang)
- **Неполнофункциональные языки** (Abap, Awk)
- **Языки СУБД** (SQL, xBase)
- **Скриптовые языки** (Perl, PHP, Python, Ruby)
- **Рефлексивные** (Objective-C, Smalltalk)
- **Эзотерические** (False, Intercal)

Fortran

Fortran (Фортран). Это первый компилируемый язык, созданный Джимом Бэкусом в 50-е годы. Программисты, разрабатывавшие программы исключительно на ассемблере, выражали серьезное сомнение в возможности появления высокопроизводительного языка высокого уровня, поэтому основным критерием при разработке компиляторов Фортрана являлась эффективность исполняемого кода.

Cobol

Cobol (Кобол). Это компилируемый язык для применения в экономической области и решения бизнес-задач, разработанный в начале 60-х годов. Он отличается большой «многословностью» — его операторы иногда выглядят как обычные английские фразы В Коболе были реализованы очень мощные средства работы с большими объемами данных, хранящимися на различных внешних носителях.

Algol

Algol (Алгол). Компилируемый язык, созданный в 1960 году. Он был призван заменить Фортран, но из-за более сложной структуры не получил широкого распространения. В1968 году была создана версия Алгол 68, по своим возможностям опережающая многие языки программирования, однако из-за отсутствия достаточно эффективных аппаратных платформ для нее не удалось своевременно создать хорошие компиляторы.

Pascal

Pascal (Паскаль). Язык Паскаль, созданный в конце 70-х годов основоположником множества идей современного программирования Никлаусом Виртом, во многом напоминает Алгол, но в нем ужесточен ряд требований к структуре программы и имеются возможности, позволяющие успешно применять его при создании крупных проектов.

Basic

Basic (Бейсик). Для этого языка имеются и компиляторы, и интерпретаторы, а по популярности он занимает одно из первых мест в мире. Он создавался в 60-х годах в качестве учебного языка и очень прост в изучении.

C

C (Си). Данный язык был создан в лаборатории Bell и первоначально не рассматривался как массовый. Он планировался для замены ассемблера, чтобы иметь возможность создавать столь же эффективные и компактные программы, и в то же время не зависеть от конкретного типа процессора.

Си во многом похож на Паскаль и имеет дополнительные средства для прямой работы с памятью (указатели).

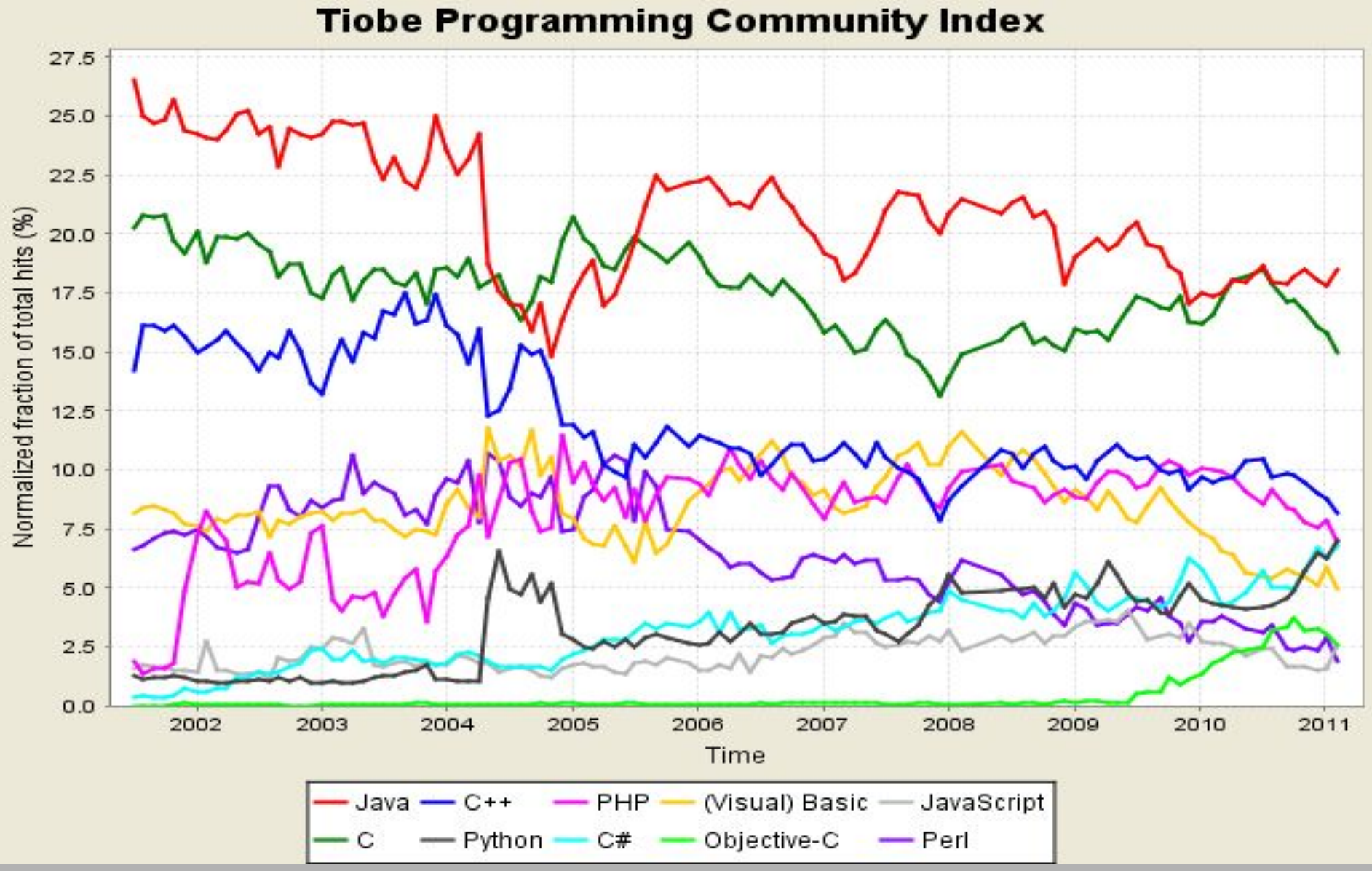
C++

C++ (Си++) — это объектно-ориентированное расширение языка Си, созданное Бьярном Страуструпом в 1979 году. Множество новых мощных возможностей, позволивших резко повысить производительность программистов, наложилось на унаследованную от языка Си определенную низкоуровневость, в результате чего создание сложных и надежных программ потребовало от разработчиков высокого уровня профессиональной подготовки.

Java

Java (Джава). Этот язык был создан компанией Sun в начале 90-х годов на основе Си++. Он призван упростить разработку приложений на основе Си++ путем исключения из него всех низкоуровневых возможностей. Но главная особенность этого языка — компиляция не в машинный код, а в платформно-независимый байт-код, который потом выполняется с помощью интерпретатора — виртуальной Java-машины JVM (Java Virtual Machine), версии которой созданы сегодня для любых платформ. Благодаря наличию множества Java-машин программы на Java можно переносить не только на уровне исходных текстов, но и на уровне двоичного байт-кода.

Рейтинг языков программирования



Рейтинг языков программирования 2010

Позиция	Язык программирования	Рейтинг
1	Java (groovy- динамическая компиляция)	18.166% (-0.48%)
2	C	17.177% (+0.33%)
3 (+1)	C++	9.802% (-0.08%)
4 (-1)	PHP	8.323% (-2.03%)
5	(Visual) Basic (макросы MS)	5.650% (-3.04%)
6	C#	4.963% (+0.55%)
7	Python	4.860% (+0.96%)
8 (+4)	Objective-C (Apple (C&Smalltalk)	3.706% (+2.54%)
9 (-1)	Perl	2.310% (-1.45%)
10	Ruby	1.941% (-0.51%)
11 (-2)	Javascript	1.659% (-1.37%)
12 (-1)	Delphi (object Pascal)	1.558% (-0.58%)
13 (+4)	Lisp	1.084% (+0.48%)
14 (+10)	Transact-SQL	0.820% (+0.42%)
15	Pascal	0.771% (+0.10%)
16 (+2)	RPG	0.708% (+0.12%)
17 (+12)	Ada	0.704% (+0.40%)
18 (-4)	SAS	0.664% (-0.14%)
19	MATLAB	0.627% (+0.05%)
20 (new)	Go	0.626% (+0.63%)

Рейтинг языков программирования 2010

Что нового в данном рейтинге? В первую очередь это быстро набирающий обороты Objective-C от Apple, который набрал 2.54% за год. Появление в первой двадцатке языка Go уже было в начале этого года.

Помимо языков от Google и Apple в 2009 высокого уровня достигли C# от Microsoft и Actionscript от Adobe. Что касается Java, то он по прежнему остался на первой строчке, хотя его популярность продолжает падать, а Си снова медленно, но верно идёт к первому месту.

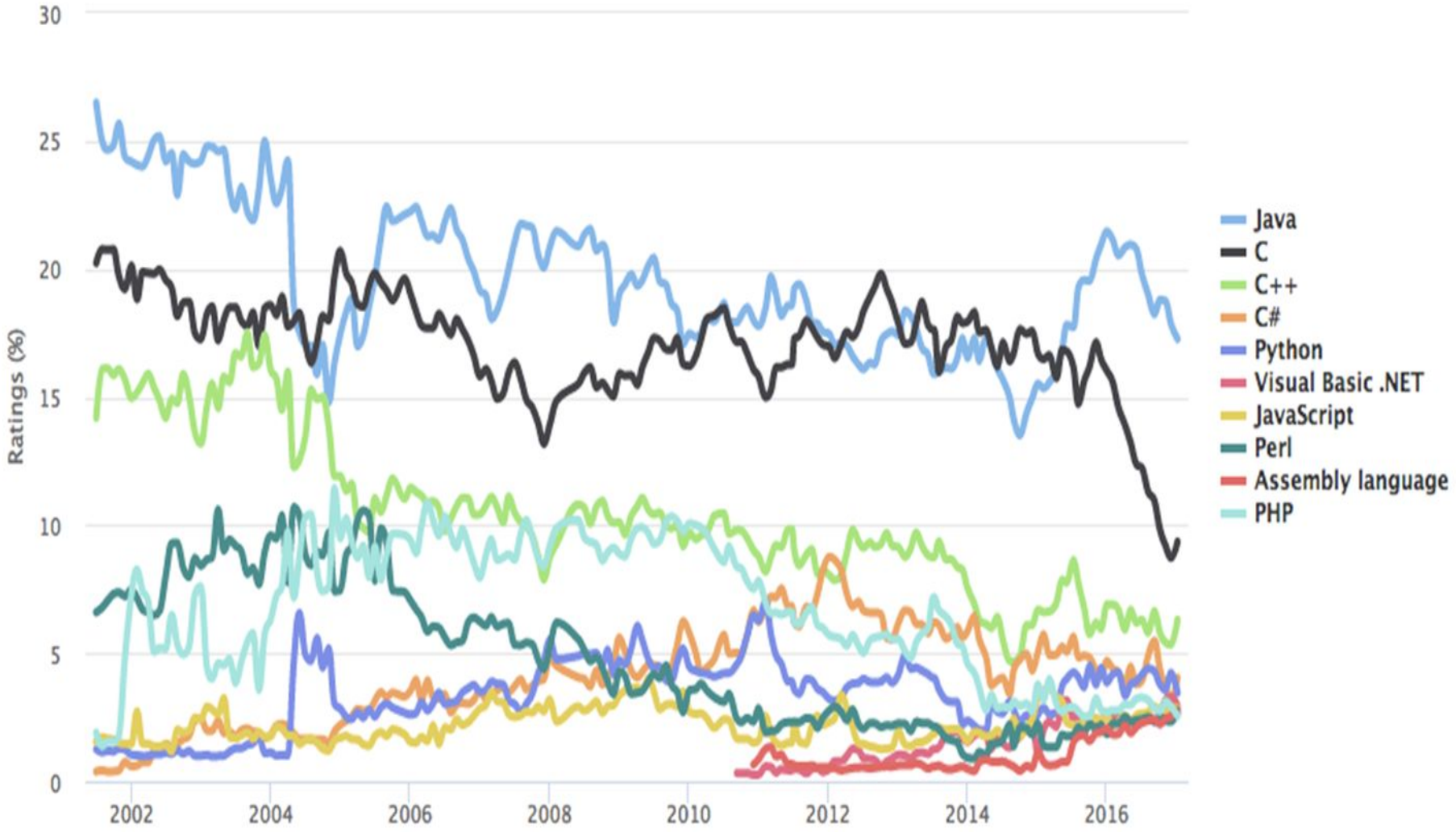
Интересно также неожиданное падение популярности PHP, и Javascript.

Пожалуй, самое удивительное — появление в двадцатке легендарного военного американского языка Ada, интерес к которому вознёс его аж на 17 строчку рейтинга (по сравнению с 29 местом год назад).

Рейтинг языков программирования 2017

Январь 2017	Январь 2016	Изменение	Язык программирования	Рейтинг	Изменение %
1	1		Java	17.278%	-4.19%
2	2		C	9.349%	-6.69%
3	3		C++	6.301%	-0.61%
4	4		C#	4.039%	-0.67%
5	5		Python	3.465%	-0.39%
6	7	^	Visual Basic.NET	2.960%	+0.38%
7	8	^	JavaScript	2.850%	+0.29%
8	11	^	Perl	2.750%	+0.91%
9	9		Assembly language	2.701%	+0.61%
10	6	⌄⌄	PHP	2.564%	-0.14%
11	12	^	Delphi/Object Pascal	2.561%	+0.78%
12	10	⌄	Ruby	2.546%	+0.50%
13	54	^^	Go	2.325%	+2.16%
14	14		Swift	1.932%	+0.57%
15	13	⌄	Visual Basic	1.912%	+0.23%
16	19	^	R	1.787%	+0.73%
17	26	^^	Dart	1.720%	+0.95%
18	18		Objective-C	1.617%	+0.54%
19	15	⌄⌄	MATLAB	1.578%	+0.35%
20	20		PL/SQL	1.539%	+0.52%

Рейтинг языков программирования



Lisp

LISP (Лисп). Интерпретируемый язык программирования, созданный в 1960 году Джоном Маккарти. Ориентирован на структуру данных в форме списка и позволяет организовывать эффективную обработку больших объемов текстовой информации.

SAS

Serial Attached SCSI (SAS) — последовательный компьютерный интерфейс, разработанный для подключения различных устройств хранения данных. SAS разработан для замены параллельного интерфейса SCSI и использует тот же набор команд SCSI.
Последняя реализация SAS обеспечивает передачу данных со скоростью до 12Гбит/с на одну линию. В 2017-му году появилась спецификация SAS со скоростью передачи данных 24Гбит/с.

MATLAB (matrix laboratory)

MATLAB — пакет прикладных программ для решения задач технических вычислений и одноимённый язык программирования, используемый в этом пакете. **MATLAB** используют более 1 млн. инженерных и научных работников, он работает на большинстве современных операционных систем, включая Linux, Mac, Solaris, Windows.

RPG (Report Program Generator)

RPG (Report Program Generator) — синтаксис изначально сходен с командным языком механических табуляторов компании IBM. Был разработан для облегчения перехода инженеров, обслуживавших эти табуляторы, на новую технику и переноса данных. Наиболее распространённой версией языка, по всей видимости, являлась RPG II.

Компания IBM продолжает поддержку языка и в настоящее время, так как на нём написан громадный объём кода, который невыгодно переводить на другие языки программирования.

PERL

Perl — высокоуровневый интерпретируемый динамический. Название языка представляет собой аббревиатуру, которая расшифровывается как Practical Extraction and Report Language — «практический язык для извлечения данных и составления отчётов».
Первоначально аббревиатура состояла из пяти символов и в таком виде в точности совпадала с английским словом pearl («жемчужина»). Но затем стало известно, что такой язык существует, и букву «a» убрали.
Основной особенностью языка считаются его богатые возможности для работы с текстом, в том числе работа с регулярными выражениями, встроенная в синтаксис. Перл унаследовал много свойств от языков Си и скриптовых языков командных оболочек UNIX.

Ada

Ada (Ада). Назван по имени леди Августы Ады Байрон, дочери английского поэта Байрона и его отдаленной родственницы Анабеллы Милбэнк. В 1980 году сотни экспертов Министерства обороны США отобрали из 17 вариантов именно этот язык, разработанный небольшой группой под руководством Жана Ишбиа. Он удовлетворил на то время все требования Пентагона, а к сегодняшнему дню в его развитие вложены десятки миллиардов долларов. Структура самого языка похожа на Паскаль. В нем имеются средства строгого разграничения доступа к различным уровням спецификаций, доведена до предела мощность управляющих конструкций.

Prolog

Prolog (Пролог). Создан в начале 70-х годов Аланом Колмероэ. Программа на этом языке, в основу которого положена математическая модель теории исчисления предикатов, строится из последовательности фактов и правил, а затем формулируется утверждение, которое Пролог будет пытаться доказать с помощью введенных правил. Человек только описывает структуру задачи, а внутренний «мотор» Пролога сам ищет решение с помощью методов поиска и сопоставления.

Smalltalk

Smalltalk (Смолток). Работа над этим языком началась в 1970 году в исследовательской лаборатории корпорации XEROX, а закончились спустя 10 лет, воплотившись в окончательном варианте интерпретатора SMALLTALK-80. Данный язык оригинален тем, что его синтаксис очень компактен и базируется исключительно на понятии объекта. В этом языке отсутствуют операторы или данные. Все, что входит в Смолток, является объектами, а сами объекты общаются друг с другом исключительно с помощью сообщений (например, появление выражения 1+1 вызывает посылку объекту 1 сообщения «+», то есть «прибавить», с параметром 1, который считается не числом-константой, а тоже объектом). Больше никаких управляющих структур, за исключением «оператора» ветвления в языке нет, хотя их можно очень просто смоделировать. Сегодня версия Visual Age for Smalltalk активно развивается компанией IBM.

Популярные языки программирования

1

Java - один из самых популярных языков для back-end разработки современных корпоративных веб-приложений. И это его основное преимущество. С помощью языка Java и его фреймворков веб-разработчики могут создавать масштабируемые веб-приложения для широкого круга пользователей. Java - основной язык для разработки нативных приложений (отдельная программа с собственным программным обеспечением) под Android и других приложений для смартфонов и планшетов. Большим плюсом этого языка называют WORA ("Write once, run anywhere”) - "пишешь один раз, работает везде" принцип, объявленный SunMicrosystems, чтобы доказать кросс-платформенность Java. Но этот плюс не отменяет тот факт, что этот язык работает медленнее, чем другие подобные.

Популярные языки программирования

2

JavaScript - популярный язык среди молодых разработчиков. Он подходит для создания интерактивности сайта, или построения пользовательских интерфейсов с помощью одного из десятков популярных фреймворков. Этот язык редко встретишь за пределами браузера, вероятно, потому, что это единственное место, где он полезен. Тем не менее, JavaScript стоит учить, не только потому, что он и его десятки фреймворков набирают популярность, но и потому, что в будущем язык позволит создавать более сложные вещи.

Популярные языки программирования

3

C# стоит учить, потому что его знание поможет достаточно легко получить работу. Это самый популярный язык сейчас для разработки приложений для Windows, и очень популярный для мобильных устройств. Кроме того, движок для разработки игр Unity также использует C# в качестве одного из основных языков. Он очень похож на другие объектно-ориентированные языки программирования и достаточно легко учится при наличии базовых знаний C++ или Java.

Популярные языки программирования

4

PHP Хороший язык для создания веб-приложений для работы с данными. Это основная технология для создания мощных систем управления контентом, которые впоследствии можно расширять, чтобы сделать сайт более мощным. Был подвергнут критике как небезопасный язык, однако ситуация изменилась к лучшему после обновления в 2004 году. Тем, кто хочет учить PHP рекомендуется знать HTML, CSS и Javascript.

Популярные языки программирования

5

C++ Созданный в 1979 году, язык по-прежнему очень популярен и используется для построения различных типов приложений - от игр до офисных приложений. C++ предназначен для системного программирования, и подходит для разработки мощного программного обеспечения, аппаратно-ускоренных игр и приложений, требующих больших объемов памяти на настольных компьютерах, консолях и мобильных устройствах. Среди недостатков C++ программисты называют «неуклюжесть» в сравнении с Java.

Популярные языки программирования

6

Python Стоит изучить хотя бы потому, что Python - выбор Google и Ubuntu (операционная система, основанная на Debian GNU/Linux. Основным разработчиком и спонсором является компания Canonical). Но это не единственная положительная особенность языка Python, среди которых также его отличная читаемость и элегантный код. Python не требует такое количество кода для выполнения программы, как другие языки. Синтаксис ядра минималистичен, но библиотека включает большой объем полезных функций.

Популярные языки программирования

7

Ruby - динамический, рефлективный, интерпретируемый высокоуровневый язык программирования для быстрого и удобного объектно-ориентированного программирования. Язык простой в освоении и невероятно мощный, плюс на нем написаны тысячи популярных веб-приложений по всему миру. Если вы любите объекты, этот язык вам подходит. Его основным преимуществом является скорость. Ruby очень похож на Python, но менее "человеческий".

Перспективные языки программирования

1

Erlang - функциональный язык программирования, разработанный компанией Ericsson, для разработки распределенных систем реального времени. Его главная особенность - параллельность. Его стоит изучать, потому что крупные банки с миллионами пользователей используют Erlang для банковских систем.

Перспективные языки программирования

2

R - Широко используется для разработки статистического программного обеспечения, но не очень популярен среди разработчиков. Этот язык рекомендуется знать тем, кто нуждается в серьезном анализе данных. Он работает на всех платформах и интегрируется со многими языками программирования, такими как Java, Ruby, C++, Python. Хотя он и не так популярен сейчас, ситуация может измениться в лучшую сторону. В январе 2015 Microsoft приобрела компанию Revolution Analytics, по их словам, для вклада в дальнейшее развитие языка R.

Перспективные языки программирования

3

Язык программирования **Swift** захватил разработчиков, как новый, более быстрый и легкий путь разрабатывать под Mac и iOS, по сравнению с Objective-C. Тем не менее, он актуален только в экосистеме Apple. Хороший для Apple - плохой для разработчика, который не хочет работать исключительно для Apple, особенно учитывая популярность Android. Стоит учить, если вы хотите внести свой вклад в мир игр на iOS.

Перспективные языки программирования

4

Go (Golang) В Интернете гораздо больше информации о том, почему больше Go плохой язык, чем хороший. Этот язык разработан Google. Так, по данным Google, Go обеспечивает фундаментальную поддержку параллельного выполнения программ и коммуникации, и предлагает подход к построению системного программного обеспечения на многоядерных компьютерах. Этот язык может быть включен в список перспективных, однако есть определенные сомнения в отношении его будущего.

Go (Golang)

Он мультипоточен, со строгой типизацией, высокоуровневый. Есть поддержка определений функций и процедур, а также взаимосвязей между ними. Потоки исполняют процедуры, вызывают функции и запрашивают необходимые связи по мере надобности. Потоки разных агентов взаимодействуют между собой при помощи асинхронных сообщений. Потоки одного агента могут устанавливать динамические связи друг с другом, образовывая своего рода общую память.

Сверхвысокоуровневый язык программирования

Сверхвысокоуровневый язык программирования (язык программирования сверхвысокого уровня, англ. *very high-level programming language*, **VHLL**) —язык программирования с очень высоким уровнем абстракции. В отличие от языков программирования высокого уровня, где описывается принцип «**как нужно сделать**», в сверхвысокоуровневых языках программирования описывается лишь принцип «**что нужно сделать**». Термин впервые появился в середине 1990-х годов для обозначения группы языков, используемых для быстрого прототипирования, написания одноразовых скриптов и подобных задач (Ruby, Haskell, Perl, AWK).

Другие языки программирования

Haskell. стандартизированный функциональный язык программирования общего назначения. Является одним из самых распространённых языков программирования с поддержкой **отложенных вычислений**. Поскольку язык функциональный, то основная управляющая структура — это функция.

Есть встроенная поддержка многозадачного и параллельного программирования, развитый инструментарий (средства автоматического тестирования, отладки и профилирования, в том числе для параллельных программ), существует несколько тысяч библиотек с открытым исходным кодом.

Другие языки программирования

AWK. интерпретируемый скриптовый С-подобный язык построчного разбора и обработки входного потока (например, текстового файла) по заданным шаблонам (регулярным выражениям). AWK рассматривает **входной поток как список записей**. Каждая запись делится на поля. На основе этой информации выполняется некоторый определённый программистом алгоритм обработки. По умолчанию разделителем записей является символ новой строки, разделителем полей — символ пробела или табуляции, или последовательность таких символов. Символы-разделители можно явно определить в программе. Символ-разделитель полей можно определить и в командной строке. Каждая **запись поочерёдно сравнивается со всеми шаблонами**, и каждый раз когда она соответствует шаблону, выполняется указанное действие.

Другие языки программирования

Delphi (Object Pascal) — императивный, структурированный, объектно-ориентированный язык программирования со строгой статической типизацией переменных. Основная область использования — написание прикладного программного обеспечения (Windows, Mac OS, iOS, Android). Структурированный язык запросов **SQL (Structured Query Language)**. Он основан на мощной математической теории и позволяет выполнять эффективную обработку баз данных, манипулируя не отдельными записями, а группами записей**.**
HTML. Общеизвестный язык для оформления документов. Он очень прост и содержит элементарные команды форматирования текста, добавления рисунков, задания шрифтов и цветов, организации ссылок и таблиц.

Другие языки программирования

Tcl/Tk *(Tool Command Language)***.** В конце 80-х годов Джон Оустерхаут придумал скрипт-язык Tel и библиотеку Tk. В Tel он попытался воплотить видение идеального скрипт-языка. Tel ориентирован на **автоматизацию рутинных процессов и состоит из мощных команд**, предназначенных для работы с абстрактными нетипизированными объектами. Он независим от типа системы и при этом позволяет создавать программы с графическим интерфейсом.

VRML (предок X3D). В 1994 году был создан язык VRML для организации виртуальных трехмерных интерфейсов в Интернете. Он позволяет описывать в текстовом виде различные трехмерные сцены, освещение и тени, текстуры (покрытия объектов), создавать свои миры, путешествовать по ним, «облетать» со всех сторон, вращать в любых направлениях, масштабировать, регулировать освещенность и т. д.

Другие языки программирования

PL/I (ПЛ/1). **В середине 60-х годов компания IBM решила взять все лучшее из языков Фортран, Кобол и Алгол. В результате в 1964 году на свет появился новый компилируемый язык программирования, который получил название Programming Language One. В этом языке было реализовано множество уникальных решений. По своим возможностям ПЛ/1 значительно мощнее многих других языков (Си, Паскаля). Этот язык и сегодня продолжает поддерживаться компанией IBM.**

Другие языки программирования

Forth (Форт). **Результат попытки Чарльза Мура в 70-х годах создать язык, обладающий мощными средствами программирования, который можно эффективно реализовать на компьютерах с небольшими объемами памяти, а компилятор мог бы выдавать очень быстрый и компактный код — то есть служил заменой ассемблеру. Однако сложности восприятия программного текста, записанного в непривычной форме, сильно затрудняли поиск ошибок, и с появлением Си язык Форт оказался забытым.**

Другие языки программирования

Командный интерпретатор Shell, используемый в операционных системах семейства UNIX, в котором пользователь может либо **давать команды операционной системе по отдельности, либо запускать скрипты, состоящие из списка команд**.

Основное назначение этого языка состоит в том, чтобы предоставить пользователю удобные средства взаимодействия с системой.

Скриптовые языки и языки программирования

1

Bash (Bourne Again shell), Csh (C shell), JavaScript, Ksh (Korn shell), Perl, Python, Ruby, Sh (Bourne shell), Tcl

2

Assembler, C, C++, C#, FORTRAN, Forth, Java, LISP, Pascal

CASE (computer-aided software engineering)

Эволюция CASE-средств

- **CASE-технологии развивались с целью преодоления ограничений ручных применений, методологий структурного анализа и проектирования. CASE-технологии не являются самостоятельными методологиями и используются для более эффективного их применения.**

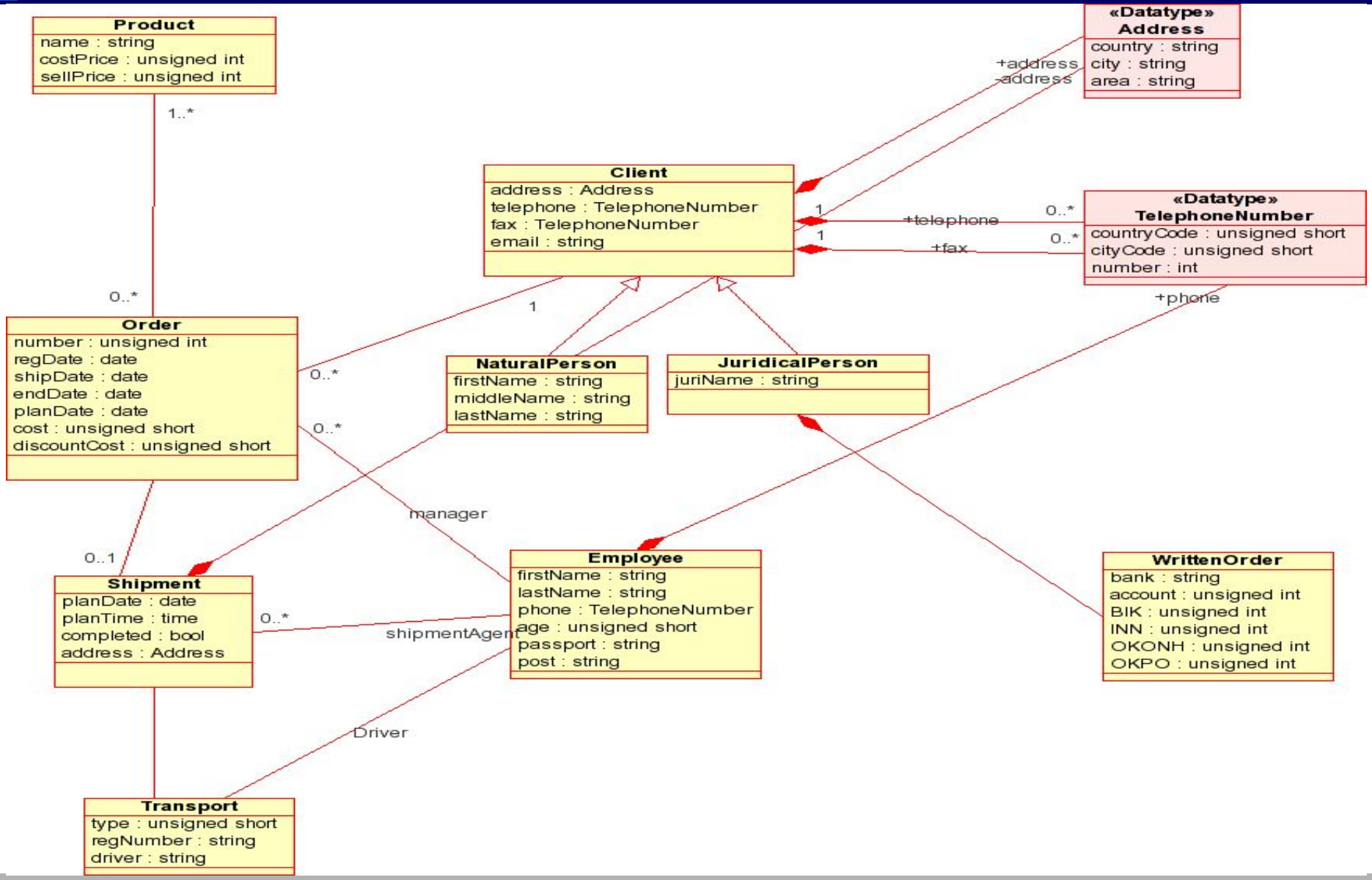
Выделяют 2 генерации CASE-средств:

- ***CASE-средства анализа требований, проектирования спецификаций и структуры, редактирования интерфейсов (первая генерация CASE-I);***
- ***CASE-средства генерации исходных текстов и реализации интегрированного окружения поддержки полного жизненного цикла (ЖЦ) разработки программного обеспечения (ПО) (вторая генерация CASE-II).***

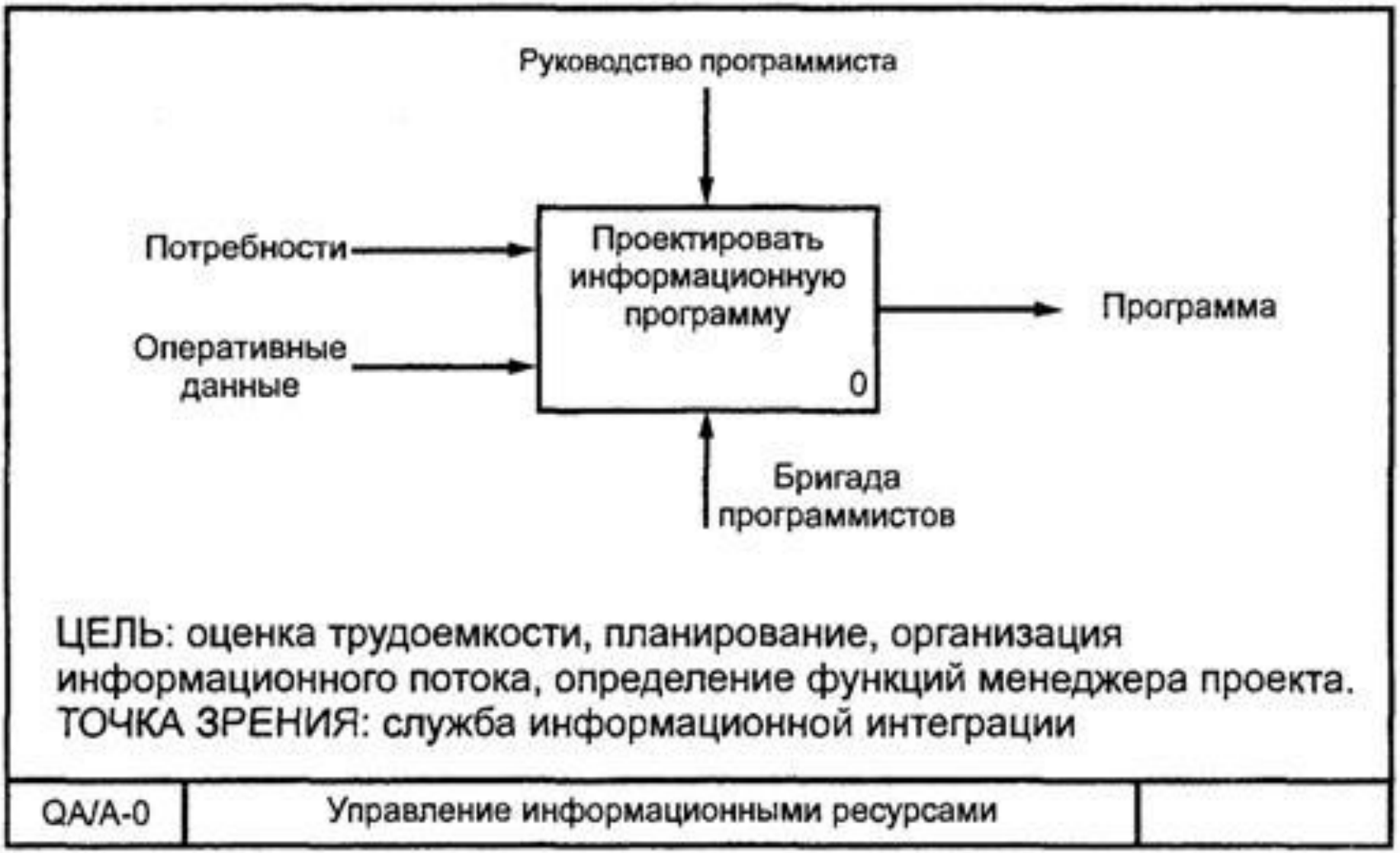
Case- технологии

Основной целью CASE-технологии является разграничение процесса проектирования программных продуктов от процесса кодирования и последующих этапов разработки, максимально автоматизировать процесс разработки. Для выполнения поставленной цели CASE-технологии используют два принципиально разных подхода к проектированию: **структурный (IDEF) и объектно-ориентированный (UML)**.

Построение UML- диаграмм

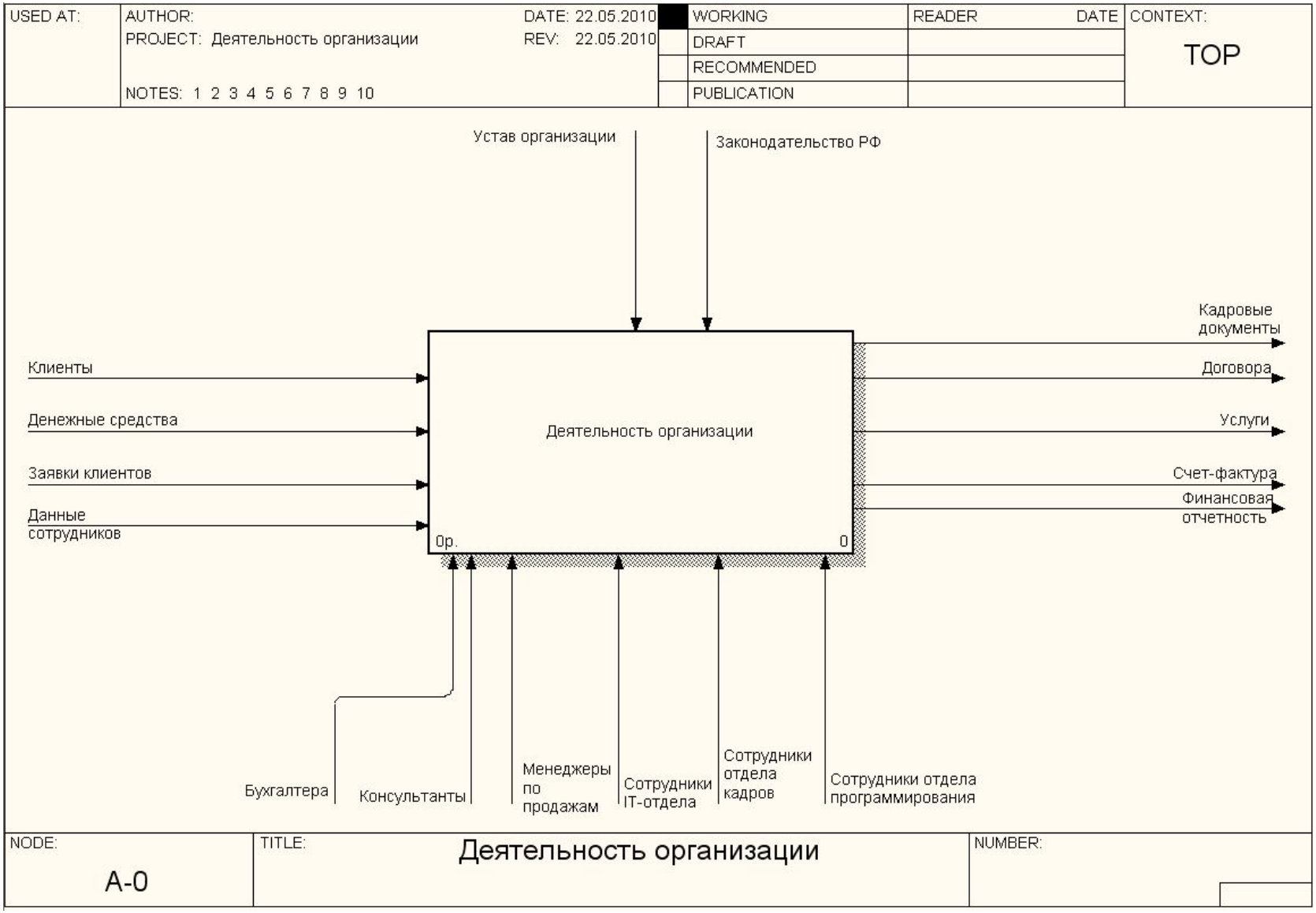


Построение IDEF0- диаграмм



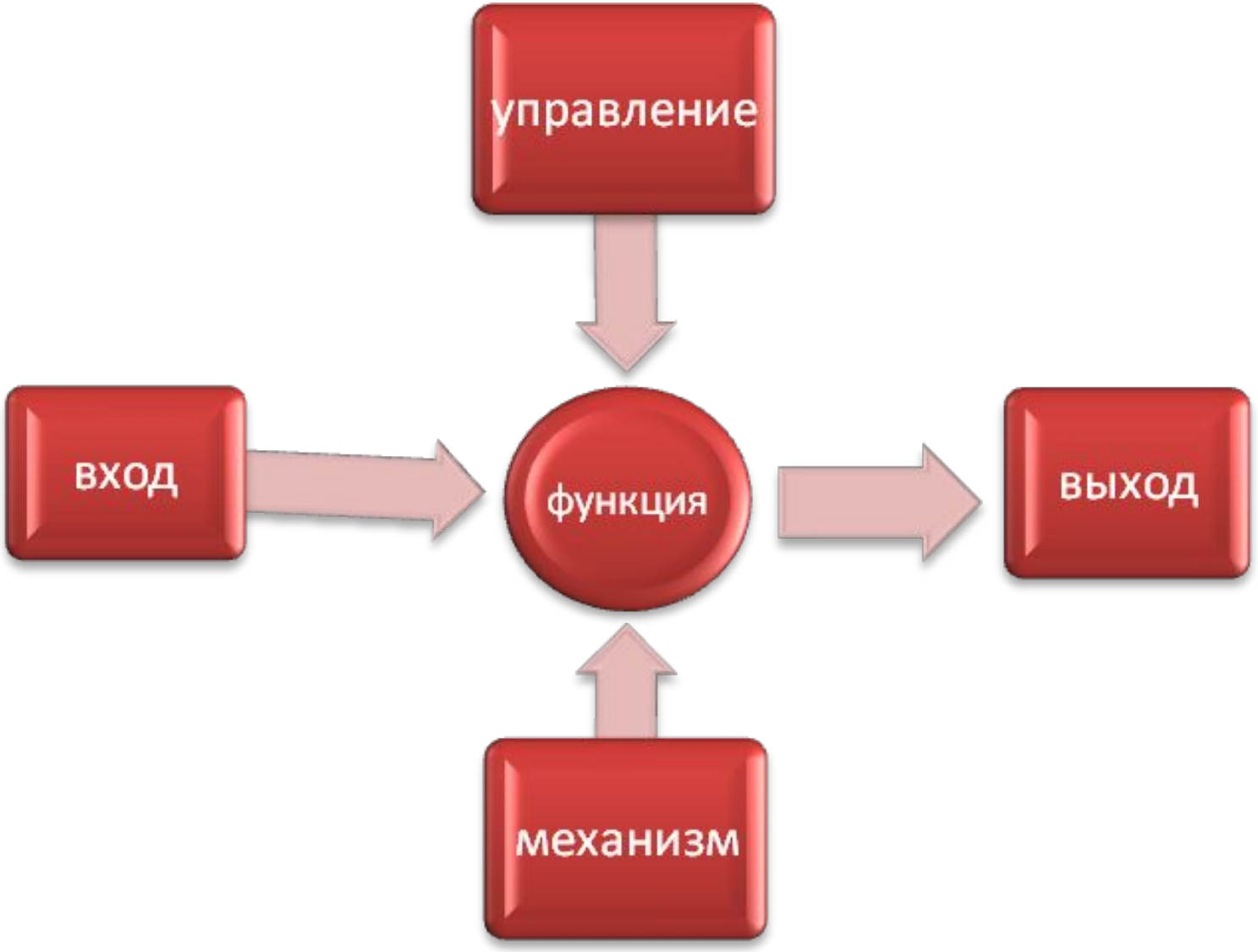
Построение IDEF0- диаграмм

Построение IDEF0- диаграмм

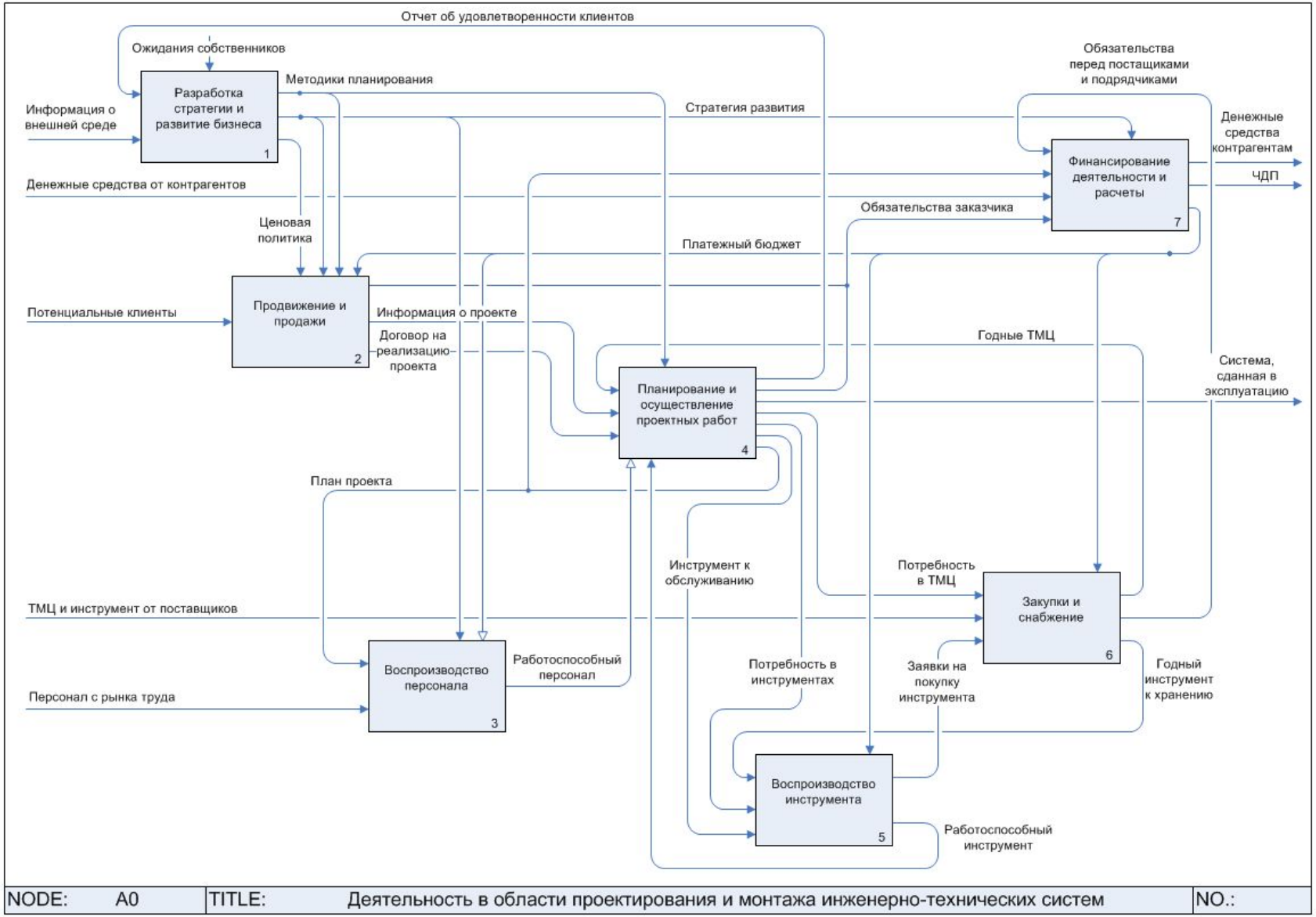


Построение IDEF0- диаграмм

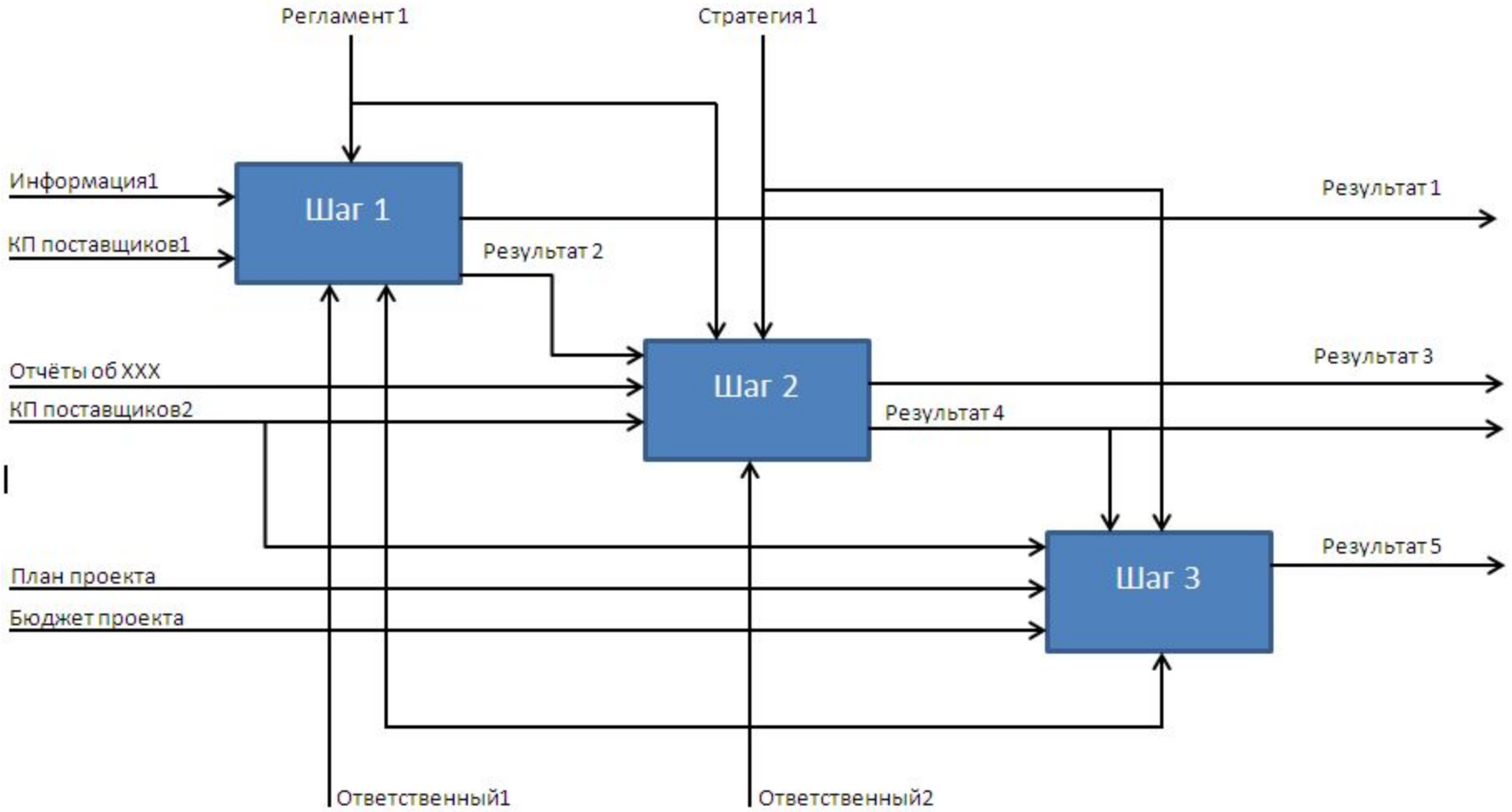
Построение IDEF0- диаграмм



Построение IDEF0- диаграмм



Построение IDEF0- диаграмм



Разработка, управляемая моделями

Model-driven development (MDD)— это стиль разработки программного обеспечения, когда модели становятся основными артефактами разработки, из которых генерируется код и другие артефакты

Разработка, управляемая моделями

Модель — это абстрактное описание программного обеспечения, которое скрывает информацию о некоторых аспектах с целью представления упрощенного описания остальных. Модель может быть исходным артефактом в разработке, если она фиксирует информацию в форме, пригодной для интерпретаций людьми и обработки инструментами. **Модель определяет нотацию и метамодель**. **Нотация** представляет собой совокупность графических элементов, которые применяются в модели и могут быть интерпретированы людьми. **Метамодель** описывает понятия используемые в модели и фиксирует информацию в виде метаданных, которые могут быть обработаны инструментами.

Модели описанные на предметно-ориентированном языке программирования могут быть использованы, как точки расширения каркасов.

Архитектура, управляемая моделью

Model Driven Architecture, MDA — создаваемая консорциумом OMG разновидность концепции «Разработка, управляемая моделями»: модельно-ориентированного подхода к разработке программного обеспечения. Его суть состоит в построении абстрактной метамодели управления и обмена метаданными (моделями) и задании способов её трансформации в поддерживаемые технологии программирования (Java, CORBA, XML и др.). Создание метамодели определяется технологией моделирования **MOF (Meta Object Facility)**, являющейся частью концепции MDA. Название концепции не совсем удачно, так как она определяет вовсе не архитектуру а именно метод разработки программного обеспечения.

Архитектура, управляемая моделью

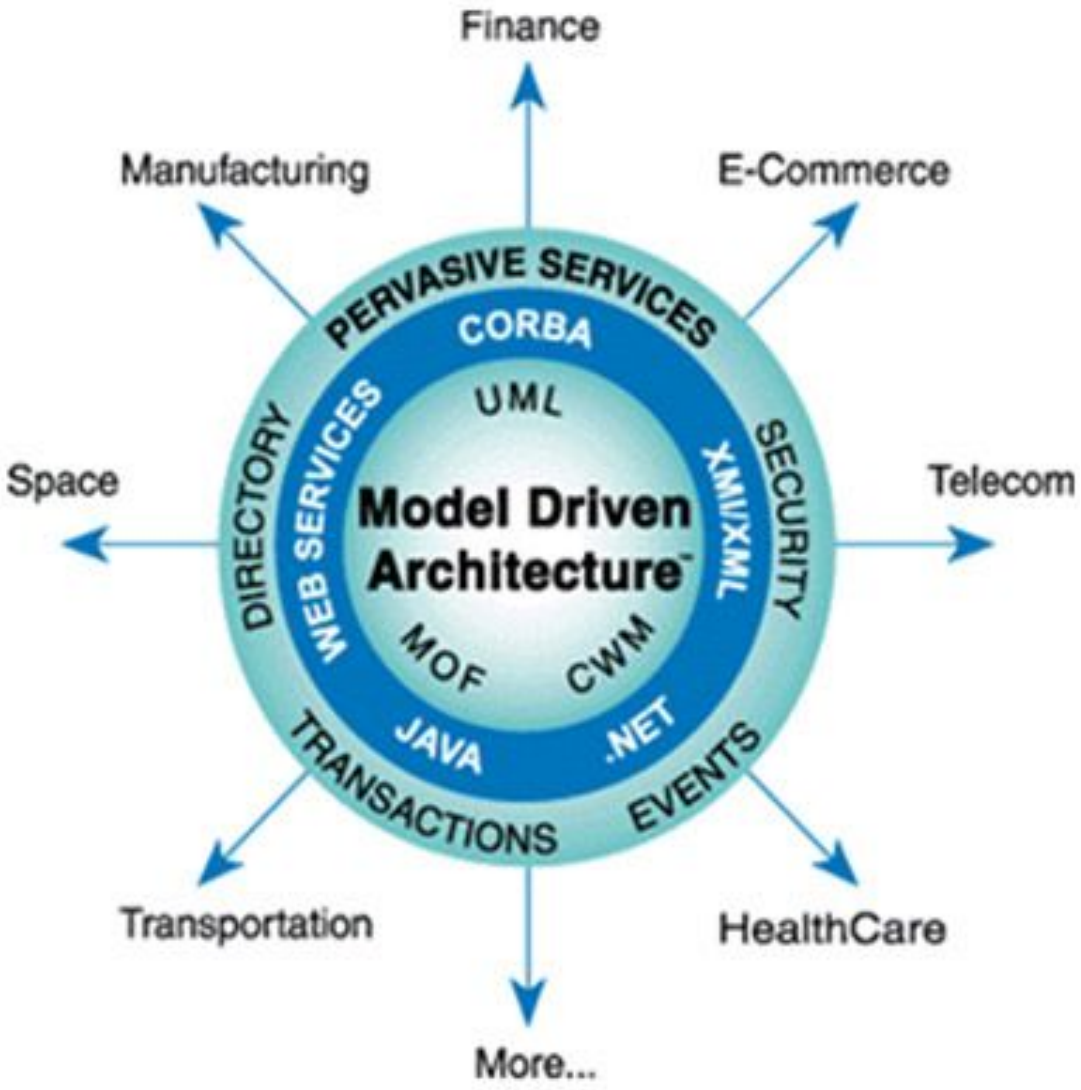
Для конструирования программного приложения должна быть построена подробная, формально точная модель, из которой потом может быть автоматически генерирован исполняемый программный код приложения.

Основные шаги разработки:

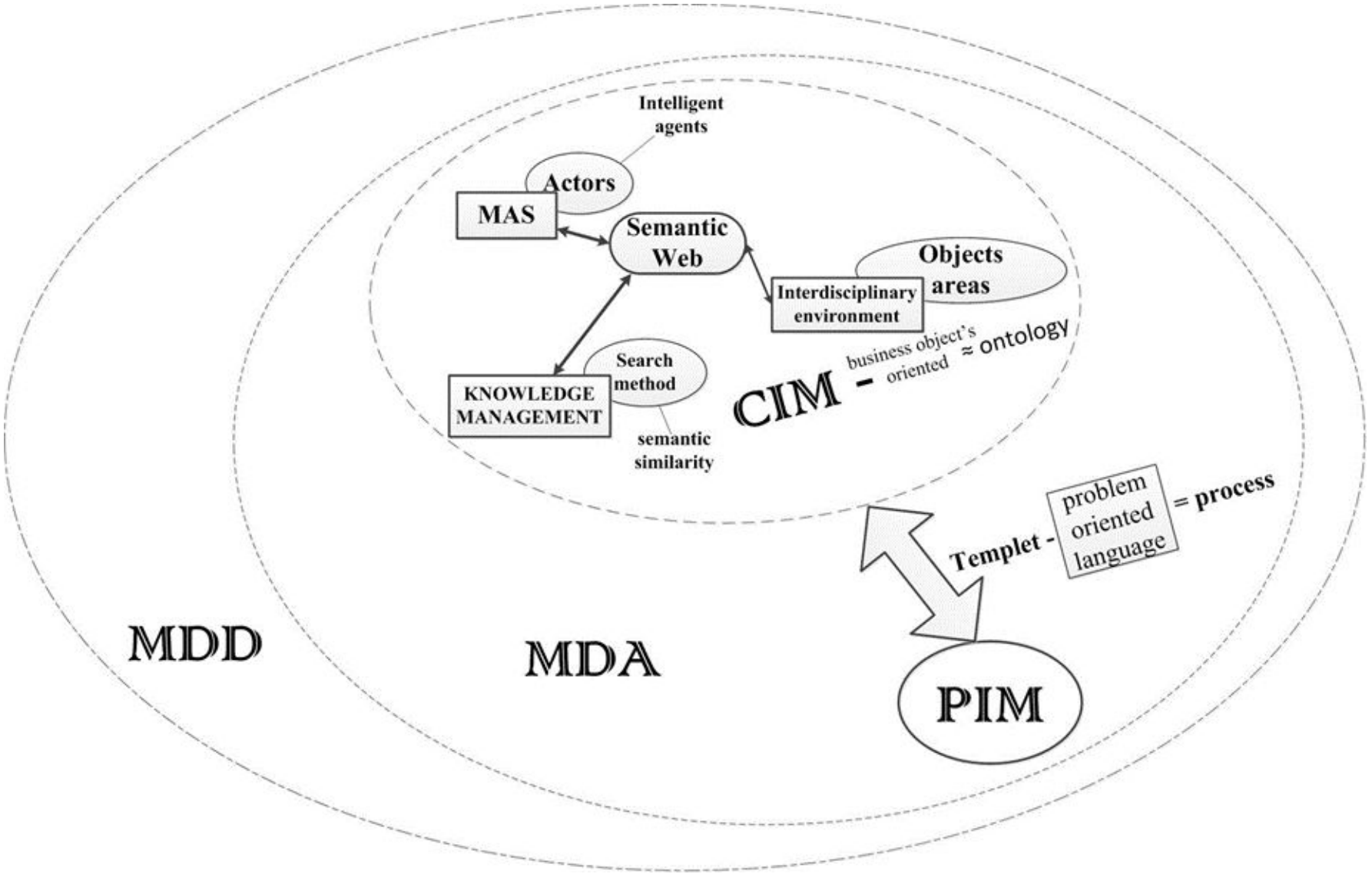
1. Сначала разрабатывается модель предметной области проектируемого приложения, полностью независимая от имплементирующей технологии. Она называется **Platform Independent Model (PIM).**
2. Затем PIM автоматически трансформируется специальным инструментом в платформо-зависимую модель **(Platform Specifical Model, PSM).**
3. **PSM переводится в исходный код** на соответствующем языке программирования.

Такова схема в идеальном OMG-мире. В реальных современных проектах часть бизнес-логики приходится реализовать вручную. Но поскольку этот код отделен от генерированного системой, большой проблемы это не представляет.

Архитектура, управляемая моделью



Архитектура, управляемая моделью



ОСНОВЫ CASE-ТЕХНОЛОГИЙ

CASE-средства анализа требований, проектирования спецификаций и структуры, редактирования интерфейсов

CASE-средства генерации исходных текстов и реализации интегрированного окружения поддержки полного жизненного цикла (ЖЦ) разработки программного обеспечения (ПО)

Модель жизненного цикла программного обеспечения

Case - модель жизненного цикла программного обеспечения

ОСНОВЫ CASE-ТЕХНОЛОГИЙ

Во сколько раз сокращает время на кодирование и тестирование применение Case- технологий?

ОСНОВЫ CASE-ТЕХНОЛОГИЙ

Способ разработки	Анализ	Проектирование	Кодирование	Тестирование
Традиционная разработка	20%	15%	20%	45%
Использование структурных методологий проектирования	30%	30%	15%	25%
Использование CASE-технологий	40%	40%	5%	15%

ОСНОВЫ CASE-ТЕХНОЛОГИЙ

Графические модели

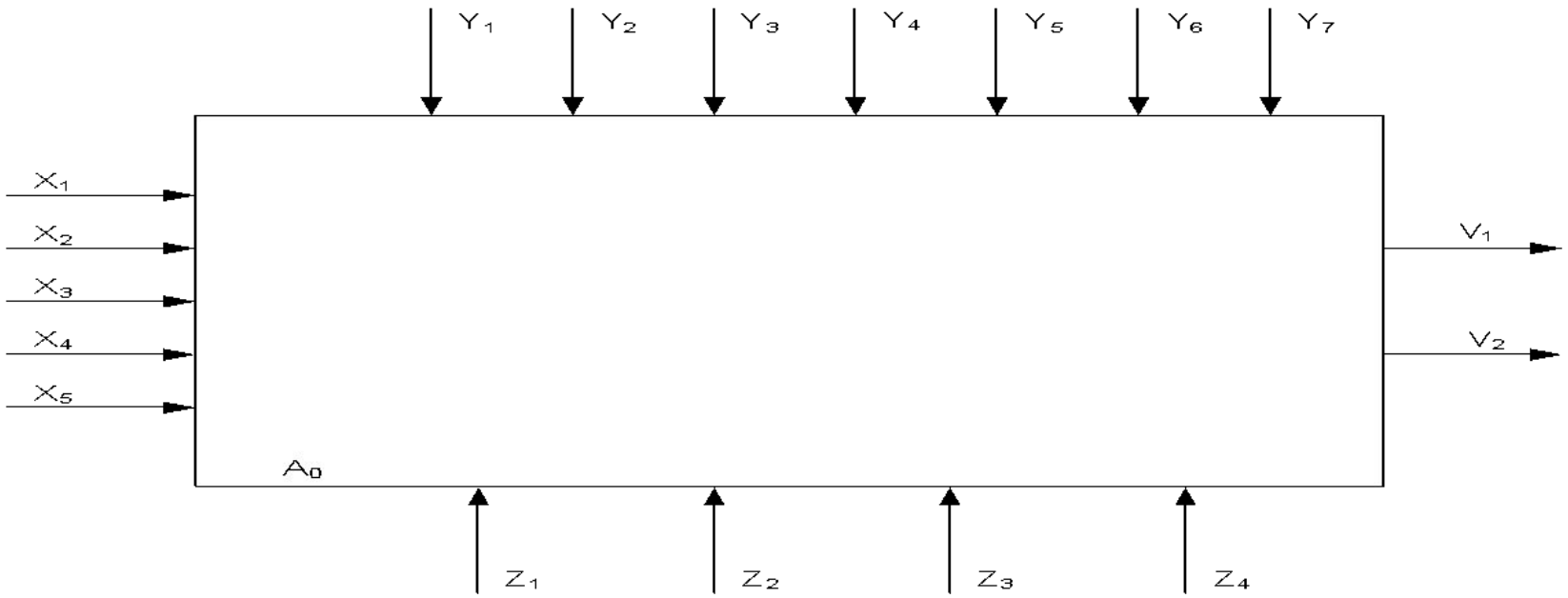
- CASE-программы являются схематическими проектами и формами, что упрощает восприятие. Для представления программ применяются структурные диаграммы различных типов. Для CASE существует 4 типа диаграмм:

1. *Диаграммы функционального проектирования (DFD)* – диаграммы потоков данных.
2. *Диаграммы моделирования данных (ERD)* – диаграммы «сущность-связь».
3. *Диаграммы моделирования поведения (STD)* – диаграммы переходов состояний.
4. *Структурные диаграммы* (карты).

Основные компоненты интегрированного CASE- пакета

1. Репозиторий
2. Средства ввода данных в репозиторий
3. Средства анализа проектирования
4. Средства вывода

SADT – Structured Analysis and Design Technique



[A0]

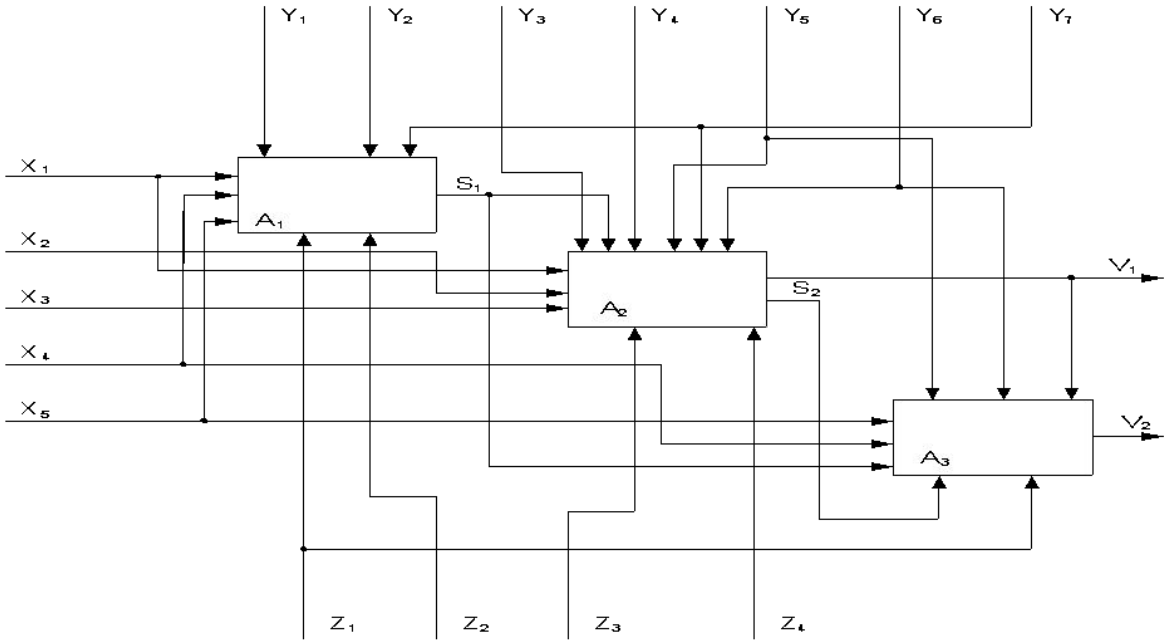
Inputs: X_1, X_2, X_3, X_4, X_5;

Outputs: V_1, V_2;

Controls: Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_7;

Mechanisms: Z_1, Z_2, Z_3, Z_4;

SADT – Structured Analysis and Design Technique



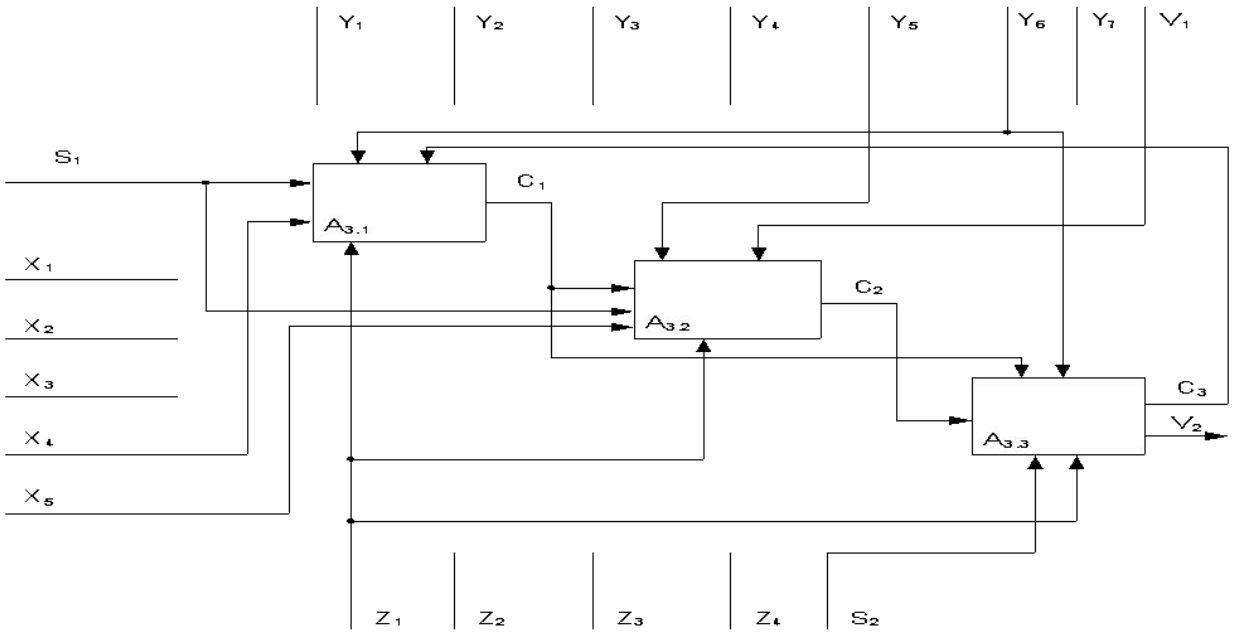
Sub-Activities:

[A1]
Inputs: X1, X4, X5;
Outputs: S1;
Controls: Y1, Y2, Y7;
Mechanisms: Z1, Z2;

[A2]
Inputs: X1, X2, X3;
Outputs: V1, S2;
Controls: Y3, Y4, Y5, Y6, Y7, S1;
Mechanisms: Z3, Z4;

[A3]
Inputs: X4, X5, S1;
Outputs: V2;
Controls: Y5, Y6, V1;
Mechanisms: Z1, S2;

SADT – Structured Analysis and Design Technique



[A3.1]
Inputs: X_4, S_1;
Outputs: C_1;
Controls: Y_6, C_3;
Mechanisms: Z1;

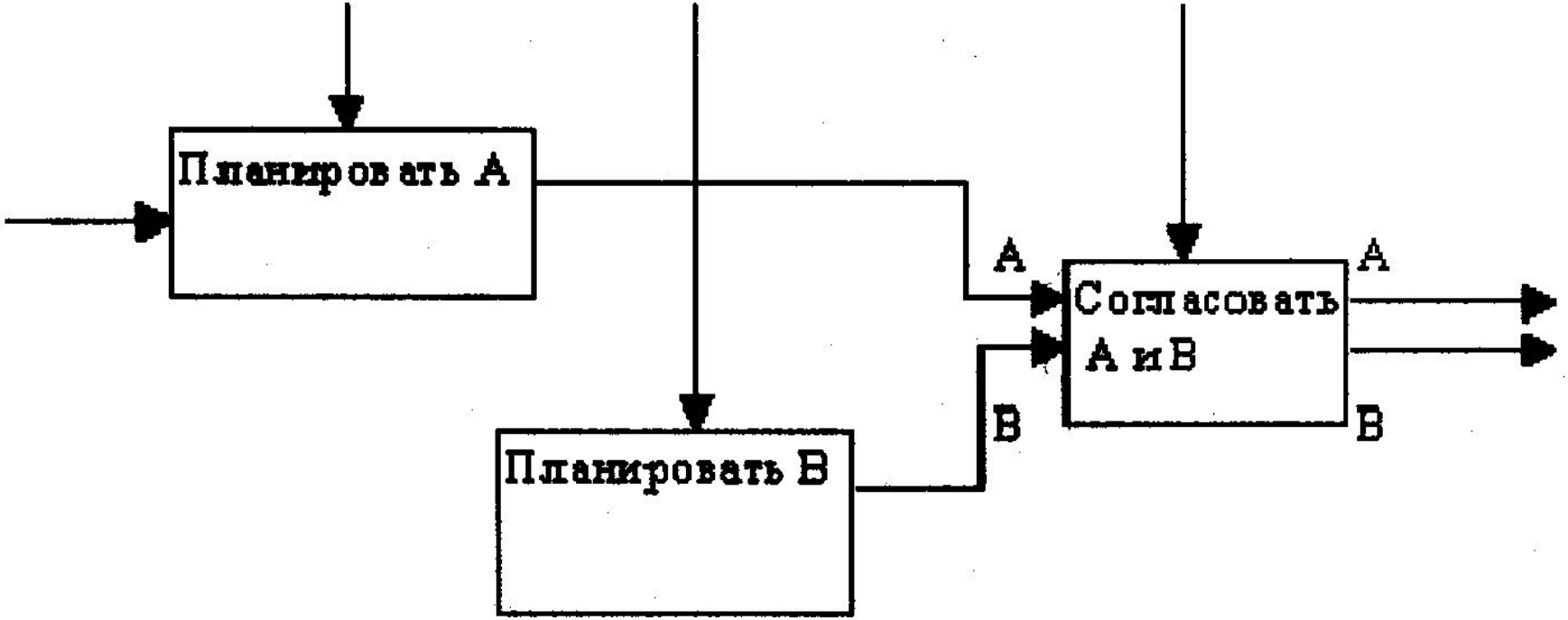
[A3.2]
Inputs: X_5, S_1, C_1;
Outputs: C_2;
Controls: Y_5, V_1;
Mechanisms: Z_1;

[A3.3]
Inputs: C_2;
Outputs: V_2, C_3;
Controls: Y_6, C_1;
Mechanisms: Z_1, S_2.

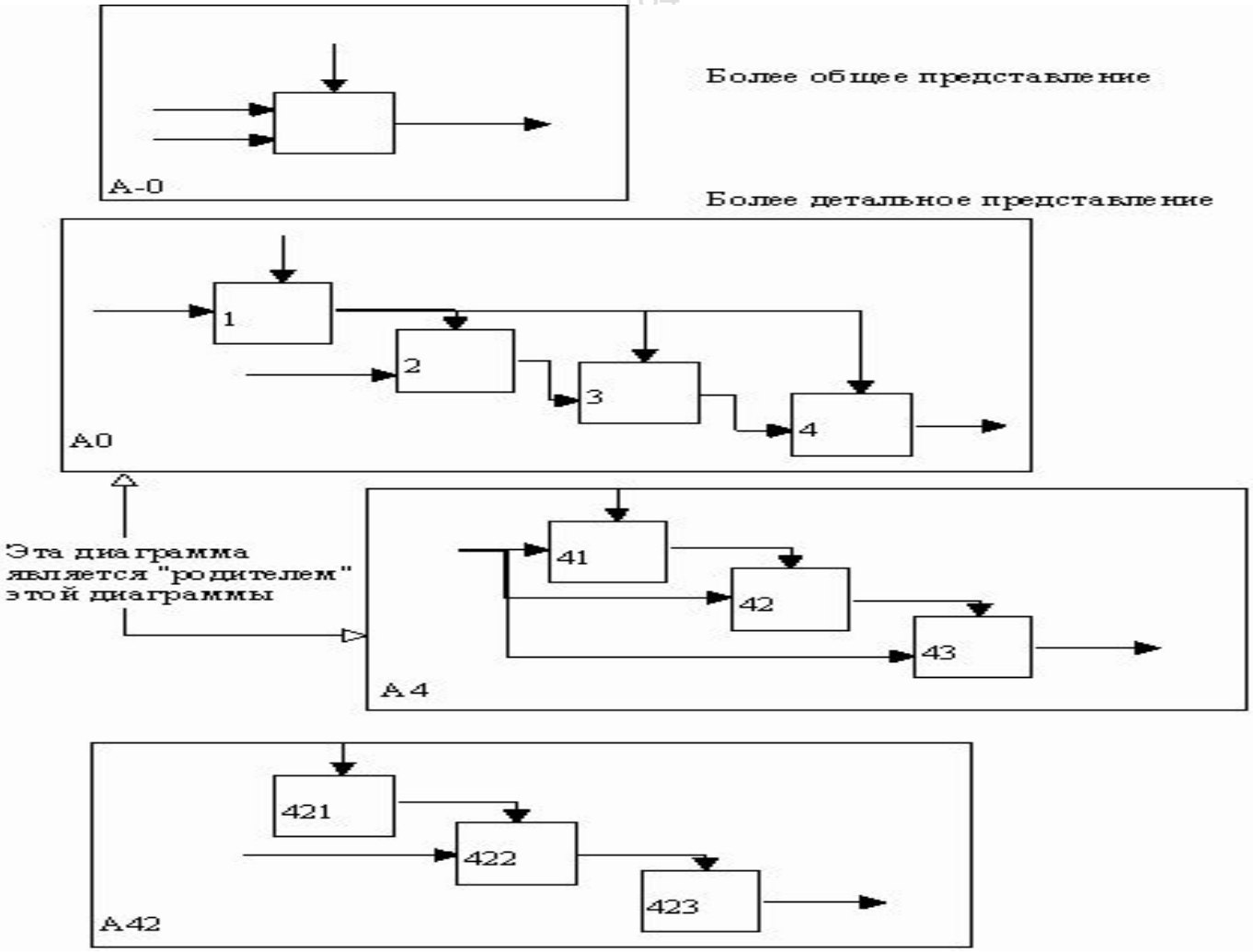
СТРУКТУРНЫЙ ПОДХОД ПРОЕКТИРОВАНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ (ИС)

Тип связи	Относительная значимость
Случайная	**0**
Логическая	**1**
Временная	**2**
Процедурная	**3**
Коммуникационная	**4**
Последовательная	**5**
Функциональная	**6**

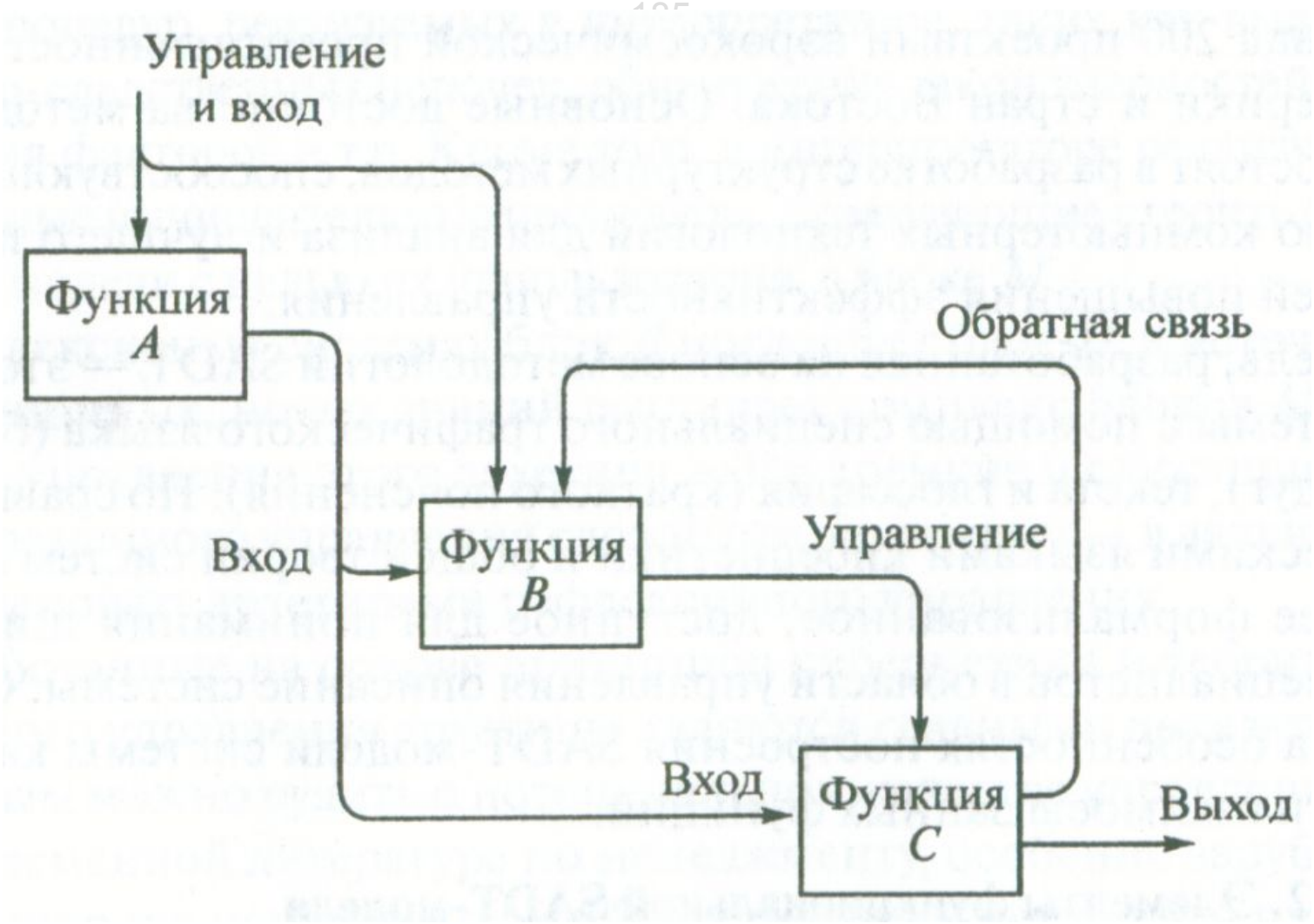
КОММУНИКАЦИОННАЯ СВЯЗЬ



ПОСЛЕДОВАТЕЛЬНАЯ СВЯЗЬ



ФУНКЦИОНАЛЬНАЯ СВЯЗЬ



ARIS

Одной из современных методологий бизнес-моделирования, получившей широкое распространение является методология **ARIS**, которая расшифровывается как Architecture of Integrated Information Systems - проектирование интегрированных информационных систем. Ее использует программное средство ARIS Toolset

ARIS

Одна из главных идей ARIS состоит в разделении создаваемых моделей на *четыре типа*:

организационные модели – представляют моделируемую систему как иерархию организационных подразделений, должностей и конкретных лиц, связи между ними, а также территориальную привязку подразделений;

функциональные модели – содержат иерархию целей, стоящих перед аппаратом управления, и комплекс функций, необходимых для достижения поставленных целей;

информационные модели – отражают информацию, необходимую для реализации функций системы;

модели управления – отражают реализацию управленческих процессов в системе.

ARIS


Кто?
Оргструктура
Что?
Информация
Функции
На основе Чего?
Процессы
Каким образом?

ARIS

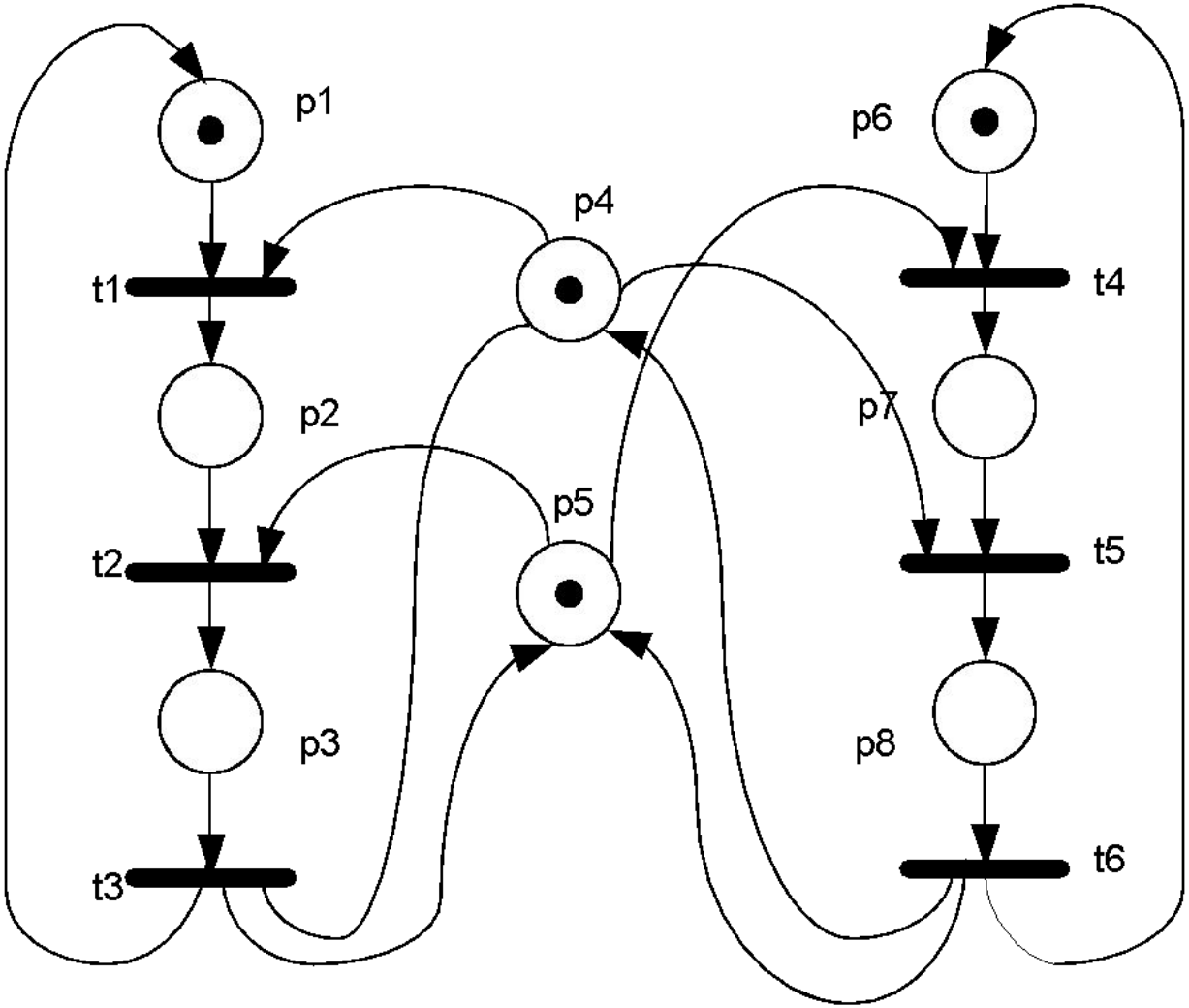
Другой особенностью ARIS является использование различных уровней описания:

Уровень определения требований. На данном уровне разрабатываются модели системы - как она организована, какие деловые процессы в ней присутствуют, какие данные при этом используются;

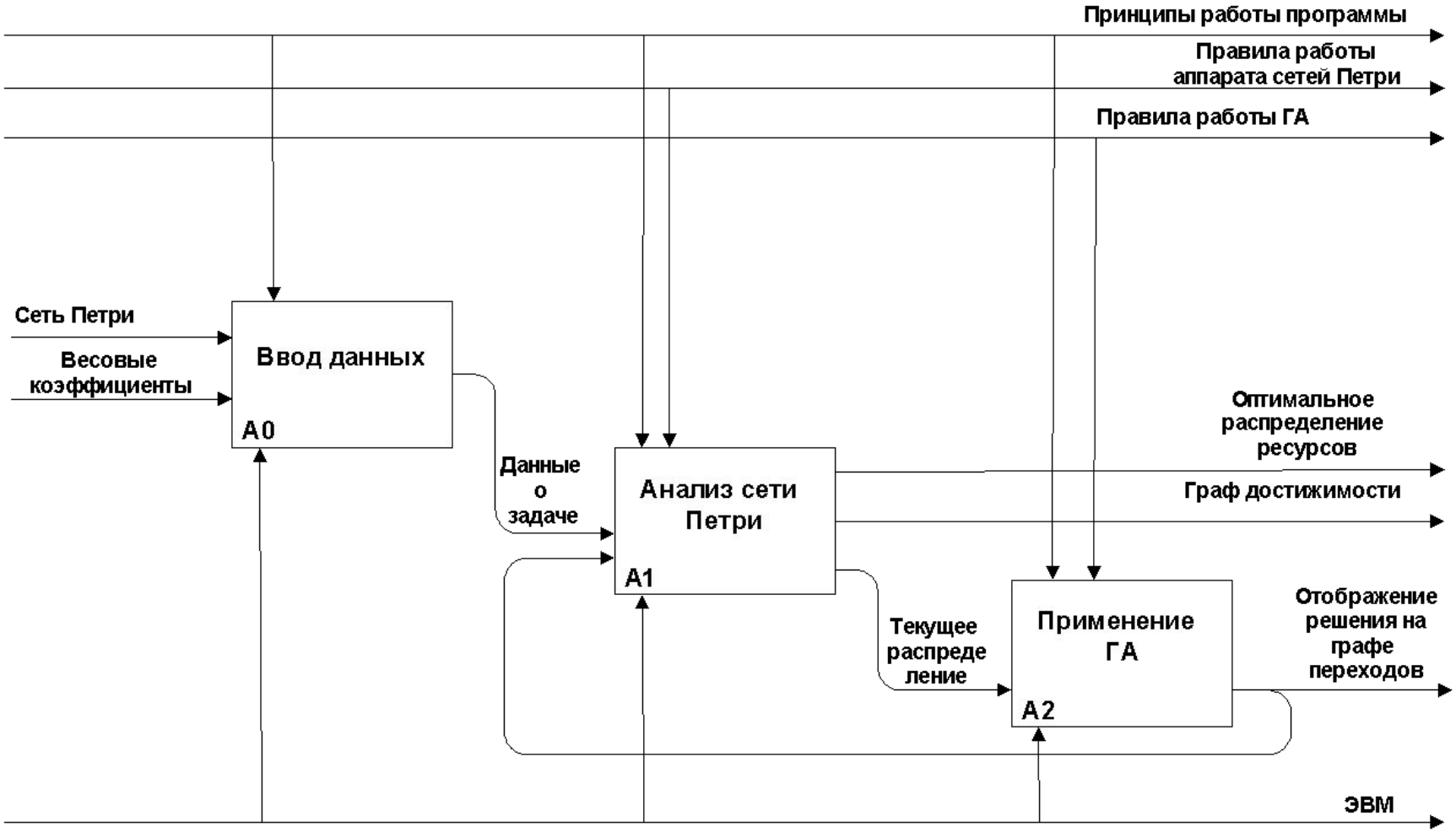
Уровень проектной спецификации. Этот уровень описывает способы реализации требований;

Уровень описания реализации. Здесь происходит преобразование спецификации в физическое описание конкретных программных и технических средств.

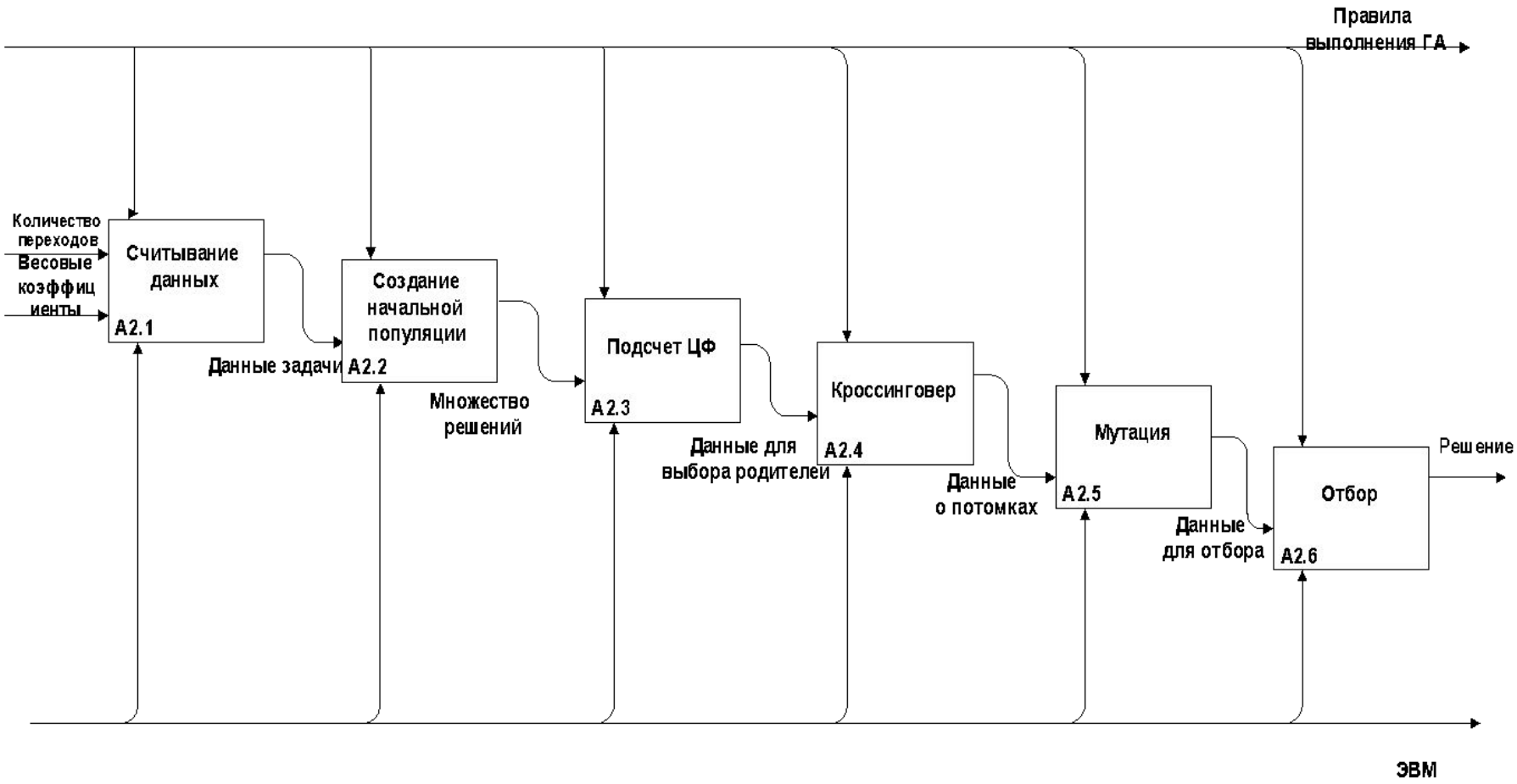
РАСПРЕДЕЛЕНИЕ РЕСУРСОВ



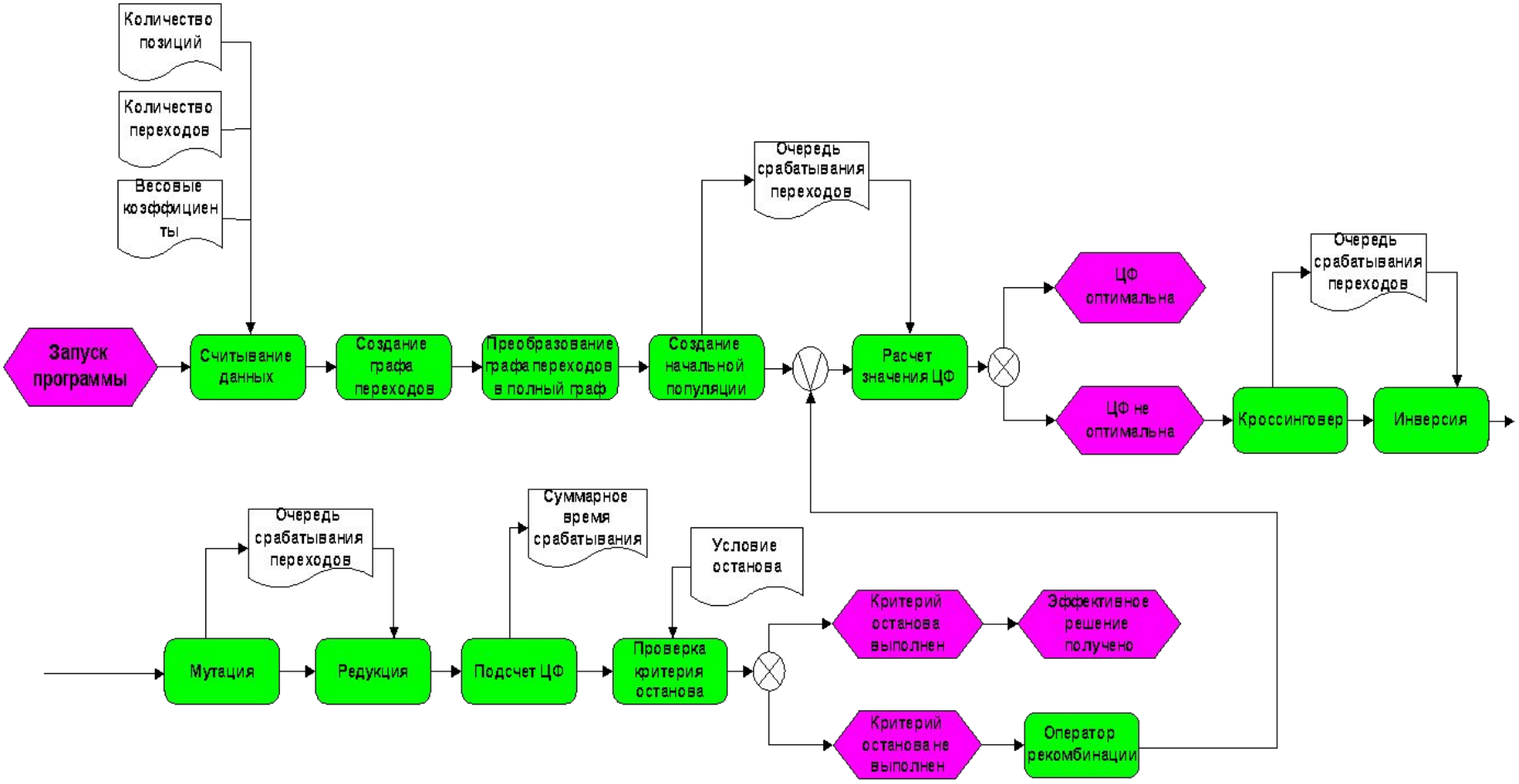
РАСПРЕДЕЛЕНИЕ РЕСУРСОВ (IDEF0)



РАСПРЕДЕЛЕНИЕ РЕСУРСОВ (IDEF0)



РАСПРЕДЕЛЕНИЕ РЕСУРСОВ (ARIS)



IDEF0 или ARIS

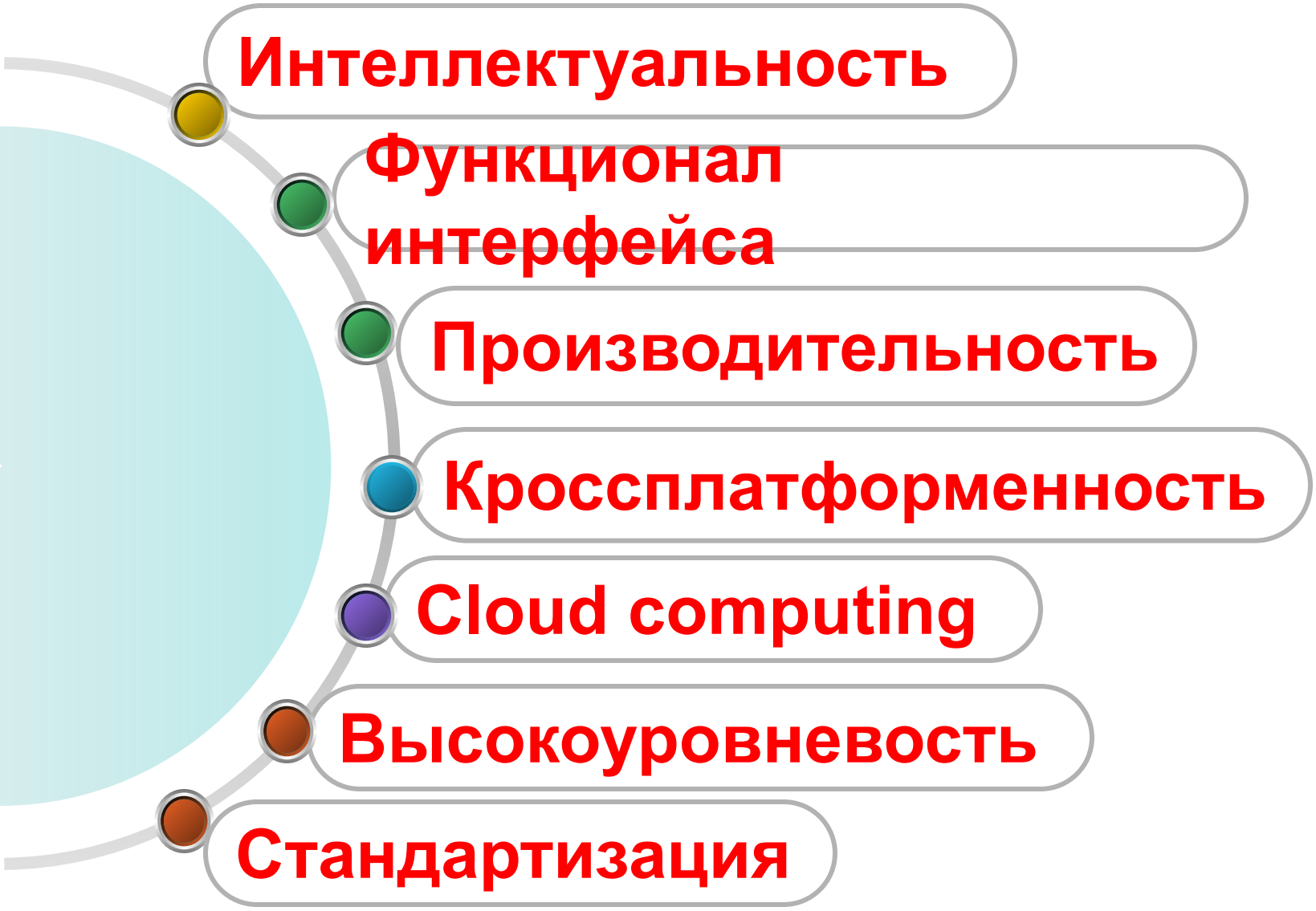
1) Четыре типа моделей позволяют ARIS более корректно провести классификацию входной информации;
2) В ARIS содержится большое количество стандартных готовых объектов для описания процессов;
3) ARIS предлагает больше возможностей для проведения экономического и функционального анализов.

IDEF0 или ARIS

Какой главный недостаток ARIS?

(отсутствие автоматической кодогенерации)

Тенденции развития ПО



Интеллектуальность

Прогнозирование последующих действий пользователя. Создание адаптирующихся интерфейсов.

Повышение конкурентоспособности ПО лежит в области повышения интеллектуальности продукта. Даже в простых программах где, казалось бы, некуда “приткнуть” интеллектуальность, можно предпринять ряд шагов, делающих программу более удобной в использовании.

Функционал интерфейса

Лет десять-пятнадцать назад удобство пользовательского интерфейса не было решающим фактором при выборе ПО. Ценилась больше функциональность. Это было связано с тем, что программы были не столь функциональны, инструментарий программиста был не такой мощный. В результате, программирование одной функции было огромной работой. Если ваше ПО имело на 2-3 функции больше, чем у конкурента, то у вас были большие шансы на успех. Сегодня практически любой функционал легко и быстро повторяется конкурентами. Получить длительное по времени конкурентное преимущество можно, внедрив более интеллектуальный функционал и, как ни удивительно, разработав хороший интерфейс пользователя.

Производительность

Ярким примером, иллюстрирующим тему данного абзаца, можно назвать гонку браузеров за производительностью их движков. Разработчики активно оптимизируют операционные системы, различное прикладное ПО.

У пользователя несколько процессоров, много памяти, графический ускоритель- так почему бы не задействовать это на “полную катушку”? В конечном итоге, быстрый, отзывчивый и удобный интерфейс очень понравится пользователю.

Кроссплатформеность

Обеспечить кроссплатформенность тому или иному алгоритму на сегодняшний день не сложно. Но вот интерфейсы… с ними загвоздка.

Как решить эту проблему?

Кроссплатформеность

Разработчики приноровились использовать web-интерфейсы даже в исключительно off-лайновых приложениях. Такой способ позволяет снять ряд проблем при создании кроссплатформенных интерфейсов.

Проблема!

Web-приложения, инсталлируемое ПО- у каждого типа приложений есть свои достоинства и недостатки. С одной стороны, web-приложения позволяют нам просто делать свое дело и не заботиться об установке ПО, о резервных копиях данных. С другой стороны, инсталлируемое ПО позволяет работать в off-лайновом режиме, задействовать все имеющиеся ресурсы на компьютере.

Как получить преимущества от обоих типов ПО?

Cloud computing

Предлагается расширить использование облаков, задействуя их в полноценном инсталлируемом ПО.

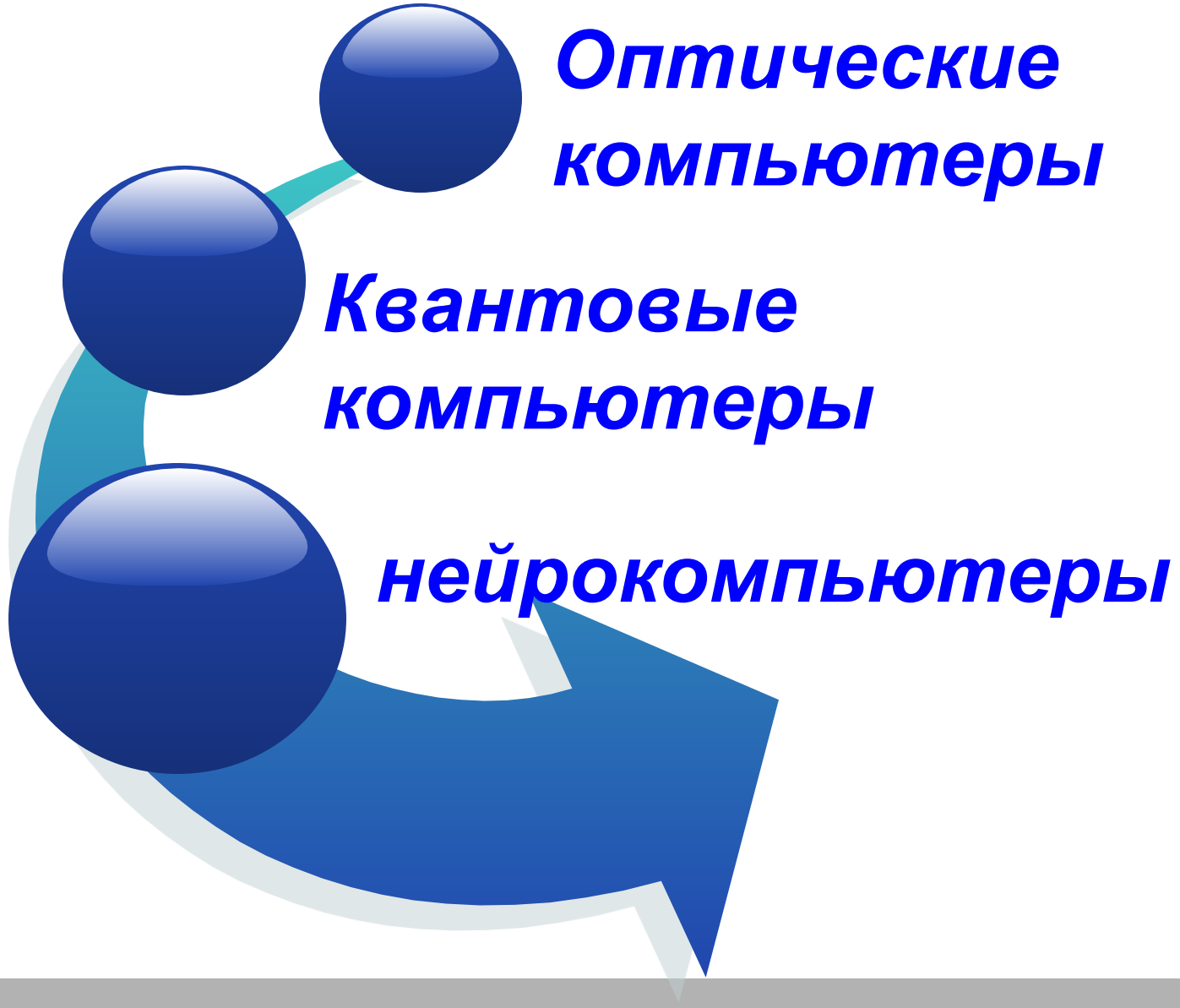
Высокоуровневость

Наделение коммерческого ПО элементами искусственного интеллекта- тема еще свежая, “мало раскопанная”, поэтому тут есть, где развернуться, есть интерес со стороны потребителей, есть деньги. Можно констатировать, что обладание своими низкоуровневыми технологиями на сегодняшний день - недостаток. Ведь на поддержку и развитие их надо тратить время и деньги. При этом, конкурентных преимуществ никаких. С другой стороны- высокоуровневое программирование с новыми перспективами, денежными рынками.

Стандартизация

Все больший приоритет приобретают стандарты при проектировании ПО. Формируются приемы программирования, нарабатываются методики ведения проектов. Хотя проблема быстрого устаревания знании еще актуальна, но она уже явно менее остра, чем 20 лет назад. ИТ стабилизируются в своем развитии, выходят на плоское плато S-образной кривой развития.

Тенденции развития вычислительной техники



Оптические компьютеры

- аналоговые интерференционные оптические вычисления
- использование оптических соединений для передачи сигналов
- создание компьютера, полностью состоящего из оптических устройств обработки информации

Три основных направления развития оптических методов в ВТ

Какое природное явление лежит в основе работы элементов оптического компьютера?

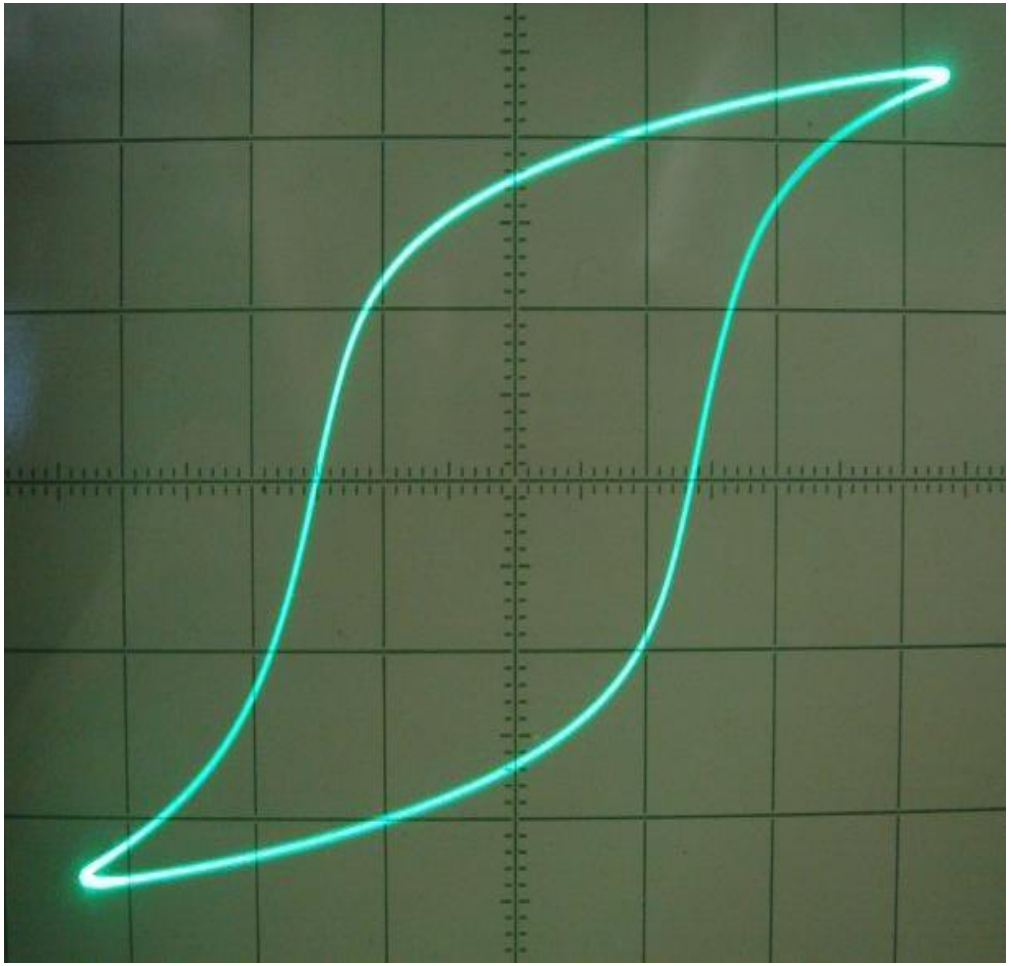
Оптические компьютеры

В основе работы различных компонентов оптического компьютера (трансфазаторы-оптические транзисторы, триггеры, ячейки памяти, носители информации) лежит явление оптической бистабильности. Оптическая бистабильность - это одно из проявлений взаимодействия света с веществом в нелинейных системах с обратной связью, при котором определенной интенсивности и поляризации падающего на вещество излучения соответствуют **два** (аналог 0 и 1 в полупроводниковых системах) **возможных стационарных состояния световой волны**, прошедшей через вещество, отличающихся амплитудой и (или) параметрами поляризации.

Принцип работы

Увеличение интенсивности падающего на вещество светового луча до некоторого значения I_1 приводит к резкому возрастанию интенсивности прошедшего луча; на обратном же ходе при уменьшении интенсивности падающего луча до некоторого значения $I_2<I_1$ интенсивность прошедшего луча остается постоянной, а затем резко падает. Таким образом, интенсивности падающего пучка I, значение которой находится в пределах петли гистерезиса, соответствуют два значения интенсивности прошедшего пучка, зависящих от предыдущего оптического состояния поглощающего вещества.

Гистерезис



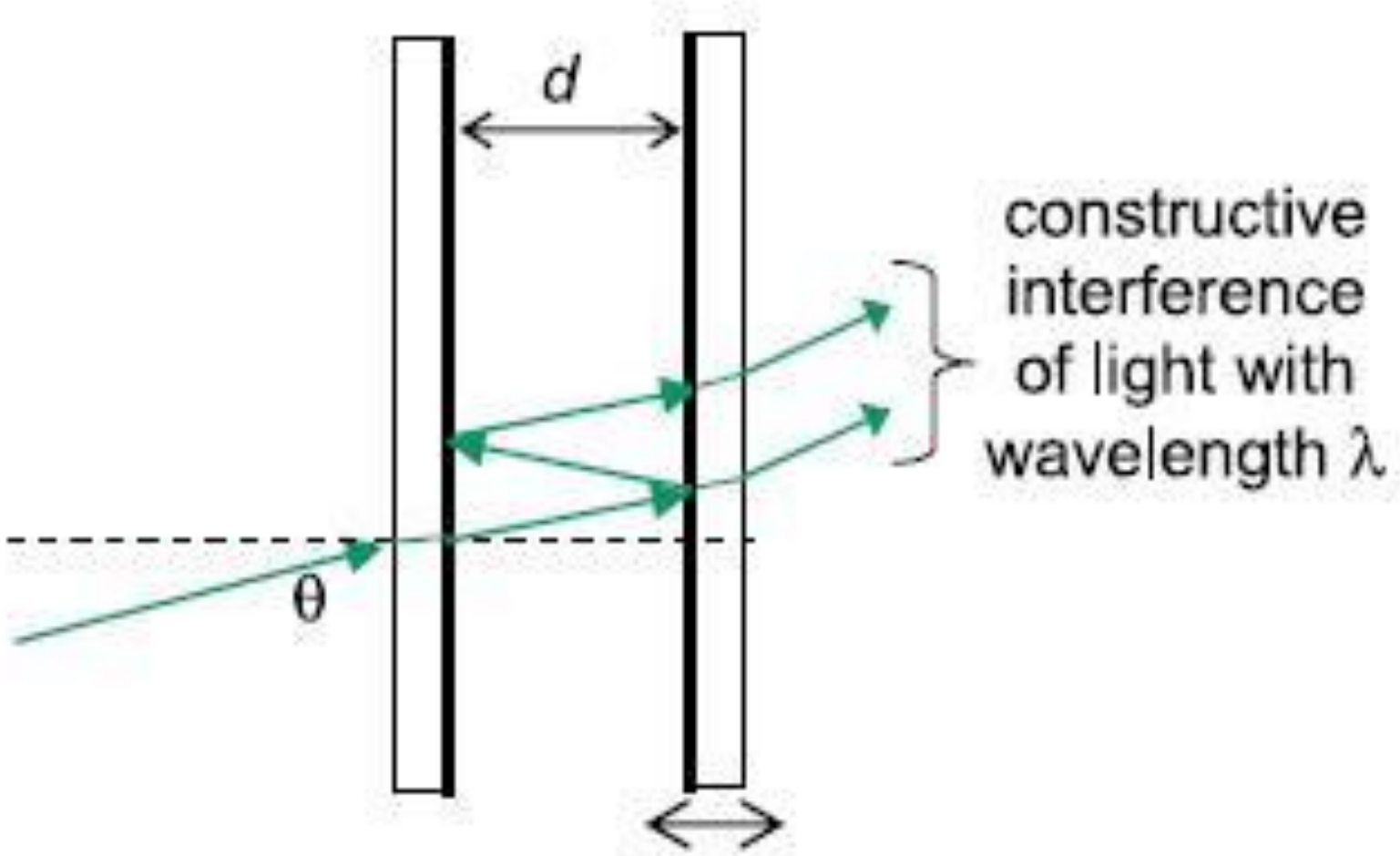
Гистерезис – это свойство биологических, физических и прочих систем, в которых мгновенный отклик на воздействия зависит от их текущего состояния, а на интервале времени поведение системы определяется ее предысторией. Петлей гистерезиса называется график, демонстрирующий это свойство.

Каким образом реализуется оптическое логическое устройство?

Оптические логические устройства

Весь набор полностью оптических логических устройств для синтеза более сложных блоков оптических компьютеров реализуется на основе пассивных **нелинейных резонаторов-интерферометров**. В зависимости от начальных условий (начального положения пика пропускания и начальной интенсивности оптического излучения) в пассивном нелинейном резонаторе, **нелинейный процесс завершается установлением одного из двух устойчивых состояний пропускания падающего излучения**. А из нескольких нелинейных резонаторов можно собрать любой, более сложный логический элемент (триггер).

Нелинейные резонаторы



Какой минимальный размер у оптического элемента памяти?

Оптические элементы памяти

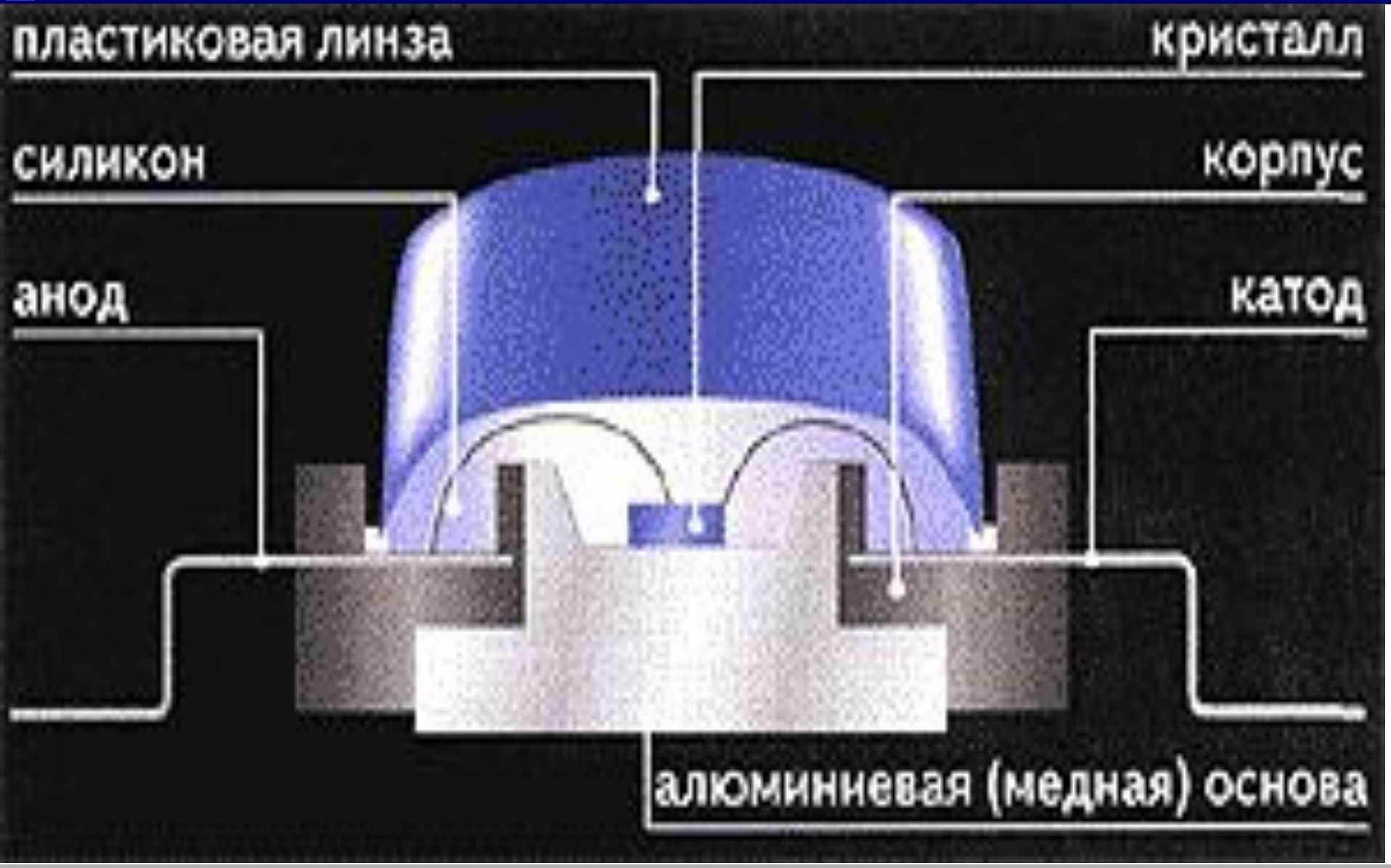
Элементы памяти оптического компьютера представляют собой полупроводниковые нелинейные оптические интерферометры, в основном, созданными из арсенида галлия (GaAs). Минимальный размер оптического элемента памяти определяется минимально необходимым числом атомов, для которого устойчиво наблюдается оптическая бистабильность. Это число составляет ~1000 атомов, что соответствует 1-10 нанометрам.

Как устроен светодиод?

light emitting diode

Светодиод состоит из полупроводникового кристалла на токонепроводящей подложке, корпуса с контактными выводами и оптической системой. Для повышения жизнестойкости пространство между кристаллом и пластиковой линзой заполнено прозрачным силиконом. Алюминиевая основа служит для отвода избыточного тепла. Которого, надо сказать, выделяется совсем небольшое количество.

light emitting diode



Какое физическое явление приводит к свечению в полупроводниковом кристалле светодиода?

light emitting diode

Свечение в полупроводниковом кристалле возникает при рекомбинации электронов и дырок в области p-n-перехода. Область p-n-перехода, образуется контактом двух полупроводников с разными типами проводимости.

Очевидно, что чем больший ток проходит через светодиод, тем он светит ярче, поскольку чем больше ток, тем больше электронов и дырок поступают в зону рекомбинации в единицу времени.

light emitting diode

Чтобы p-n-переход стал излучать свет, ширина зоны в активной области светодиода должна быть близка к энергии квантов света видимого диапазона. Во-вторых, полупроводниковый кристалл должен содержать мало дефектов, из-за которых рекомбинация происходит без излучения. Чтобы соблюсти оба условия, зачастую одного p-n-перехода в кристалле оказывается недостаточно, и производители вынуждены идти на изготовление многослойных полупроводниковых структур, так называемых гетероструктур.

Какие цвета свечения свойственны для p-n-перехода?

Как получить белый цвет свечения светодиода?

Проблема цвета

Белый свет от светодиодов можно получить несколькими способами. **Первый** — смешать цвета по технологии RGB. На одной матрице плотно размещаются красные, голубые и зеленые светодиоды, излучение которых смешивается при помощи оптической системы, например линзы. В результате получается белый свет. **Второй** способ заключается в том, что на поверхность светодиода, излучающего в ультрафиолетовом диапазоне (есть и такие), наносится три люминофора, излучающих, соответственно, голубой, зеленый и красный свет. По принципу люминесцентной лампы. **Третий** способ - это когда желто-зеленый или зелено-красный люминофор наносятся на голубой светодиод. При этом два или три излучения смешиваются, образуя белый или близкий к белому свет.

Где в IT- сфере уже активно используются оптические технологии?

Последние разработки

Суть последних разработок: **избавиться от преобразования светового сигнала (фотонов) в электрический (электронов)** на всем протяжении прохождения по цепи. Если раньше процессы протекающие в чипе можно было описать как фотон-электрон-фотон, то сейчас, благодаря «диоду для света» встроенному в чип — фотон-фотон. Оптический компьютер становится реальностью.

Последние разработки

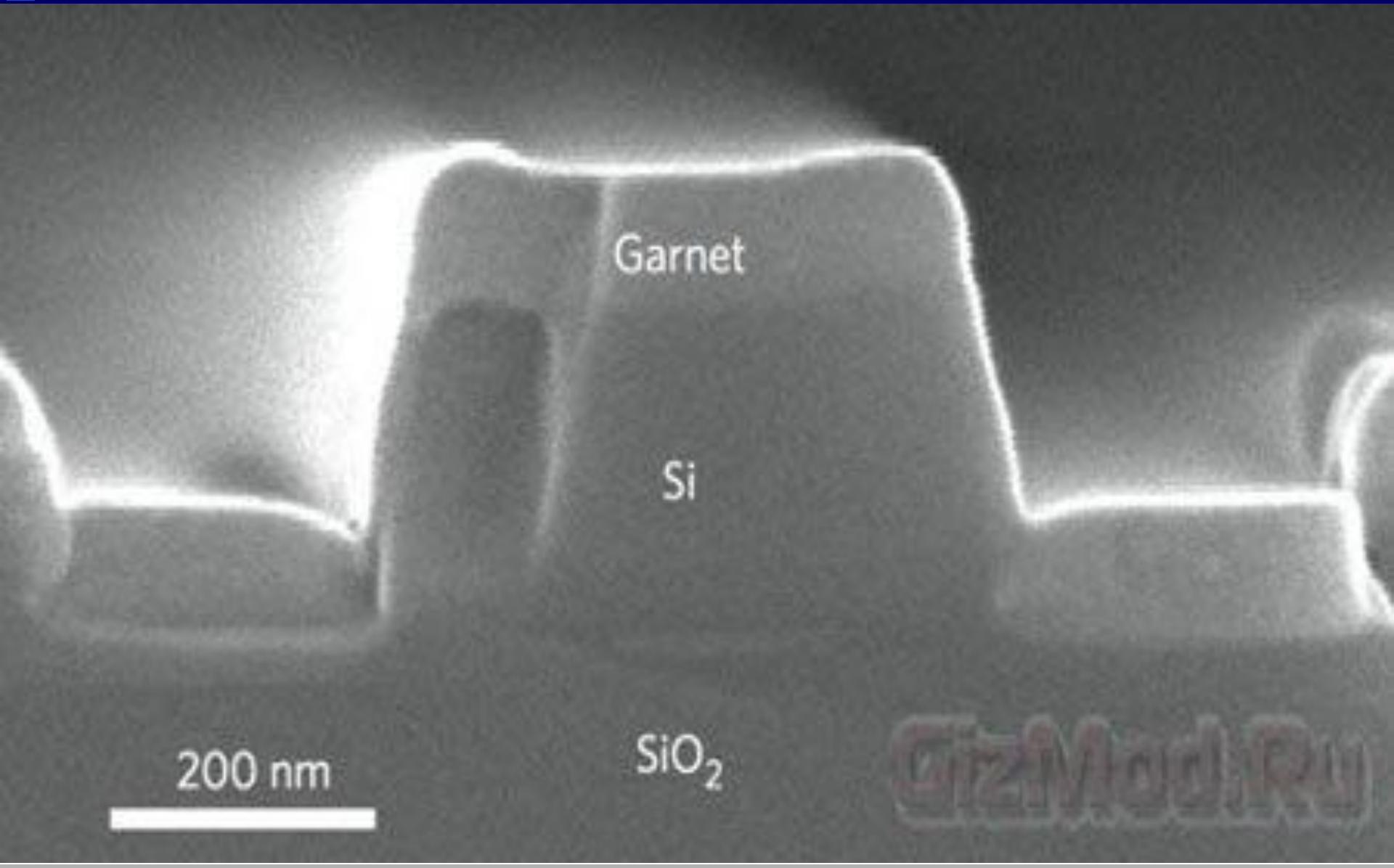
Это значит, что по волоконно-оптической сети к компьютеру может подводиться сразу **несколько пучков информации** без преобразования и замедления сигнала. Скорость света превосходит скорость компьютера на электронах. По медным проводам может проходить **только один электронный поток данных**.

«Диод для света»

Он работает так же, как его электрический аналог, пропуская лишь в одном направлении, только **не ток, а свет**.

Ключевым для разработки является выбор материала, обладающего необходимыми свойствами. На его роль подходит **гранат**, пленка из которого на поверхности кремниевого чипа по-разному преломляет свет, падающий на нее под разными углами, в результате пропуская свет только в одном направлении. Основная заслуга специалистов заключается в том, что они нашли способ формирования этой пленки с применением стандартного оборудования для выпуска кремниевых чипов.

«Диод для света»



Проблемы создания ОК

К настоящему времени уже созданы и оптимизированы отдельные составляющие оптических компьютеров – оптические процессоры, ячейки памяти, однако до полной сборки еще далеко. Основной проблемой, стоящей перед учеными, является синхронизация работы отдельных элементов оптического компьютера в единой системе, поскольку уже существующие элементы характеризуются различными параметрами рабочей волны светового излучения (интенсивность, длина волны), и уменьшение его размера. Если для конструирования оптического компьютера использовать уже разработанные компоненты, то обычный PC имел бы размеры легкового автомобиля.

Каковы потенциальные преимущества оптических компьютеров?

Потенциальные преимущества ОК

- световые потоки, в отличие от электрических, могут пересекаться друг с другом
- световые потоки могут быть локализованы в поперечном направлении до нанометровых размеров и передаваться по свободному пространству
- скорость распространения светового сигнала выше скорости электрического
- взаимодействие световых потоков с нелинейными средами распределено по всей среде, что дает новые степени свободы в организации связи и создании параллельных архитектур

Потенциальные преимущества ОК

Создание большего количества параллельных архитектур, по сравнению с полупроводниковыми компьютерами, является основным достоинством оптических компьютеров, оно позволяет преодолеть ограничения по быстродействию и параллельной обработке информации, свойственные современным ЭВМ. Развитие оптических технологий все равно будет продолжаться, поскольку полученные результаты важны не только для создания оптических компьютеров, но также и для оптических коммуникаций и сети Internet.

Квантовые компьютеры

Идея о квантовых вычислениях была высказана Юрием Маниным в 1980 году, одна из первых моделей квантового компьютера была предложена Ричардом Фейнманом в 1981 году.

Необходимость в квантовом компьютере возникает тогда, когда мы пытаемся исследовать методами физики сложные многочастичные системы, подобные биологическим. Пространство квантовых состояний таких систем растет как экспонента от числа ***n*** составляющих их реальных частиц, что делает невозможным моделирование их поведения на классических компьютерах уже для ***n=10***.

В чем заключается физический принцип «суперпозиции»?

Квантовые компьютеры

Бит имеет лишь два состояния - 0 и 1, тогда как состояний кубита (qubit, Quantum Bit) значительно больше. Существуют волновые функции, которые называются собственными для какой-либо определенной величины. Квантовая система может находиться в состоянии с волновой функцией, равной линейной комбинации собственных функций, соответствующих каждому из возможных значений (такое состояние называется сложным), т. е. физически - ни в возбужденном, ни в основном состоянии (суперпозиция). Это означает, что кубит в одну единицу времени равен и 0, и 1, тогда как классический бит в ту же единицу времени равен либо 0, либо 1. Как для классических, так и для квантовых компьютеров были введены элементарные логические операции: дизъюнкция, конъюнкция и квантовое отрицание, при помощи которых будет организована вся логика квантового компьютера.

Квантовые компьютеры

Квантовую суперпозицию можно проиллюстрировать, например, так: «Вообразите атом, который мог бы подвергнуться радиоактивному распаду в определённый промежуток времени. Или не подвергнуться. Мы можем ожидать, что у этого атома есть только два возможных состояния: „распад“ и „не распад“, но в квантовой механике у атома может быть некое объединённое состояние — „распада — не распада“, то есть ни то, ни другое, а как бы между. Вот это состояние и называется „**суперпозицией**“»

Квантовые компьютеры

Согласно законам квантовой механики, энергия электрона, связанного в атоме, не произвольна. Она может иметь лишь определенный прерывный (дискретный) ряд значений E_0, E_1,... E_n называемых уровнями энергии (**спектр атома**). Самый нижний уровень энергии E_0, при котором энергия наименьшая, называется основным. Остальные уровни (E_1, E_2,... En) соответствуют более высокой энергии и называются возбужденными. Излучение и поглощение электромагнитной энергии происходит отдельными порциями - квантами, или фотонами. При поглощении фотона энергия увеличивается - он переходит «вверх» - с нижнего на верхний уровень, при излучении фотона атом совершает переход вниз.

Квантовые компьютеры

Если атом в данный момент времени находится в одном из возбужденных состояний E_2, то такое состояние атома неустойчиво, даже если на него не влияют другие частицы. Через очень короткое время атом перейдет в одно из состояний с меньшей энергией, например E_1. Такой самопроизвольный (спонтанный) переход с одного уровня на другой и сопровождающее его спонтанное излучение столь же случайны во времени, как радиоактивный распад ядра атома.

Предсказать точно момент перехода принципиально невозможно - можно лишь говорить о вероятности того, что переход произойдет через такое-то время.

Квантовые компьютеры

Но атом может перейти с уровня E_2 на E_1 не спонтанно, а под действием электромагнитной волны, если только частота этой волны достаточно близка к частоте перехода атома. Такая резонансная волна как бы «расшатывает» электрон и ускоряет его «падение» на уровень с меньшей энергией. Переходы, происходящие под действием внешнего электромагнитного поля, называются вынужденными (или стимулированными).

При создании квантового компьютера основное внимание уделяется вопросам управления кубитами при помощи вынужденного излучения и недопущении спонтанного излучения, которое нарушит работу всей квантовой системы.

Как избежать спонтанного излучения в квантовых компьютерах?

Для нормального функционирования квантовым компьютерам нужен холод

Абсолютный нуль, или 0 К, — это самая низкая температура, которую может иметь физическое тело. При достижении абсолютного нуля атомы прекращают колебания и перестают выделять тепло — такие условия для квантовых вычислений близки к идеальным. Например, внутри квантового компьютера, разработанного компанией D-Wave Systems, поддерживается температура 0,02 К, или -273,13°С.

Квантовые компьютеры

Идея квантовых вычислений состоит в том, что квантовая система из ***L*** двухуровневых квантовых элементов (квантовых битов, кубитов) имеет $\mathbf{2^L}$ линейно независимых состояний, а значит, вследствие принципа квантовой суперпозиции, пространство состояний такого квантового регистра является $\mathbf{2^L}$-мерным гильбертовым пространством. Операция в квантовых вычислениях соответствует повороту вектора состояния регистра в этом пространстве. Таким образом, квантовое вычислительное устройство размером *L* кубит фактически задействует одновременно 2^L классических состояний.

Физическими системами, реализующими кубиты, могут быть любые объекты, имеющие два квантовых состояния: поляризационные состояния фотонов, электронные состояния изолированных атомов или ионов, спиновые состояния ядер атомов, и т. д.

Квантовые компьютеры

Квантовый бит, называемый кубитом, находится в состоянии a(0) + b(1), так что $|a|^2$ и $|b|^2$ — вероятности получить ***0*** или ***1*** соответственно при измерении этого состояния; $|a|^2 + |b|^2 = 1$. Сразу после измерения кубит переходит в базовое квантовое состояние, соответствующее классическому результату.

Пример:

Имеется кубит в квантовом состоянии 4/5 (0) – 3/5 (1). В этом случае, вероятность получить при измерении ***0*** составляет $(4/5)^2=16/25 = 64$ %, ***1*** $(-3/5)^2=9/25 = 36$ %.

В результате измерения кубит переходит в новое квантовое состояние , то есть, при следующем измерении этого кубита мы получим ***0*** со стопроцентной вероятностью.

Квантовые компьютеры

Перейдем к системе из двух кубитов. Измерение каждого из них может дать ***0*** или ***1***. Поэтому у системы есть 4 классических состояния: ***00, 01, 10 и 11***. Общее квантовое состояние системы имеет вид ***a|00| + b|01| + c|10| + d|11|***. Теперь $|a|^2$ — вероятность измерить 00 и т. д. Отметим, что $|a|^2+|b|^2+|c|^2+|d|^2=1$ как полная вероятность.

В общем случае системы из ***L*** кубитов, у неё 2^L классических состояний (00000…(L-нулей), …00001(L-цифр), … , 11111…(L-единиц)), каждое из которых может быть измерено с вероятностями 0—100 %.

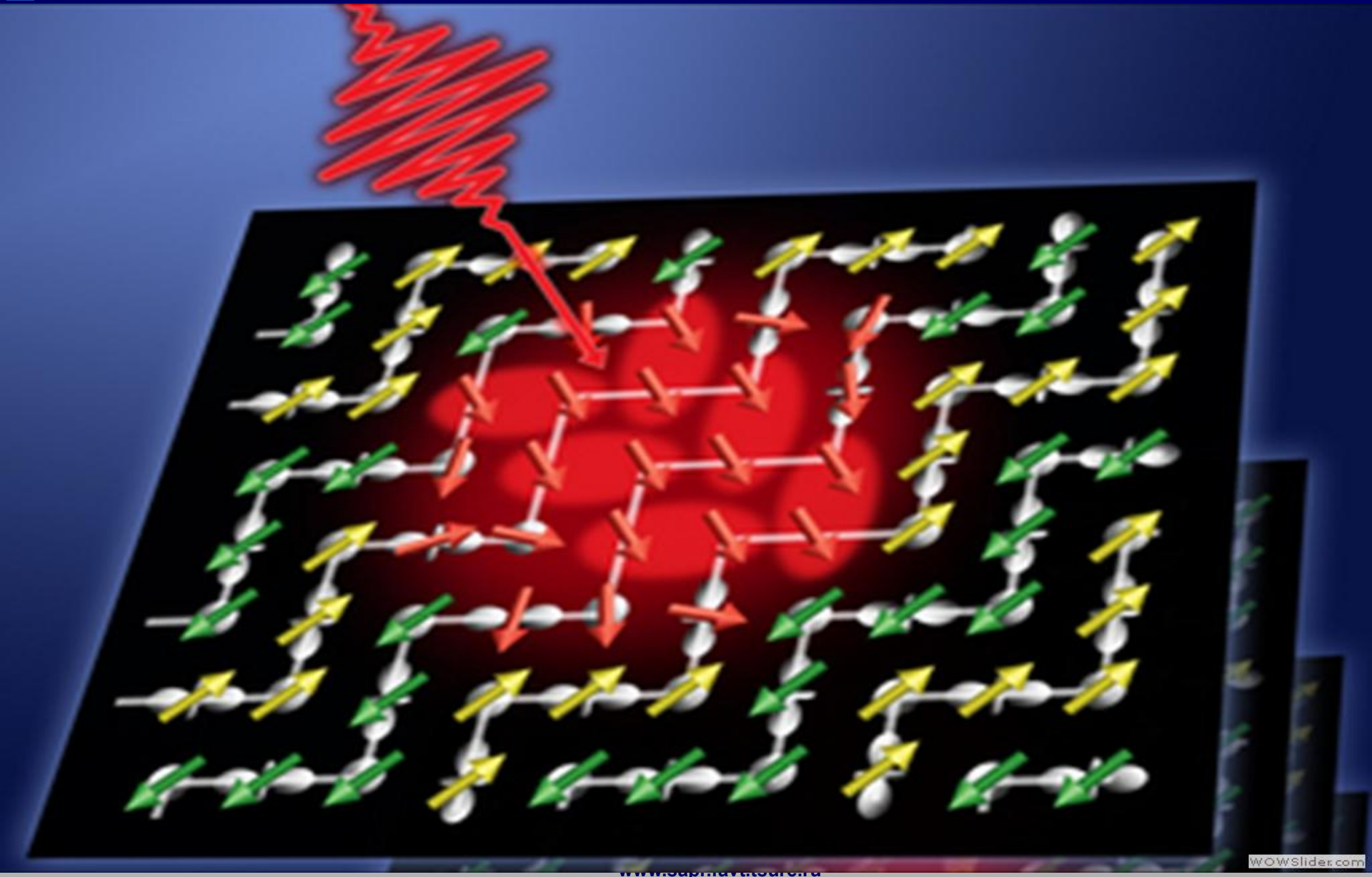
Одна операция над группой кубитов затрагивает все значения, которые она может принимать, в отличие от классического бита. Это и обеспечивает беспрецедентный параллелизм вычислений.

Квантовые компьютеры

Большая часть современных ЭВМ работают по схеме: ***n*** бит памяти хранят состояние и каждый такт времени изменяются процессором. В квантовом случае система из ***n*** кубитов находится в состоянии, являющимся суперпозицией всех базовых состояний, поэтому изменение системы касается *всех* 2^n базовых состояний одновременно. Теоретически новая схема может работать намного (в экспоненциальное число раз) быстрее классической.

Практически (квантовый) алгоритм Гровера поиска в базе данных показывает квадратичный прирост мощности против классических алгоритмов.

Квантовые компьютеры



Квантовые алгоритмы

Алгоритм Гровера позволяет найти решение уравнения $f(x) = 1,\ 0 \le x < N$ за время $O(\sqrt{N})$.

Алгоритм Шора позволяет разложить натуральное число ***n*** на простые множители за полиномиальное от ***log(n)*** время.

Алгоритм Залки — Визнера позволяет моделировать
унитарную эволюцию квантовой системы частиц за почти линейное время с использованием ***O(n)*** кубит.

Алгоритм Дойча — Йожи позволяет «за одно вычисление» определить, является ли функция двоичной переменной *f(n)* постоянной ($f_1(n)$ = 0, $f_2(n)$ = 1 независимо от *n* или«сбалансированной» ($f_3(0)$ = 0, $f_3(1)$ = 1; $f_4(0)$ = 1, $f_4(1)$ = 0).

**Алгоритм Саймона решает проблему чёрного ящика экспоненциально быстрее, чем любой классический алгоритм,
включая вероятностные алгоритмы.**

Квантовые компьютеры (успешные применения)

Благодаря огромной скорости разложения на простые множители, квантовый компьютер позволит расшифровывать сообщения, зашифрованные асимметричным криптографическим алгоритмом RSA (аббревиатура от фамилий Rivest, Shamir и Adleman) — криптографический алгоритм с открытым ключом, основывающийся на вычислительной сложности задачи факторизации больших целых чисел. До сих пор этот алгоритм считается сравнительно надёжным, так как эффективный способ разложения чисел на простые множители для классического компьютера в настоящее время неизвестен. **Для того, например, чтобы получить доступ к кредитной карте, нужно разложить на два простых множителя число длиной в сотни цифр. Даже для самых быстрых современных компьютеров выполнение этой задачи заняло бы в сотни раз больше времени, чем возраст Вселенной**. Благодаря ***алгоритму Шора*** эта задача становится вполне осуществимой, если квантовый компьютер будет построен.

Первые реализации квантовых компьютеров

В конце 2001 года IBM заявила об успешном тестировании 7-кубитного квантового компьютера, реализованного с помощью ЯМР.
На нём был исполнен алгоритм Шора и были найдены сомножители числа 15.

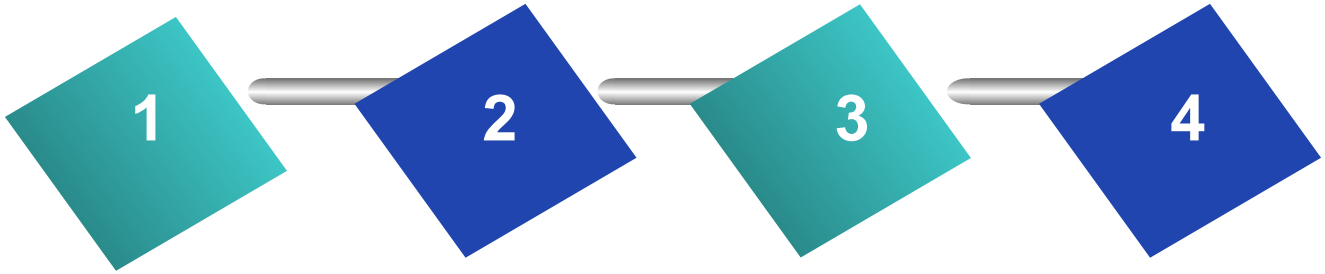
В 2005 году группой Ю. Пашкина (сотрудник лаборатории сверхпроводимости г. Москвы) при помощи японских специалистов был построен двухкубитный квантовый процессор на сверхпроводящих элементах.

В ноябре 2009 года физикам из Национального института стандартов и технологий в США впервые удалось собрать программируемый квантовый компьютер, состоящий из двух кубит.

В феврале 2012 года компания IBM сообщила о достижении значительного прогресса в физической реализации квантовых вычислений с использованием
сверхпроводящих кубитов, которые, по мнению компании, позволят начать работы по созданию квантового компьютера.

В апреле 2012 года группе исследователей из Южно-Калифорнийского университета, Технологического университета Дельфта, университета штата Айова, и Калифорнийского университета удалось построить двухкубитный квантовый компьютер на кристалле алмаза с примесями.
На нем **реализован алгоритм Гровера для 4-х вариантов перебора** (95%)

Принципы практической реализации квантовых компьютеров (1996 D.P. Divincenzo)



Точно известное число частиц системы.	Возможность приведения системы в точно известное начальное состояние.	Высокая степень изоляции от внешней среды.	Умение менять состояние системы согласно заданной последовательности элементарных преобразований.

Квантовые компьютеры (успешные применения)

Недавно немецким физикам из Института Макса Планки, Германия, удалось **записать в атом рубидия информацию о квантовом состоянии фотона**. А после считать ее обратно. Таким образом, физики создали еще один кирпичик, приблизившись к созданию квантового компьютера. Физики считают, что если число кубитов превысит значение 100, то тогда будет положено начало квантовой эры.

Почему носителем квантовой информации выбрали именно фотон?

Квантовые компьютеры (успешные применения)

Ученные предполагают, что обмен информацией между квантовыми компьютерами будет происходить посредством элементарных частиц, где информация будет зашифрована в их квантовых состояниях. Из всех частиц **ученые выбирают фотон как наиболее подходящий вариант. Поскольку перенос фотона не нуждается в передачи на расстояние материи.**

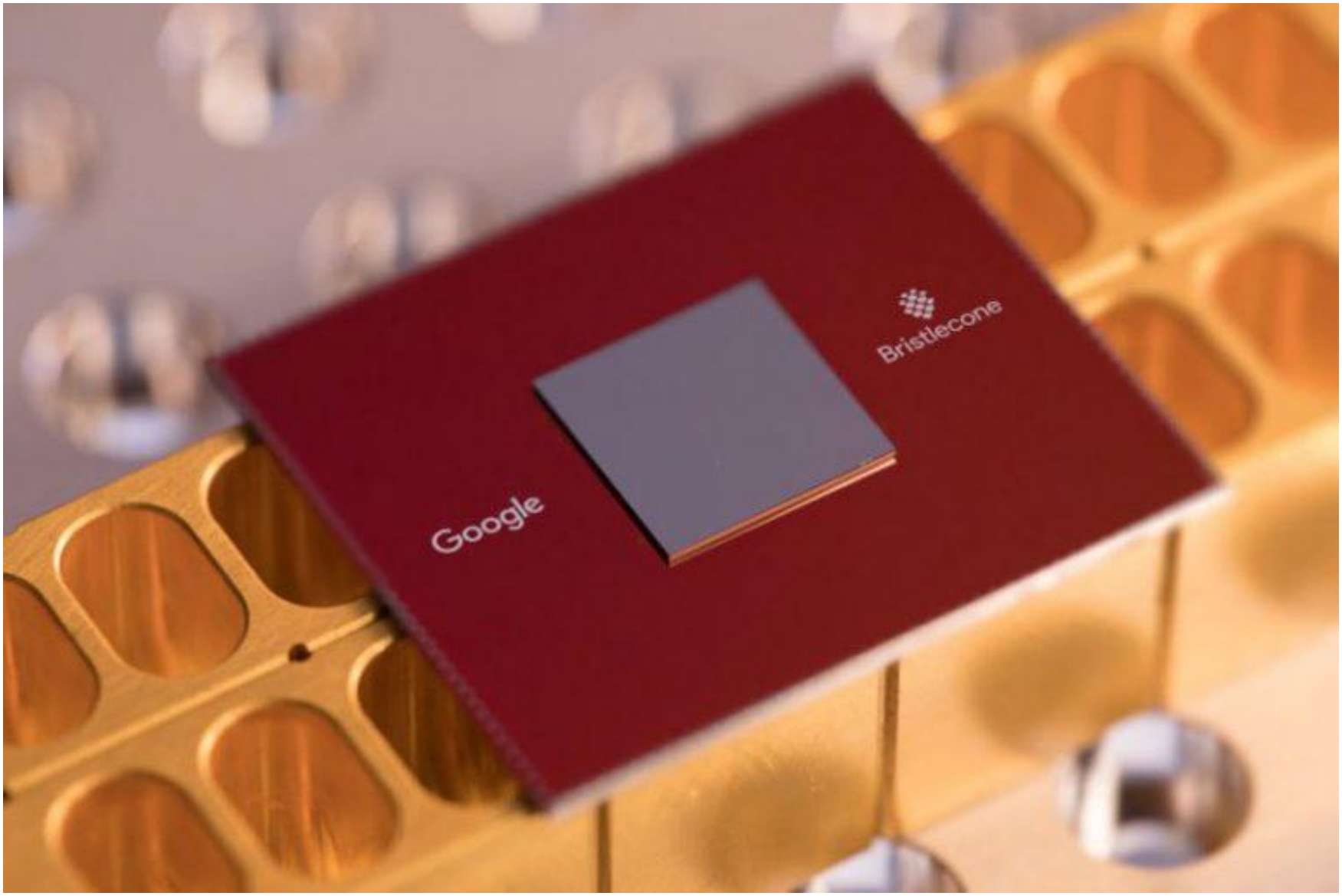
Квантовые компьютеры (успешные применения)

Ученные полагают, что благодаря хранению информации в одном атоме открываются перспективы к **миниатюризации**. Соответственно, данные будут считываться путем прямых манипуляций с атомами. Так же будет интересно узнать, была ли записана информация с фотона на атом при сохранении квантового состояния этого фотона. Соответственно, если информация не была записана, то запись можно будет повторить снова.

Квантовые компьютеры (успешные применения)

30-кубитный квантовый компьютер по мощности равен суперкомпьютеру, работающему с производительностью **10 терафлопс** (триллион операций в секунду). Мощность современных настольных компьютеров измеряется всего лишь гигафлопсах (миллиард операций в секунду).

Google представила новый квантовый процессор



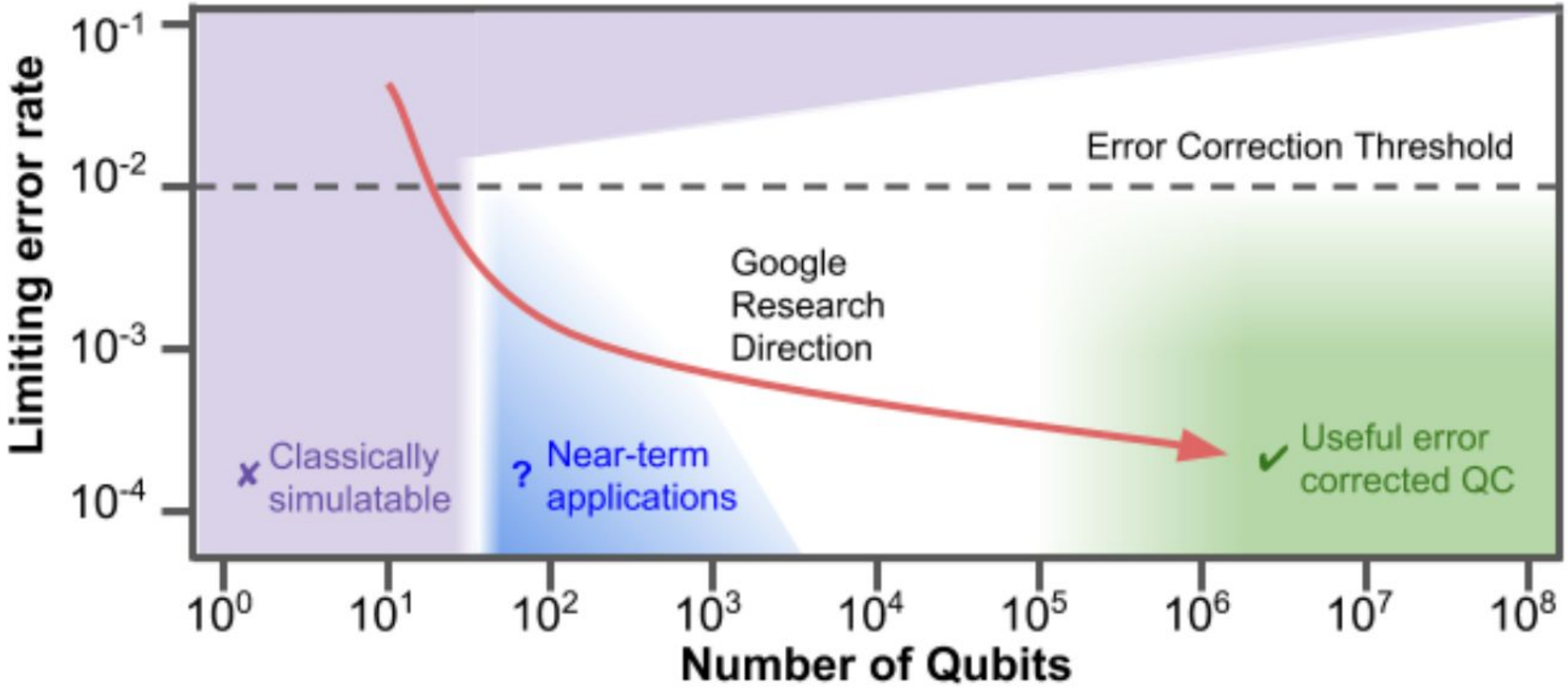
Google представила новый квантовый процессор

Корпорация Google представила 72-кубитный квантовый процессор Bristlecone. С помощью этого процессора подразделение Google Quantum AI lab, ответственное за разработку квантового компьютера, будет тестировать системные ошибки и масштабируемость технологии, а также области применения квантовой симуляции, оптимизации и машинного обучения «для решения проблем реального мира».

Google представила новый квантовый процессор

Новый 72-кубитный квантовый процессор Google Bristlecone построен по принципу, который позволил в предыдущем 9-кубитном процессоре показать низкую частоту ошибок при считывании данных (1%), при работе однокубитного вентиля — 0,1% и при работе двухкубитного вентиля — 0,6%, что, как отмечает Google, было лучшим результатом компании. Перед применением нового процессора в работе важно понять его возможности: команда создала инструмент, проверяющий его на ошибки, с помощью решения идентичных задач на квантовом процессоре и в классической симуляции. При низком количестве ошибок может быть достигнуто «квантовое превосходство».

Зависимость количества ошибок от количества кубитов в процессоре



Google представила новый квантовый процессор

Квантовые компьютеры используют **квантовую суперпозицию** и **квантовую запутанность** для передачи и обработки данных. Одной из главных задач квантовых компьютеров станет усиление искусственного интеллекта. Кубиты квантового процессора — это квантовые аналоги битов. Два расположенных рядом кубита имеют четыре состояния — оба вкл, оба выкл, вкл/выкл и выкл/вкл, каждый из них имеет вес или «амплитуду», которая способна играть роль нейрона; третий кубит в такой системе позволяет представить восемь нейронов, а четвёртый — шестнадцать. Изменение состояния четырёх кубитов приводит к обработке шестнадцати нейронов за один раз, в то время как классический компьютер обрабатывал бы эти числа по одному.

Квантовая запутанность

Понятие ***«квантовая запутанность»*** *появилось из теоретического предположения, вытекающего из уравнений квантовой механики. Оно означает вот что:* ***если 2 квантовые частицы (ими могут быть электроны, фотоны) оказываются взаимозависимыми (запутанными), то связь сохраняется, даже если их разнести в разные части Вселенной***

Квантовая запутанность

Если получить пару фотонов одновременно, то они окажутся **связанными (запутанными)**. А если замерить спин одного из них и он окажется **положительным**, то спин 2-го фотона – будьте уверены – мгновенно станет **отрицательным**. И, наоборот.

Спин квантовой частицы

Если коротко, то ***спином*** *квантовой частицы (электрона, фотона) называется её собственный угловой момент.* ***Спин можно представить в виде вектора, а саму квантовую частицу – в виде микроскопического магнитика.***

Суперпозиция

Важно понять, что когда за квантом, например, электроном никто не наблюдает, то он имеет все значения спина одновременно. ***Это фундаментальное понятие квантовой механики называется «суперпозицией».***

Как измерить спин квантовой частицы

Частицу (электрон) помещают в магнитное поле: электроны со спином против направления поля, и со спином по направлению поля отклонятся в разные стороны. Спины фотонов измеряют, направляя в поляризационный фильтр. Если спин (или поляризация) фотона «-1», то он не проходит через фильтр, а если «+1», то проходит.

Резюме

Как только Вы измерили состояние одного электрона и определили, что его спин «+1», то связанный или «запутанный» с ним электрон принимает значение спина «-1». Причём моментально, даже если он находится на Марсе. Хотя до измерения состояния 2-го электрона, он имел оба значения спина одновременно («+1» и «-1»).

Спор Энштейна с Бором

Энштейн называл «квантовую запутанность» SPUCKHAFTE FERWIRKLUNG (нем.) или ***пугающим, призрачным, сверхъестественным действием на расстоянии****.*
Энштейн не соглашался с интерпретацией Бора о квантовой запутанности частиц. Потому что это **противоречило его теории, что информация не может передаваться со скоростью больше скорости света.** В 1935 году он опубликовал статью с описанием мысленного эксперимента. Этот эксперимент назвали «Парадоксом Эйнштейна — Подольского — Розена».
Там он утверждал, что в запутанные квантовые частицы изначально заложена информация об их состояниях.

Подтверждение квантовой теории

Джон Клаузер, будучи ещё аспирантом Колумбийского университета, в 1967 отыскал забытую работу ирландского физика Джона Белла. Это была сенсация: оказывается **Беллу удалось вывести из тупика спор Бора и Энштейна**. Он предложил экспериментально проверить обе гипотезы. Для этого он предложил построить машину, которая бы создавала и сравнивала много пар запутанных частиц. Джон Клаузер принялся разрабатывать такую машину. Его машина могла создавать тысячи пар запутанных частиц и сравнивать их по разным параметрам. **Результаты экспериментов доказывали правоту Бора.**

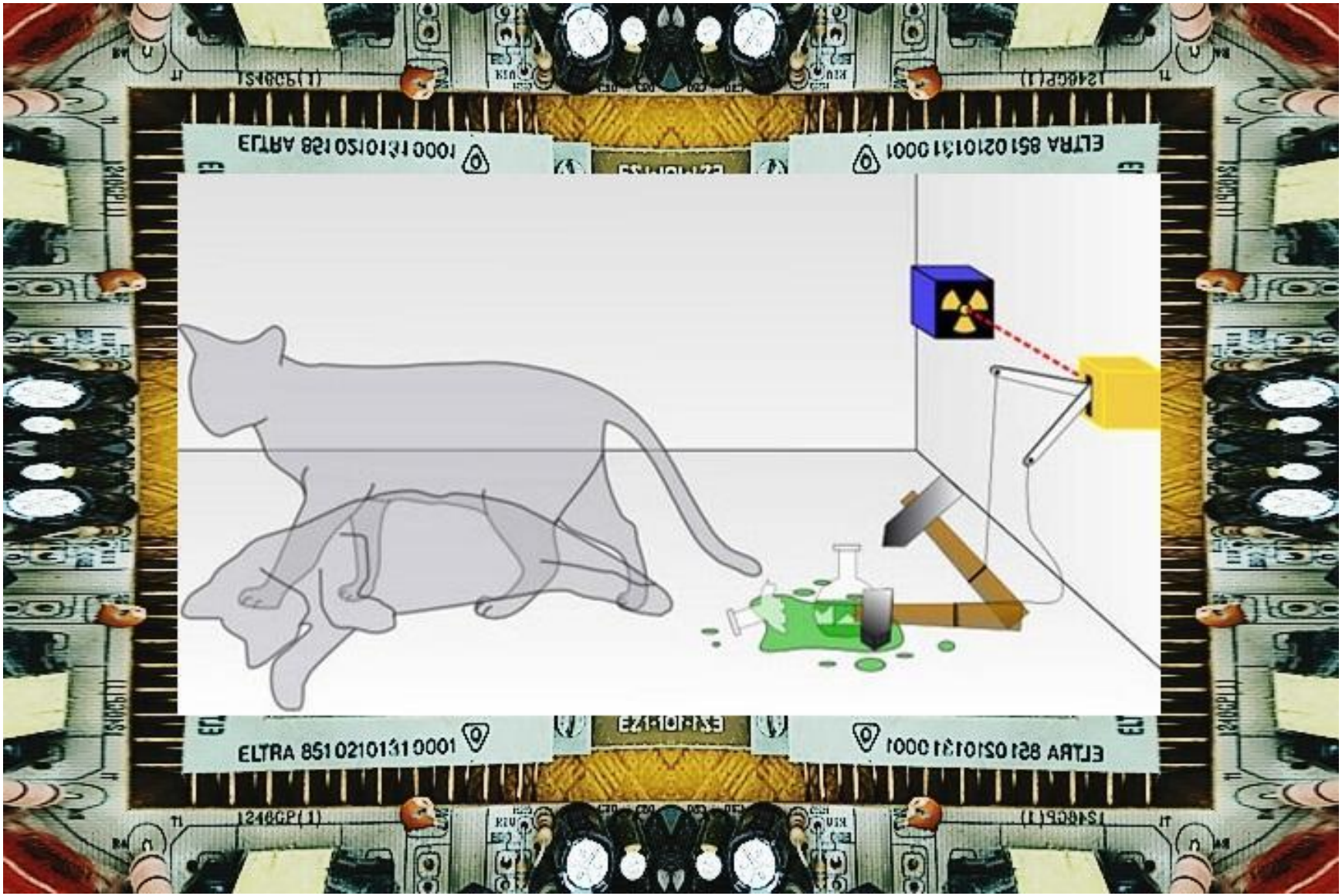
Подтверждение квантовой теории

А вскоре французский физик Ален Аспе провёл опыты, один из которых касался самой сути спора между Энштейном и Бором. **В этом опыте измерение одной частицы могло прямо повлиять на другую только в случае, если сигнал от 1-й ко 2-й прошёл бы со скоростью, превышающей скорость света**. Но сам Энштейн доказал, что это невозможно. Оставалось только одно объяснение – необъяснимая, сверхъестественная связь между частицами.

Подтверждена ли телепортация?

Японские учёные ещё в 2011 году впервые в мире телепортировали фотоны! Мгновенно переместили из пункта А в пункт Б пучок света.
Для этого Нориюки Ли со своими коллегами разложили свет на частицы – фотоны. Один фотон был «квантово запутанным» с другим фотоном. Фотоны были взаимосвязанными, хотя находились в разных точках. **Учёные уничтожили 1-й фотон в точке А, но он был мгновенно воссоздан в точке Б благодаря их «квантовой запутанности»**. До телепортации Кота Шрёдингера ещё, конечно, далеко, но 1-й шаг уже сделан.
.

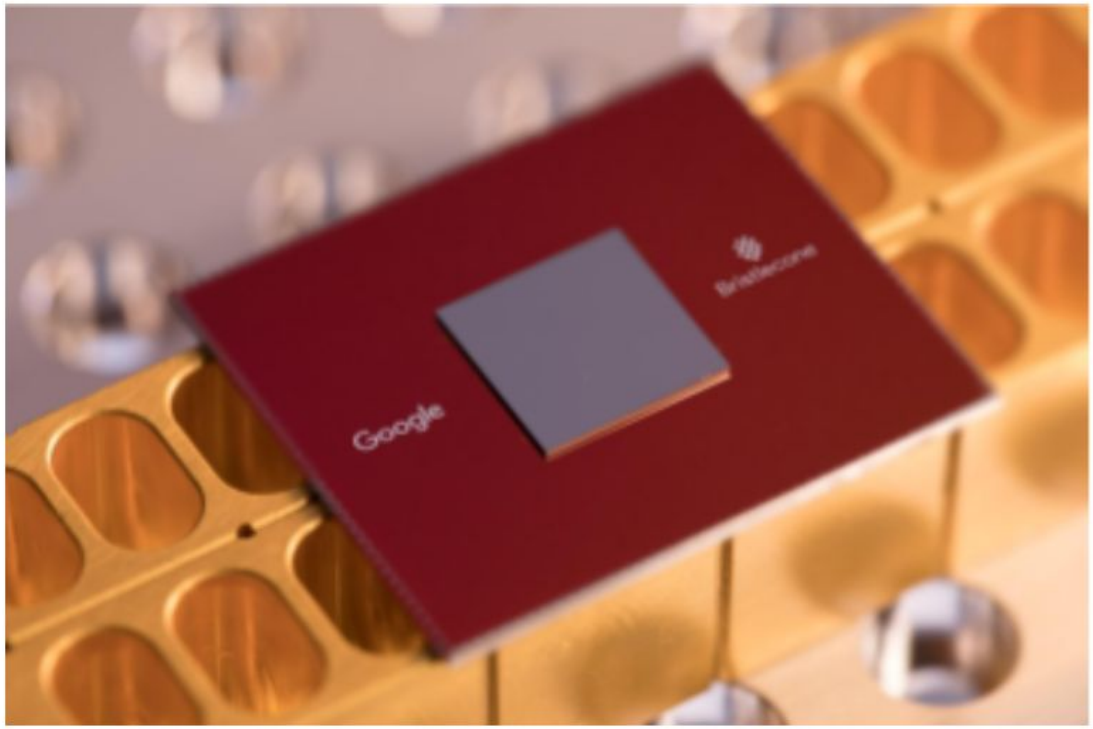
Мысленный эксперимент Шрёдингера

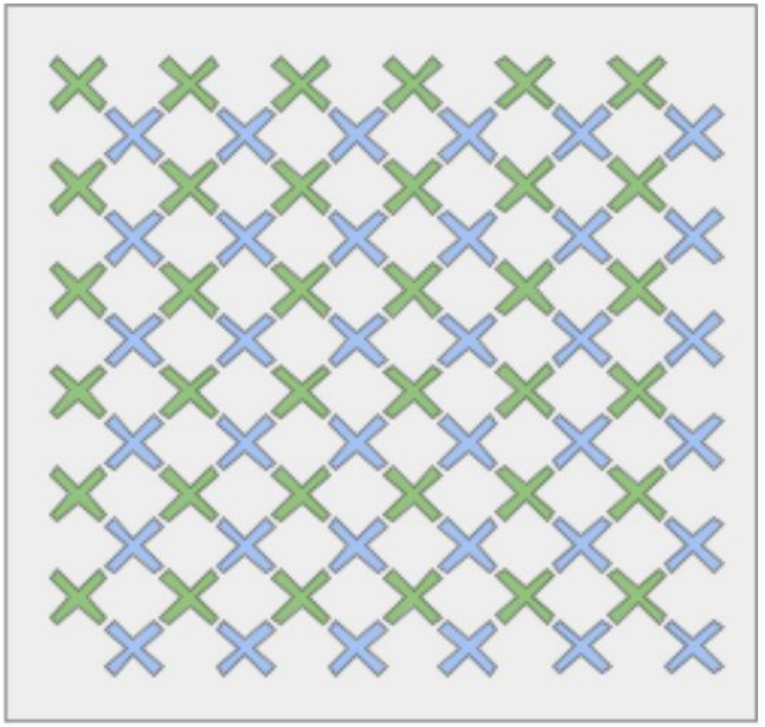


Google представила новый квантовый процессор

Одной из проблем при работе квантового компьютера является количество ошибок, которые возникают при вычислениях, считывании и записи информации в кубиты. В июне 2016 года исследователи из Google построили процессор из 9 кубитов, который показал высокую надёжность. Эту разработку они смогли масштабировать к марту 2018 года, увеличив количество кубитов до 72. В процессоре кубиты расположены в два слоя 6x6 друг над другом. Подразделение Google Quantum AI lab тестирует разработку.

Квантовый процессор Bristlecone состоит из 72 кубитов





Последние достижения

На данный момент квантовыми компьютерами занимаются ряд исследовательских команд, в том числе — IBM. В марте 2017 года компания объявила о запуске проекта IBM Q, и к июню представила два процессора: 16-кубитный для работы в научной сфере и 17-кубитный для коммерческого использования. В 2017 году IBM Research разработала 49-кубитный процессор.

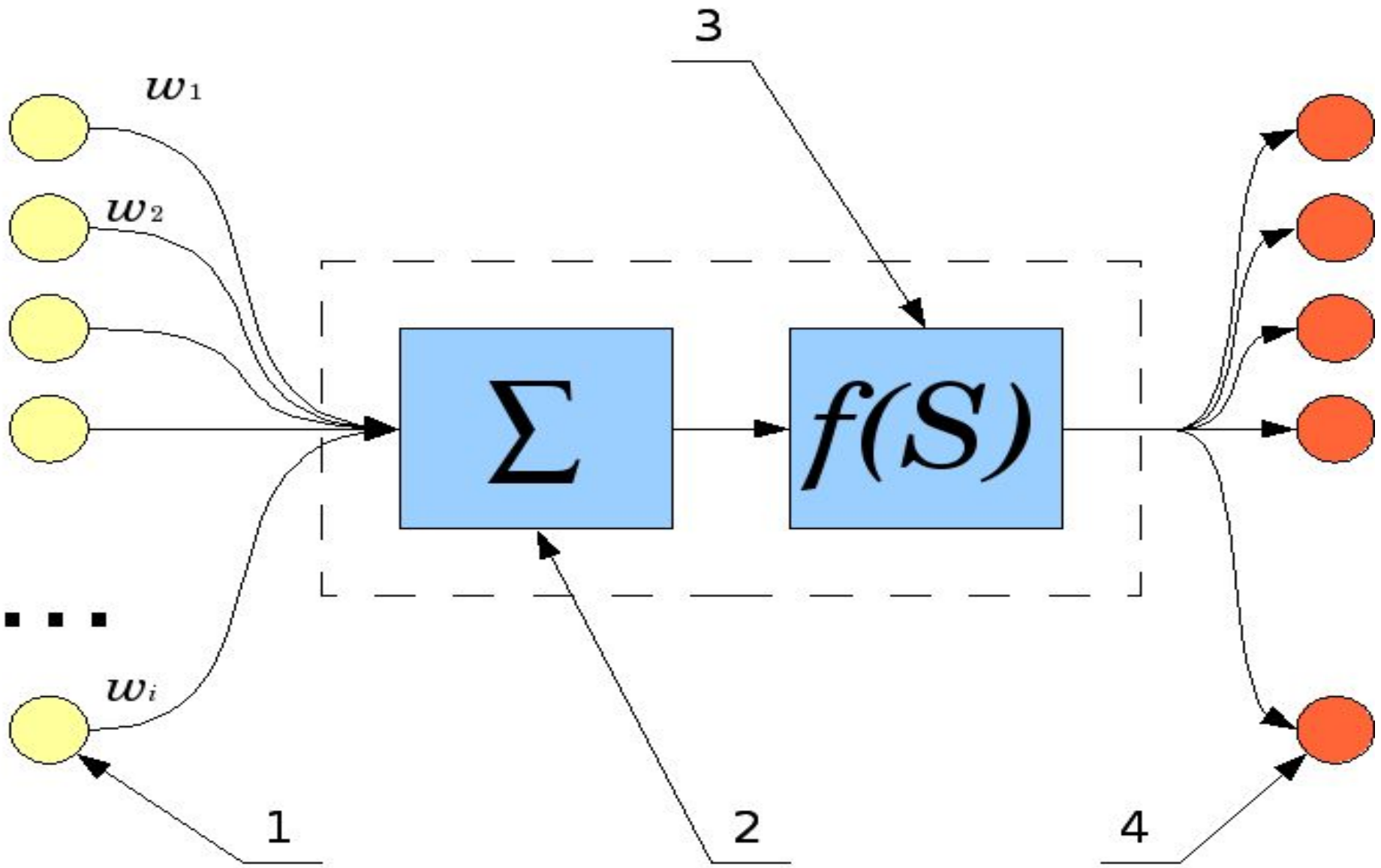
В июле 2017 года команда российских и американских учёных из Гарвардского университета, возглавляемая сооснователем Российского квантового центра (РКЦ) Михаилом Лукиным, сообщила о создании 51-кубитного квантового компьютера.

В России в марте 2018 года между Внешэкономбанком, компанией «ВЭБ Инновации», Фондом перспективных исследований (ФПИ), МГУ имени Ломоносова и АНО «Цифровая экономика» было подписано соглашe6ние о разработке 50-кубитного квантового компьютера.

Нейрокомпьютеры

Нейрокомпьютер — устройство переработки информации на основе принципов работы естественных нейронных систем. Эти принципы были формализованы, что позволило говорить о теории искусственных нейронных сетей.
Проблематика же нейрокомпьютеров заключается в построении реальных физических устройств, что позволит не просто моделировать искусственные нейронные сети на обычном компьютере, но так изменить принципы работы компьютера, что станет возможным говорить о том, что они работают в соответствии с теорией искусственных нейронных сетей.

Формальный нейрон



Как называется подход, основанный на представлении как памяти данных, так и алгоритмов системой связей?

Нейрокомпьютеры (основная идея)

В отличие от цифровых систем, представляющих собой комбинации процессорных и запоминающих блоков, нейропроцессоры содержат память, распределённую в связях между очень простыми процессорами, которые часто могут быть описаны как формальные нейроны или блоки из однотипных формальных нейронов. **Тем самым основная нагрузка на выполнение конкретных функций процессорами ложится на архитектуру системы**, детали которой в свою очередь определяются межнейронными связями. Подход, основанный на представлении как памяти данных, так и алгоритмов системой связей (и их весами), называется **коннекционизмом.**

Основные преимущества нейрокомпьютеров

1. Все алгоритмы нейроинформатики высокопараллельны, а это уже залог высокого быстродействия;

2. Нейросистемы можно легко сделать очень устойчивыми к помехам и разрушениям;

3. Устойчивые и надёжные нейросистемы могут создаваться и из ненадёжных элементов, имеющих значительный разброс параметров.

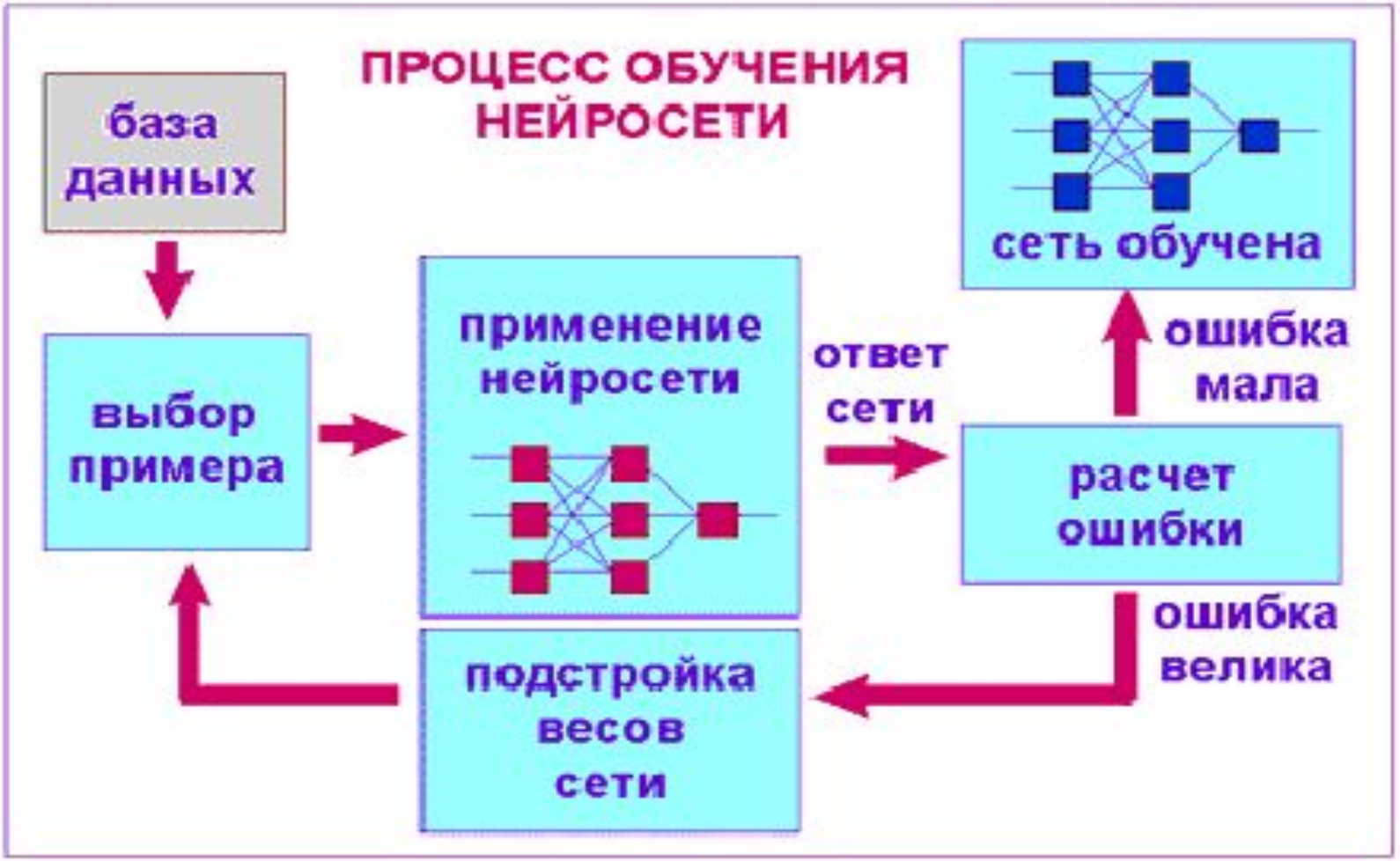
Иными словами

1. однородность системы (элементы одинаковы и чрезвычайно просты, все определяется структурой связей);

2. надежные системы из ненадежных элементов и «аналоговый ренессанс» - использование простых аналоговых элементов;

3. «голографические» системы – при разрушении случайно выбранной части система сохраняет свои полезные свойства.

Обучение нейросети

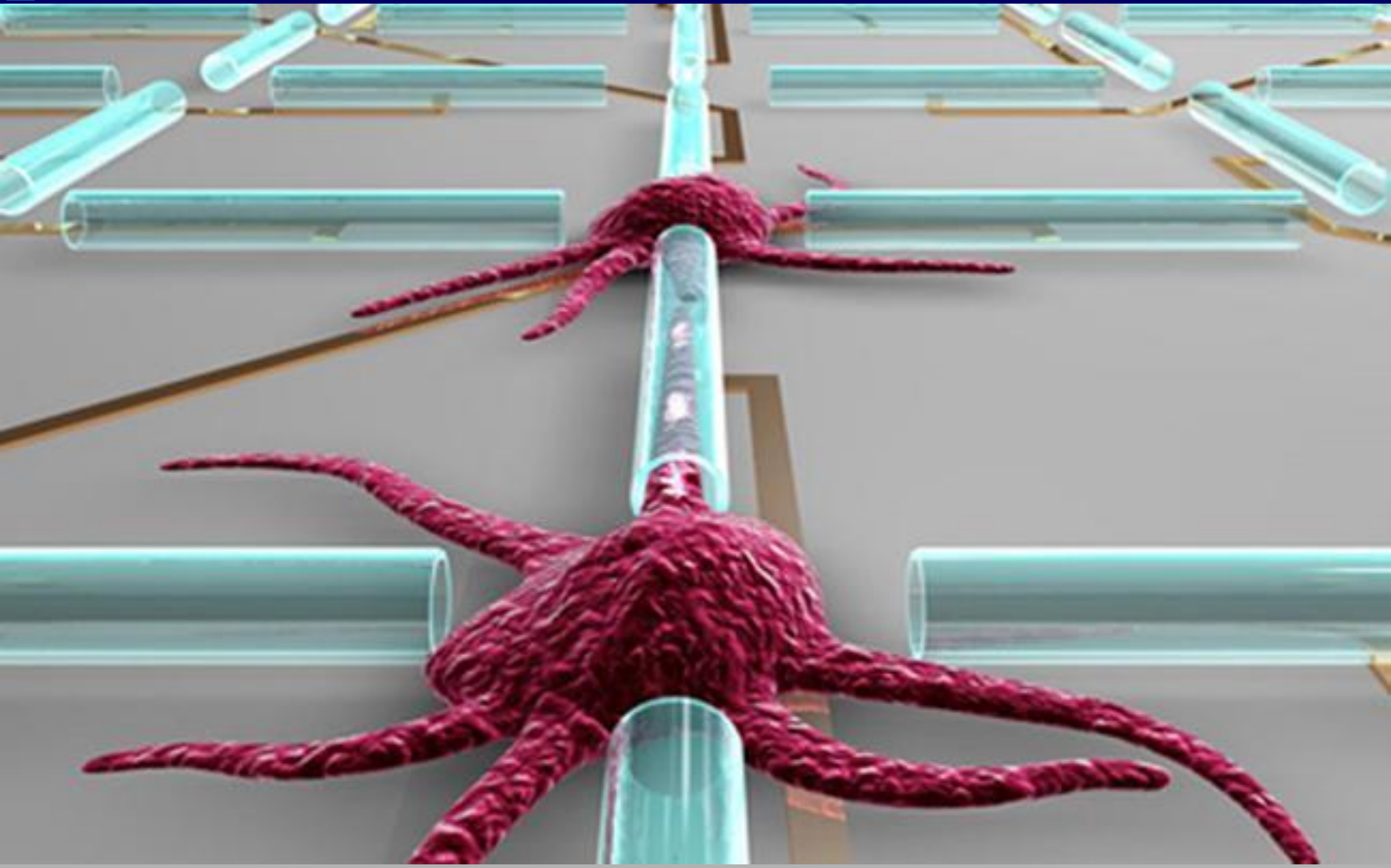


Процесс обучения нейронной сети.

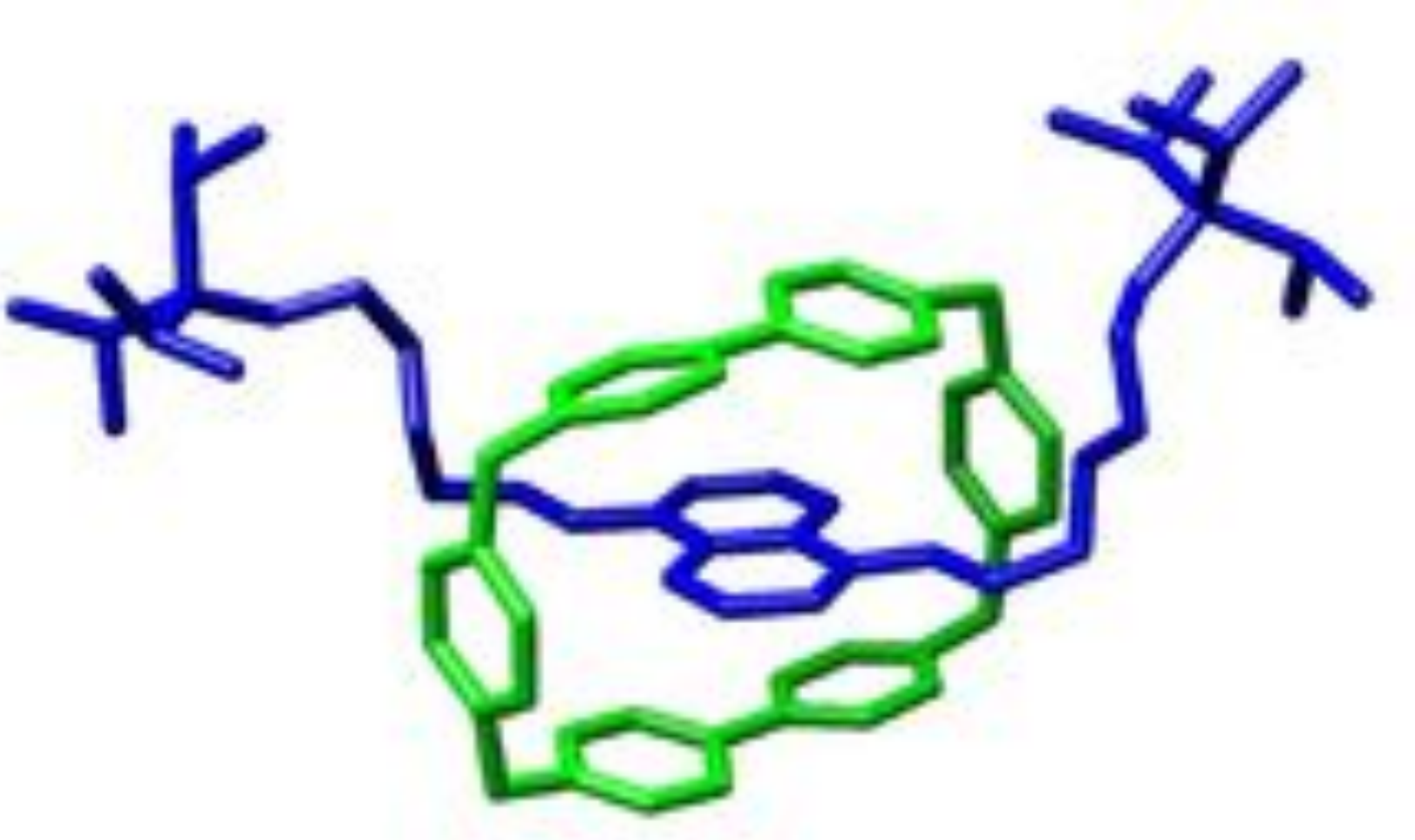
Нейрокомпьютеры (новая идея)

В нейрокомпьютинге постепенно созревает новое направление, основанное на соединении биологических нейронов с электронными элементами. Эти разработки получили наименование **Wetware (*англ.*) — «влажный продукт»**.
В настоящее время уже существует технология соединения биологических нейронов со сверхминиатюрными полевыми транзисторами с помощью нановолокон (**Nanowire (*англ.*)**). В разработках используется современная нанотехнология. В том числе, для создания соединений между нейронами и электронными устройствами используются углеродные нанотрубки.

WetWare



Молекулярные компьютеры



Молекулярные компьютеры

Компания Hewlett-Packard достигла первых успехов в изготовлении компонентов, из которых могут быть построены мощные молекулярные компьютеры. Ученые из HP и Калифорнийского университета в Лос-Анджелесе (UCLA) объявили о том, что им удалось заставить молекулы **ротаксана** (класс синтетических соединений, состоящих из молекулы гантелевидной формы и циклической молекулы, «надетой» на неё) переходить из одного состояния в другое - по существу, это означает создание молекулярного элемента памяти.

Молекулярные компьютеры

Следующим шагом должно стать изготовление логических ключей, способных выполнять функции И, ИЛИ и НЕ. Весь такой компьютер может состоять из слоя проводников, проложенных в одном направлении, слоя молекул ротаксана и слоя проводников, направленных в обратную сторону. Конфигурация компонентов, состоящих из необходимого числа ячеек памяти и логических ключей, создается электронным способом. По оценкам ученых HP, подобный компьютер будет в 100 млрд. раз экономичнее современных микропроцессоров, занимая во много раз меньше места.

Молекулярные компьютеры

Молекулярный компьютер размером с песчинку может содержать миллиарды молекул. **А если научиться делать компьютеры не трехслойными, а трехмерными, преодолев ограничения процесса плоской литографии, применяемого для изготовления микропроцессоров сегодня, преимущества станут еще больше.**

Кроме того, **молекулярные технологии сулят появление микромашин, способных перемещаться и прилагать усилие**. Причем для создания таких устройств можно применять даже традиционные технологии травления. Когда-нибудь эти микромашины будут **самостоятельно заниматься сборкой компонентов молекулярного или атомного размера**.

3. Интеллектуальный анализ данных и управление знаниями

Процесс аналитического исследования больших массивов информации

исследование

построение модели

проверка модели

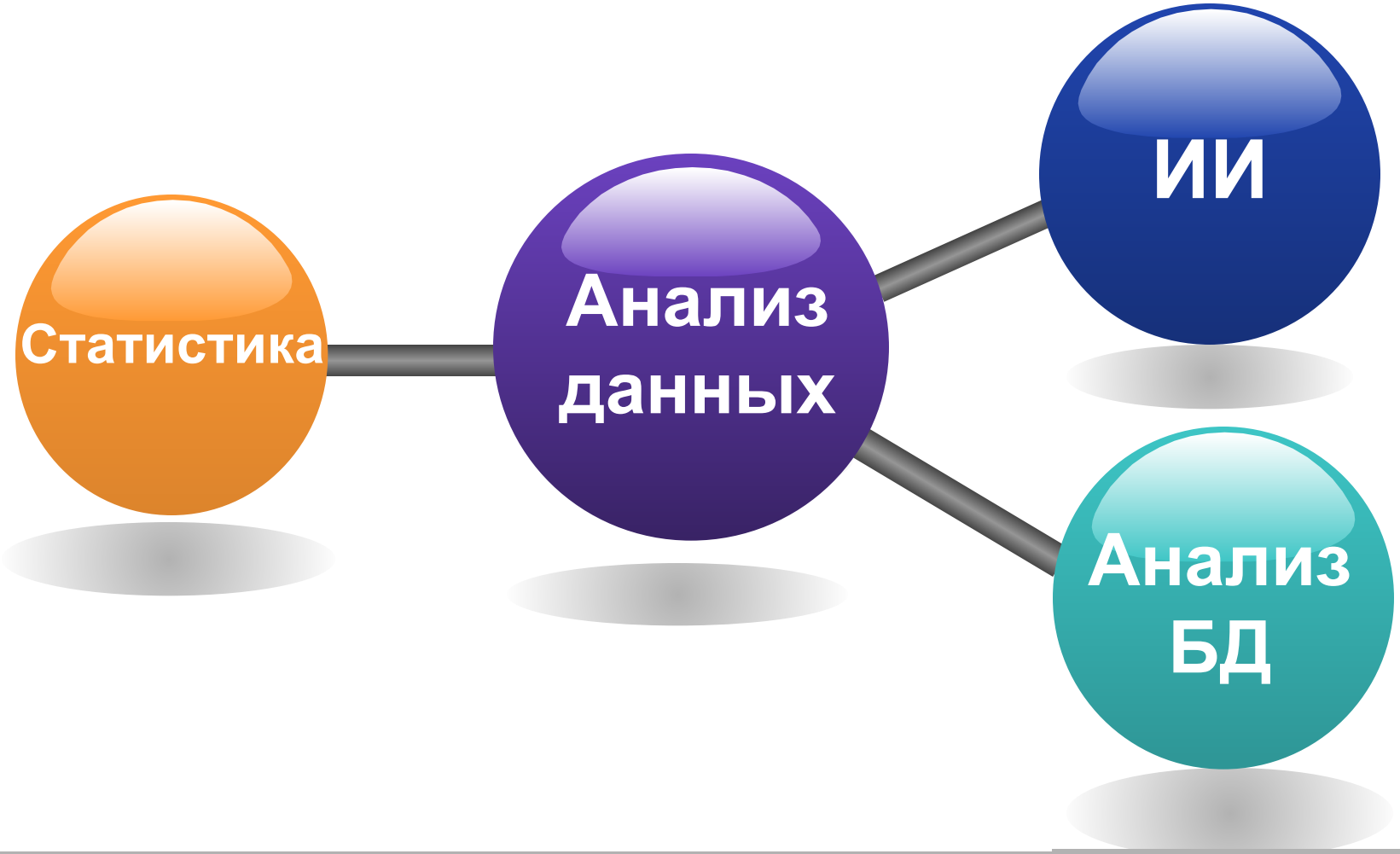
Технологии и методы анализа и интерпретации данных

В **реальной ситуации** практически невозможно проверить экономическую модель на стадии анализа и поэтому *начальные результаты имеют характер эвристик*, которые можно использовать в процессе ***принятия решения***.

Технологии и методы анализа и интерпретации данных

Методы анализа данных основываются на классических принципах **разведочного анализа данных (РАД)**

Технологии и методы анализа и интерпретации данных



Технологии и методы анализа и интерпретации данных

Хранилище данных (*Data Warehouse*) — предметно-ориентированная информационная база данных, специально разработанная и предназначенная для подготовки отчётов и бизнес-анализа с целью поддержки принятия решений. Строится на базе систем управления базами данных и систем поддержки принятия решений. Данные, поступающие в хранилище данных, как правило, доступны только для чтения.

Технологии и методы анализа и интерпретации данных

Хранилища данных ***- способ хранения больших многомерных массивов данных, который позволяет легко извлекать и использовать информацию в процедурах анализа. Эффективная архитектура хранилища данных должна быть организована таким образом, чтобы быть составной частью информационной системы управления.***

Технологии оперативной аналитической обработки данных OLAP

Термин OLAP (или FASMI - быстрый анализ распределенной многомерной информации) обозначает методы, которые дают возможность пользователям многомерных баз данных в реальном времени генерировать описательные и сравнительные сводки ("views") данных и получать ответы на различные другие аналитические запросы.

Технологии оперативной аналитической обработки данных OLAP

Технологии OLAP были разработаны для анализа данных в системах баз данных с целью поддержки принятия решений и ориентированы, главным образом, на обработку **нерегламентированных интерактивных** запросов.

Технологии оперативной аналитической обработки данных OLAP

В *многомерной модели данных база* данных представляется в виде одного или нескольких *кубов данных*, называемых иногда *гиперкубами*. Такой куб имеет несколько независимых измерений, своего рода систему координат представляемого им многомерного пространства данных. Каждому измерению соответствует некоторый *атрибут*, характеризующий какое-либо качественное свойство данных.

Технологии оперативной аналитической обработки данных OLAP

На множестве значений некоторых атрибутов измерений (элементов) могут быть определены *иерархические отношения*. Например, для атрибута-времени может использоваться иерархия «годы — кварталы — месяцы», для атрибута-территории — «регион — город — район».

Технологии оперативной аналитической обработки данных OLAP

Наборы значений измерений по одному для каждого из них определяют точки куба, называемые *ячейками*. С ячейками ассоциируются значения различных других количественных атрибутов, называемых *показателями*.

Технологии оперативной аналитической обработки данных OLAP

Для целей анализа могут строиться *сечения куба данных* (называемые также его *проекциями*) путем фиксации значений различных наборов атрибутов-координат. Может также осуществляться *сжатие куба* на основе использования значений атрибутов измерений более высоких уровней иерархии и соответствующего *агрегирования* значений ассоциированных с ними показателей.

Технологии оперативной аналитической обработки данных OLAP

Возможна также и обратная операция *детализации данных*. Таким образом, возможен анализ данных с нужной степенью детализации. Для удобства восприятия данных в процессе анализа используются различные операции визуализации данных, в частности *вращение куба* путем изменения порядка измерений.

Технологии оперативной аналитической обработки данных OLAP

Город	Товар	Январь	Февраль	Март	Итого
Москва	Позиция 1	10	22	15	47
	Позиция 2	2	7	5	14
	Позиция 3	17	34	20	71
Итого		29	63	40	132
Рязань	Позиция 4	2	0	3	5
	Позиция 3	5	6	3	14
	Позиция 5	12	22	7	41
Итого		19	28	13	60
Владивосток	Позиция 1	7	7	5	19
	Позиция 5	10	12	15	37
	Позиция 2	2	3	0	5
Итого		19	22	20	61

Технологии оперативной аналитической обработки данных OLAP

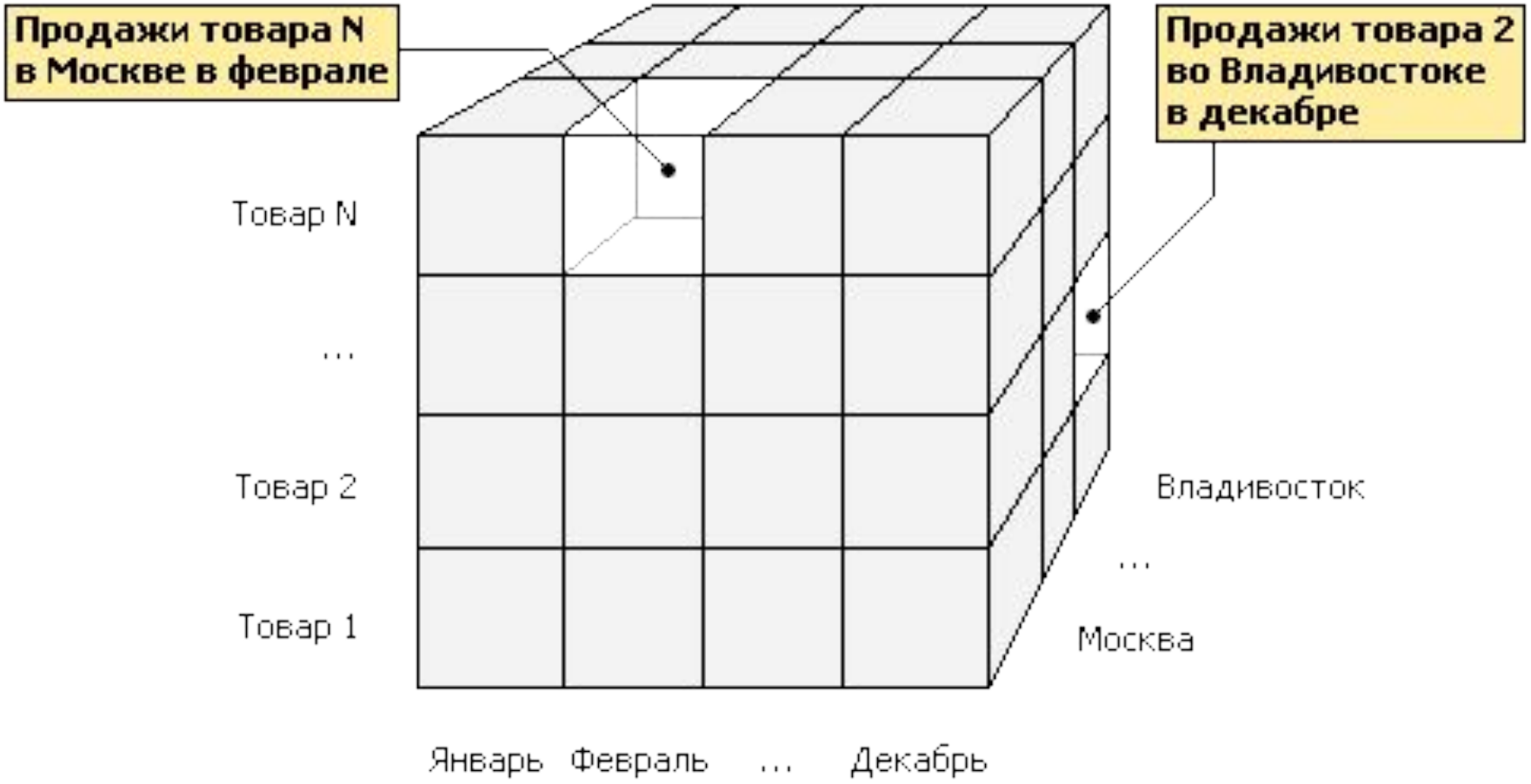


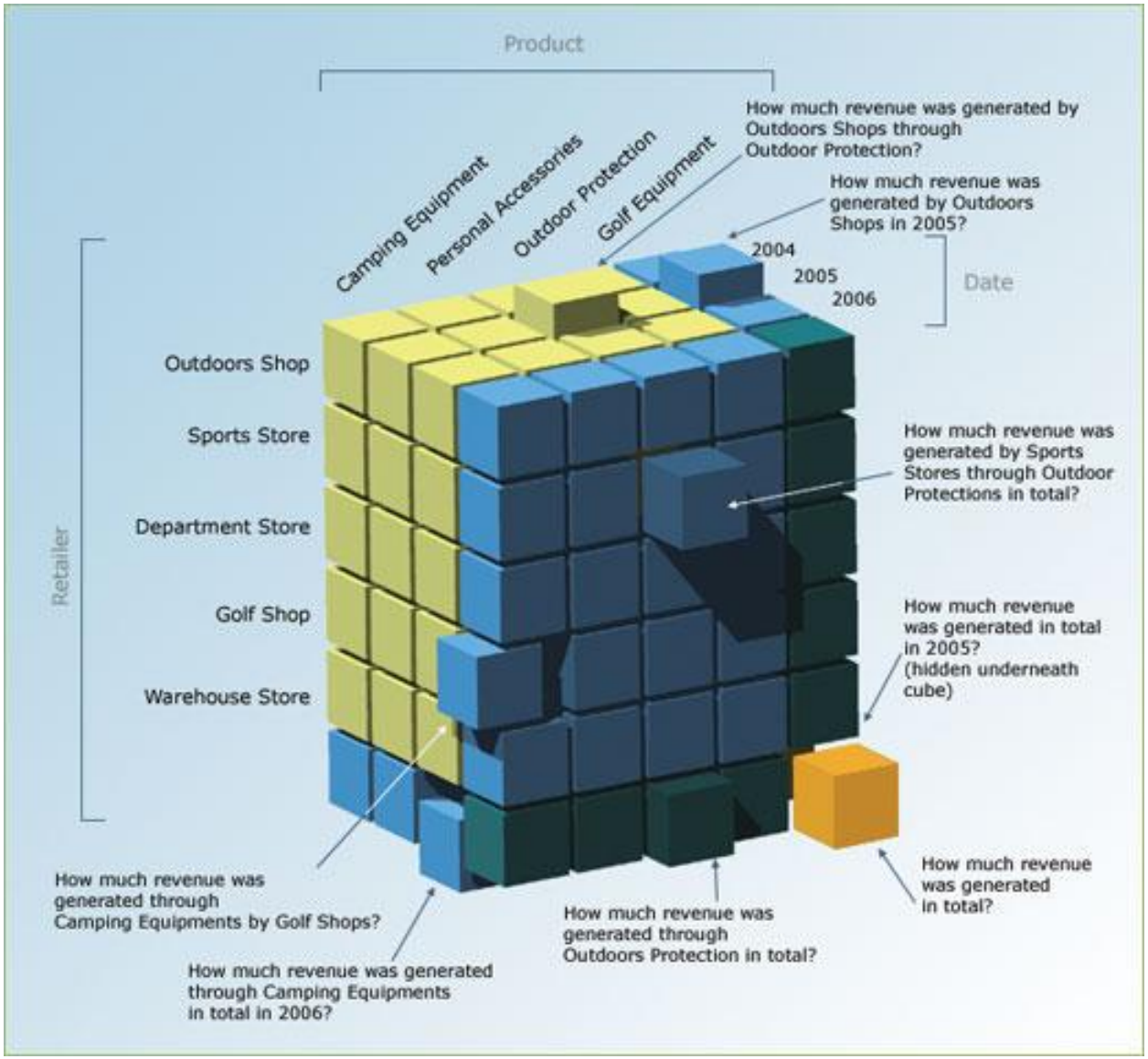
Рис. 1. Данные в трехмерном кубе

Технологии оперативной аналитической обработки данных OLAP

Сложим значения во всех ячейках по вертикали, то получим следующий отчет.

Город	Январь	Февраль	Март	Итого
Москва	29	63	40	132
Рязань	19	28	13	60
Владивосток	19	22	20	61
Итого	67	113	73	253

Технологии оперативной аналитической обработки данных OLAP



Технологии оперативной аналитической обработки данных OLAP

Технологии оперативной аналитической обработки данных OLAP

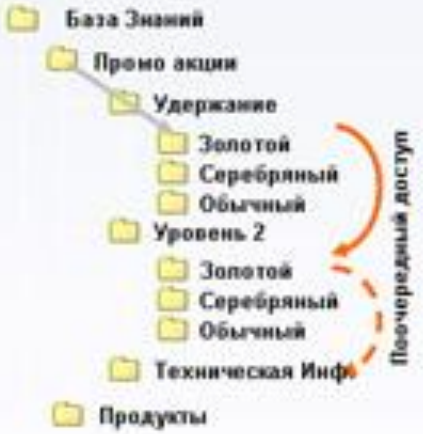
Технология InStranet

Контент распределен по размерностям многомерного куба для более быстрой навигации среди большого объема информации путем фильтрации.

Доступ пользователей к информации определяется распределением их прав.

Стандартная Структура

Легкий путь доступа к информации путем **сортировки...**



...но экстремально медленный

Многомерная Структура

Легкий путь доступа к любой информации **НА ЛЕТУ!!!**

Глубинный анализ данных (Data mining)

Специфика систем глубинного анализа данных состоит в том, что пользовательские запросы не только имеют, как правило, нерегламентированный характер, но и, в отличие от запросов в OLAP, нечетко формулируются.

Глубинный анализ данных (Data mining)

Хотя методы добычи данных можно применять к любой, предварительно не обработанной и даже неструктурированной информации, их можно также использовать для анализа данных и отчетов, полученных средствами OLAP, с целью более углубленного исследования, как правило, в более высоких размерностях.

Методы построения математических моделей в Data mining

статистические

вероятностные

оптимизационные

Основные задачи решаемые Data Mining

- классификация
- кластеризация
- выявление ассоциаций
- поиск типовых образцов на заданном множестве
- выявление объектов данных, не соответствующих характеристикам и поведению, общим для всех рассматриваемых данных
- моделирование тенденций во временных рядах

Разведочный анализ данных (РАД)

Разведочный анализ данных (РАД) применяется для нахождения связей между переменными в ситуациях, когда отсутствуют (или недостаточны) априорные представления о природе этих связей

Разведочный анализ данных (РАД)

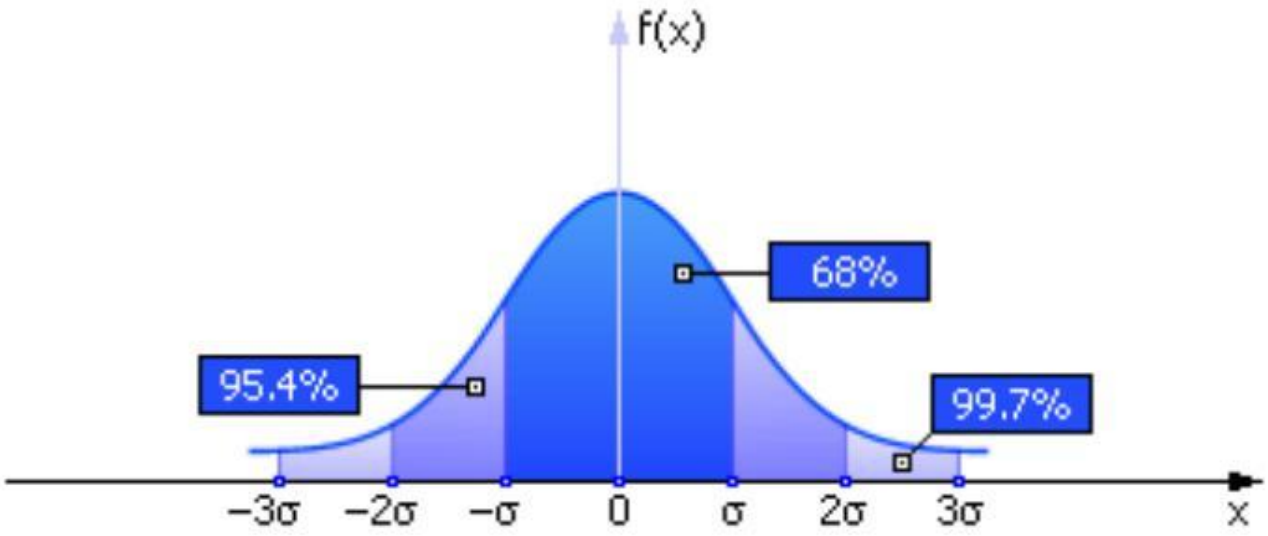
Основные методы

Анализ распределений переменных

Важным способом описания переменной является **форма ее распределения**, которая показывает, с какой частотой значения переменной попадают в определенные интервалы. Эти интервалы, называемые *интервалами группировки*, выбираются исследователем. Обычно исследователя интересует, насколько точно распределение можно аппроксимировать **нормальным**.

Анализ распределений переменных

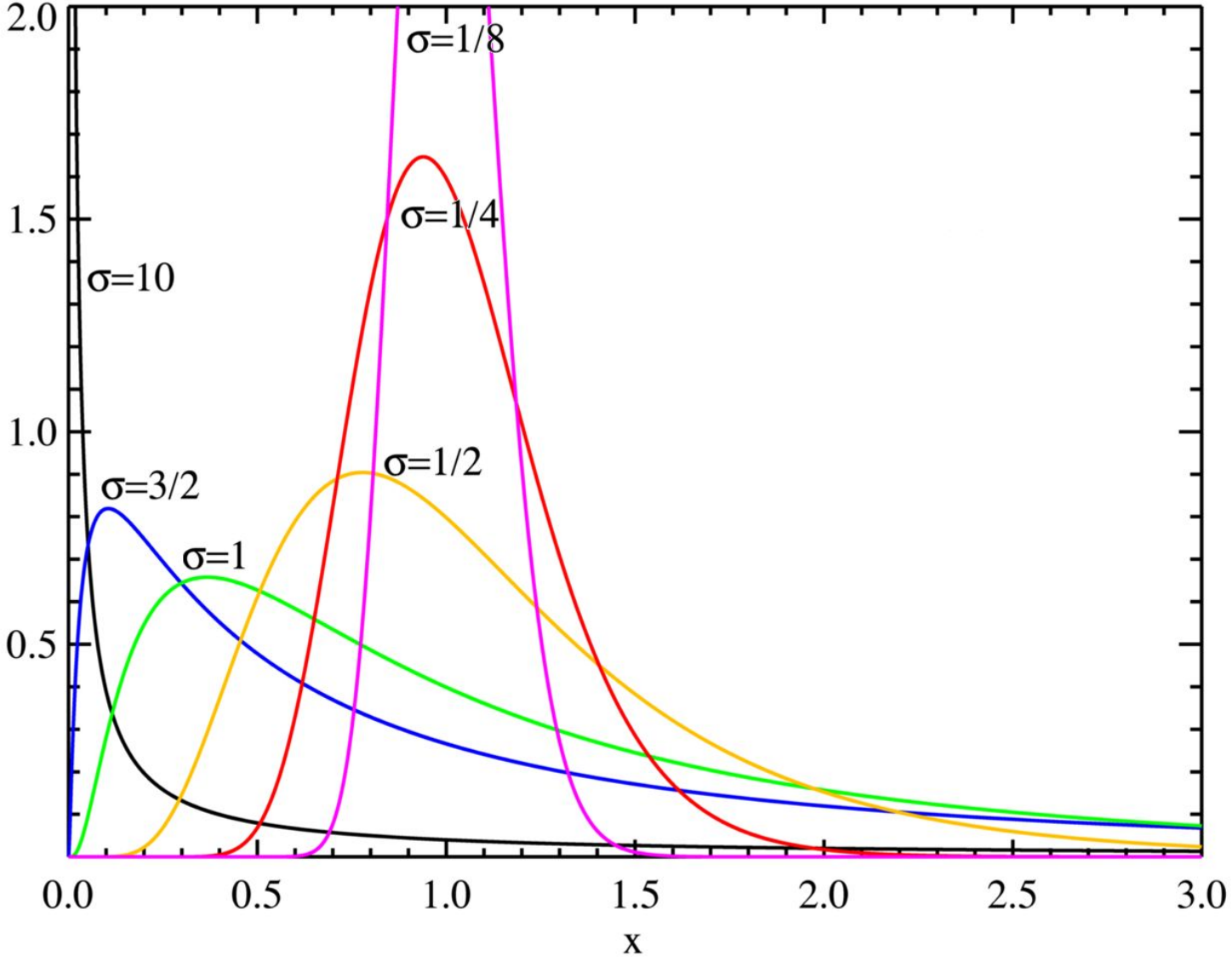
- Графический вид нормализованного распределения Гаусса



Анализ распределений переменных

Если асимметрия (показывающая отклонение распределения от симметричного) существенно отличается от 0, то распределение несимметрично, в то время как нормальное распределение абсолютно симметрично. Итак, у симметричного распределения асимметрия равна 0. Асимметрия распределения с длинным правым хвостом положительна.

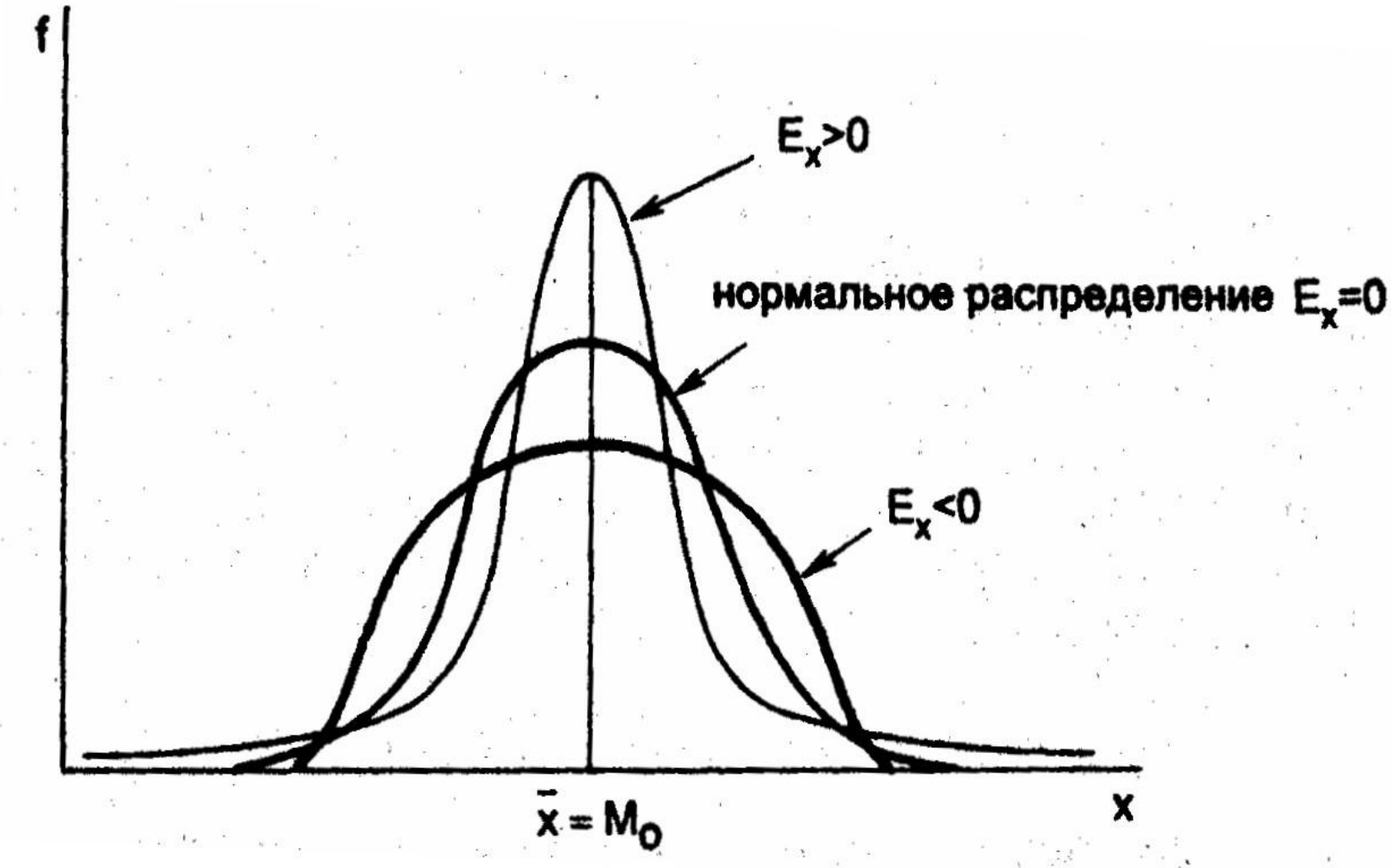
Анализ распределений переменных



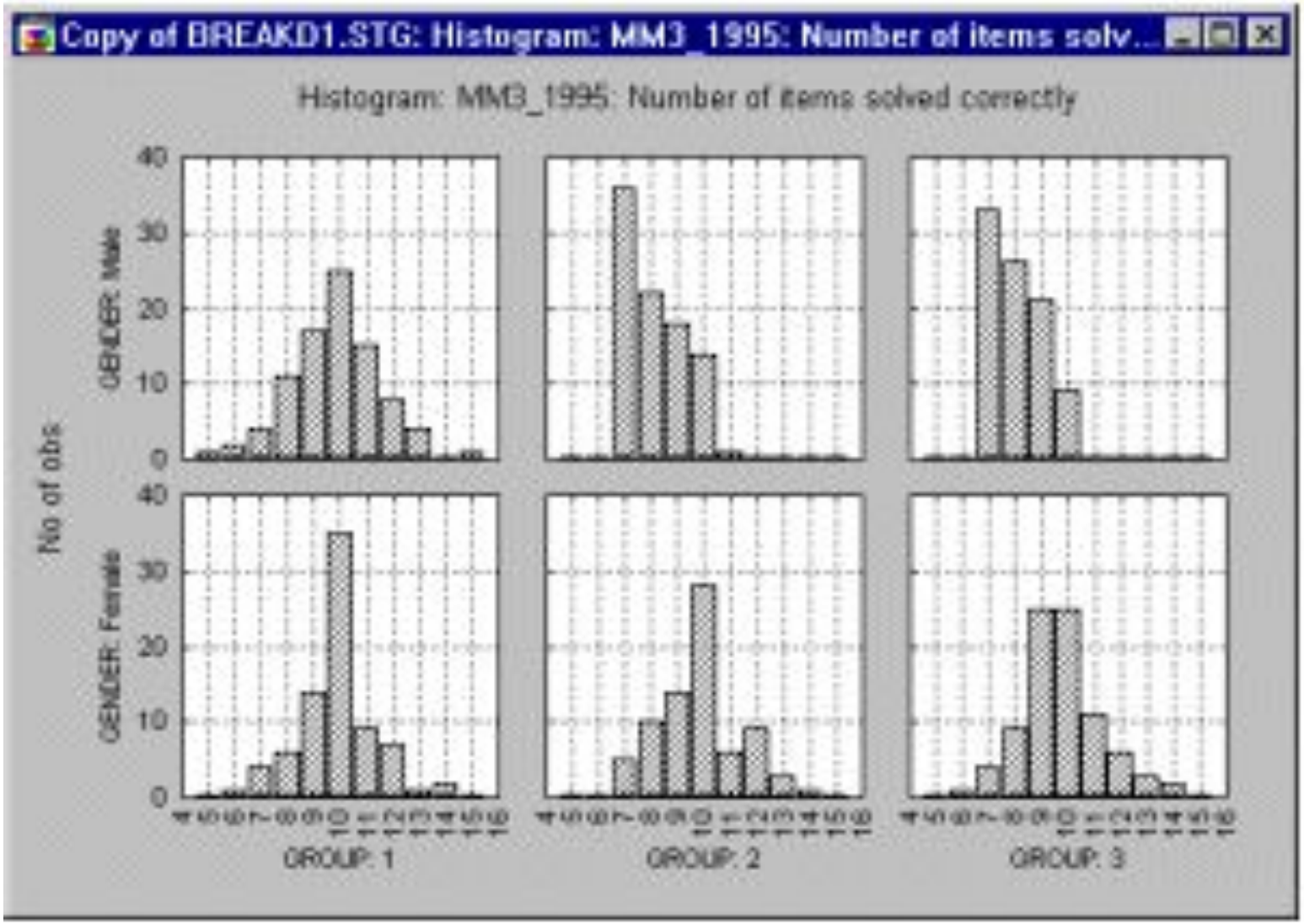
Анализ распределений переменных

Если распределение имеет длинный левый хвост, то его **асимметрия отрицательна**. Далее, если эксцесс (показывающий "остроту пика" распределения) существенно отличен от 0, то распределение имеет или более закругленный пик, чем нормальное, или, напротив, имеет более острый пик (возможно, имеется несколько пиков). Обычно, если **эксцесс положителен, то пик заострен**, если **отрицательный, то пик закруглен**. Эксцесс нормального распределения равен 0.

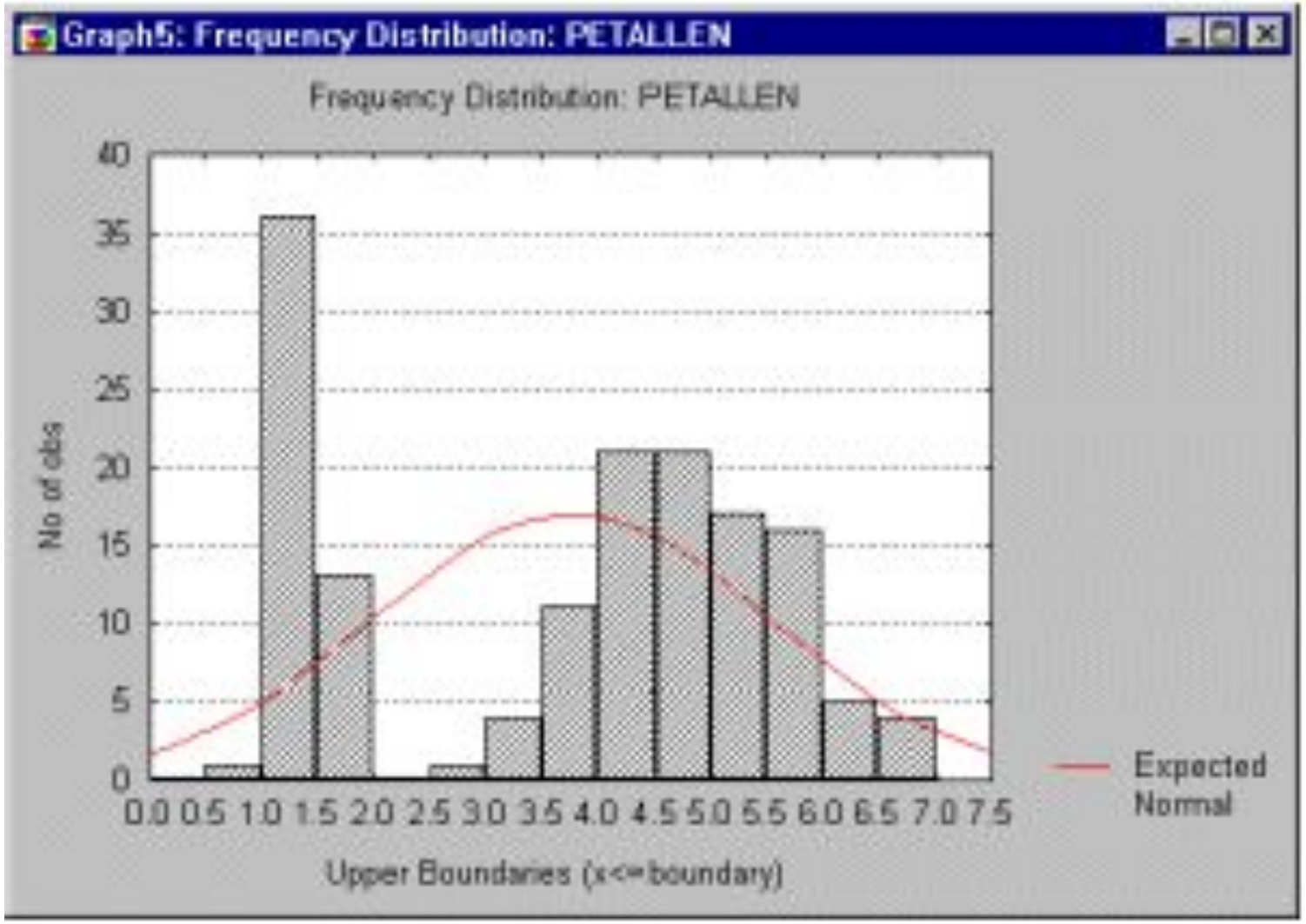
Анализ распределений переменных



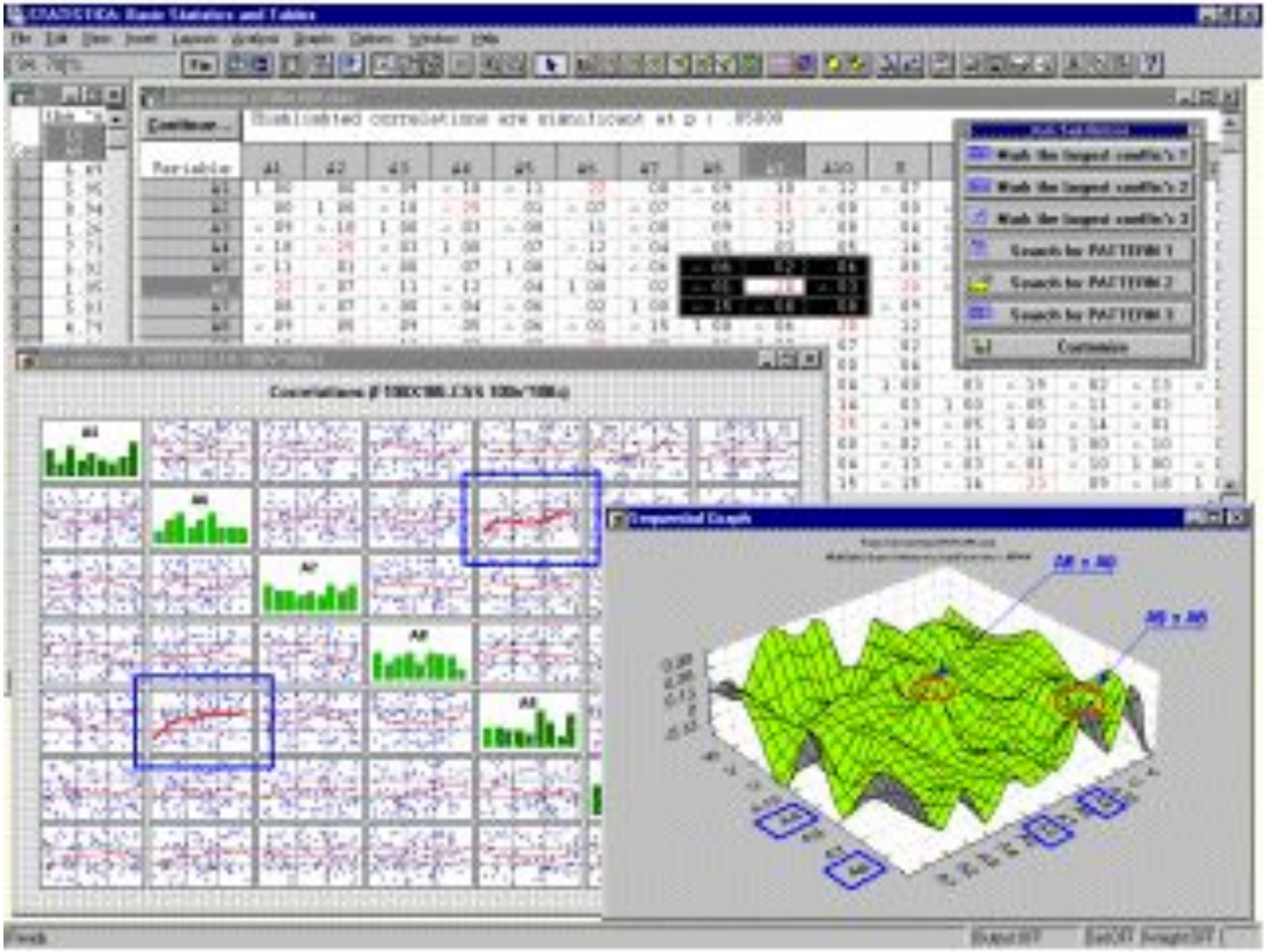
Анализ распределений переменных



Анализ распределений переменных



Анализ распределений переменных



Разведочный анализ корреляционных матриц

Корреляция представляет собой меру зависимости переменных. Наиболее известна корреляция Пирсона. При вычислении корреляции Пирсона предполагается, что переменные измерены, как минимум, в интервальной шкале. Некоторые другие коэффициенты корреляции могут быть вычислены для менее информативных шкал.

Разведочный анализ корреляционных матриц

Коэффициенты корреляции изменяются в пределах от -1.00 до +1.00. Значение -1.00 означает, что переменные имеют строгую отрицательную корреляцию. Значение +1.00 означает, что переменные имеют строгую положительную корреляцию. Значение 0.00 означает отсутствие корреляции.

Разведочный анализ корреляционных матриц

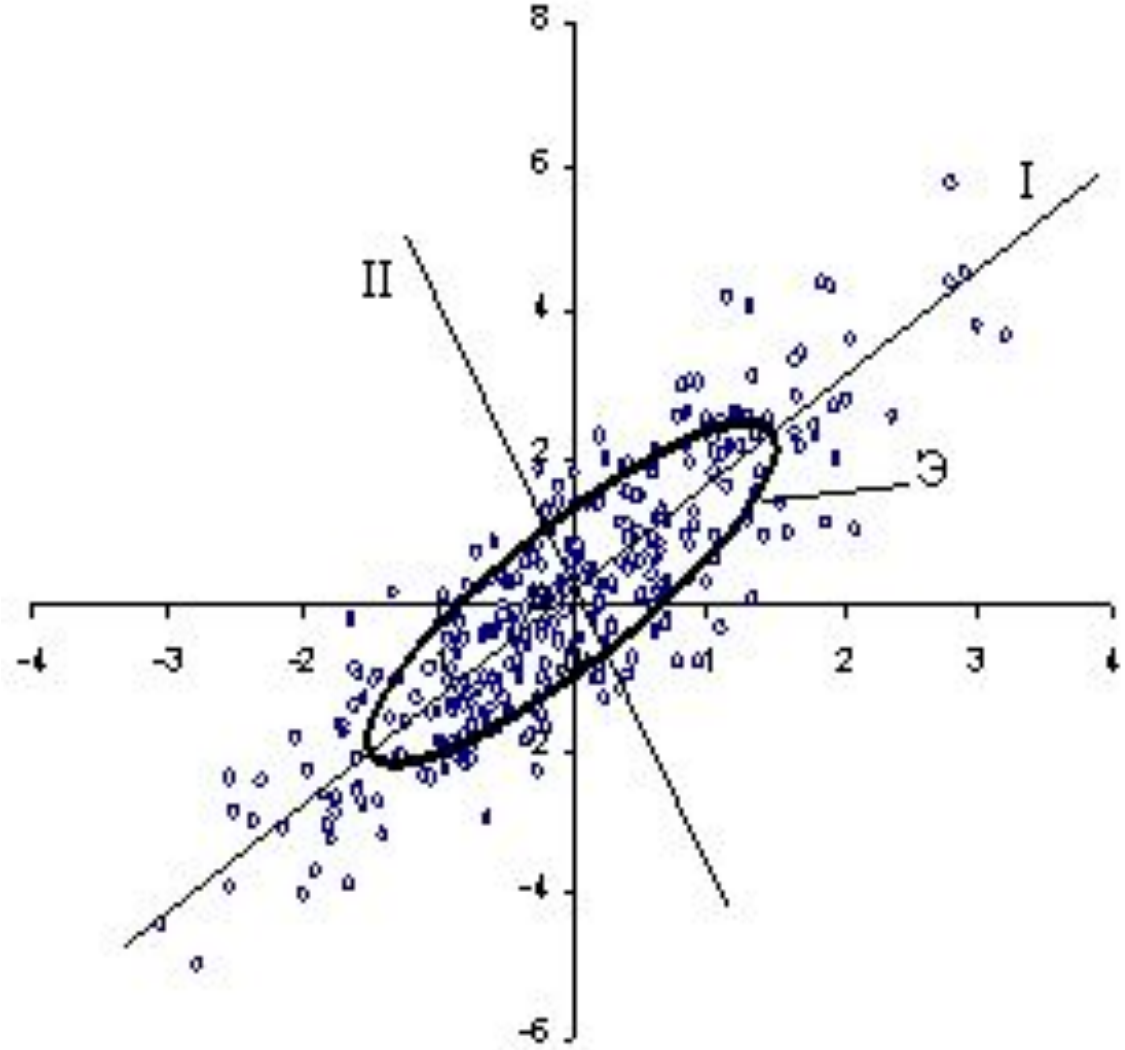
Наиболее часто используемый коэффициент корреляции Пирсона (r) называется также линейной корреляцией, т.к. измеряет степень линейных связей между переменными. Важно, что значение коэффициента корреляции не зависит от масштаба измерения.

Разведочный анализ корреляционных матриц

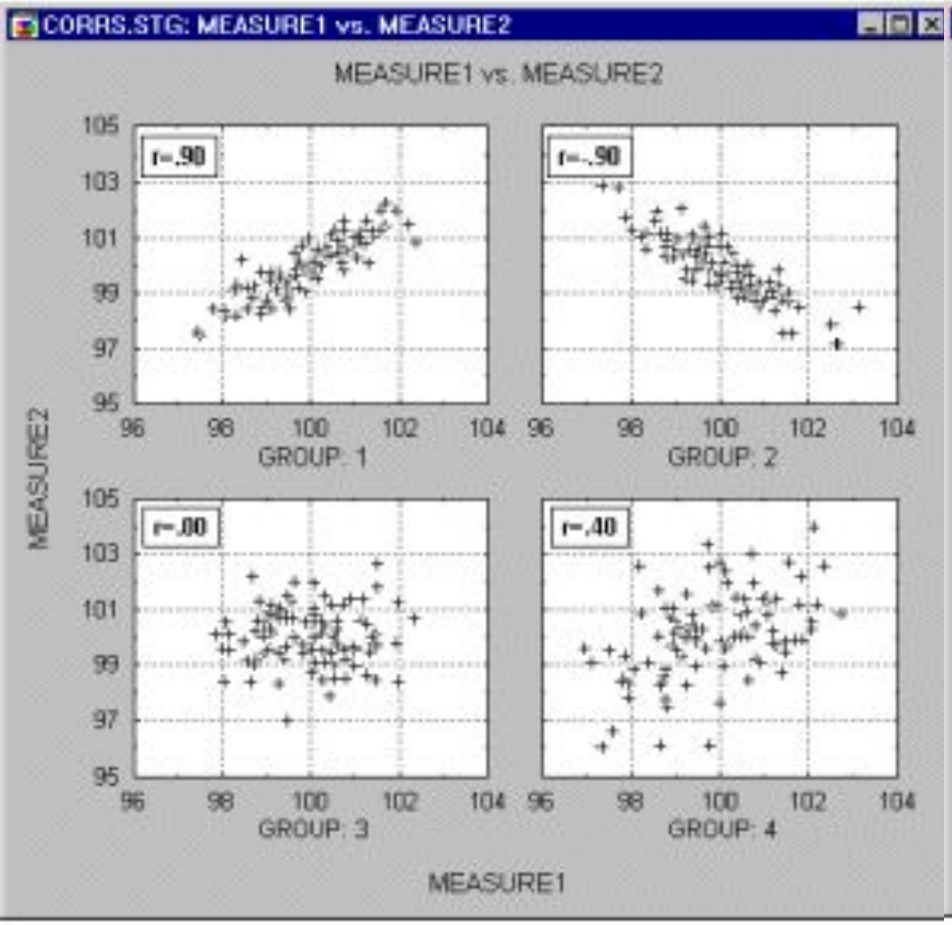
Корреляция высокая, если на графике зависимость выражена прямой линией (с положительным или отрицательным углом наклона). Интервальная шкала позволяет не только упорядочить наблюдения, но и количественно выразить расстояния между ними (на шкале не обязательно присутствует абсолютная нулевая отметка).

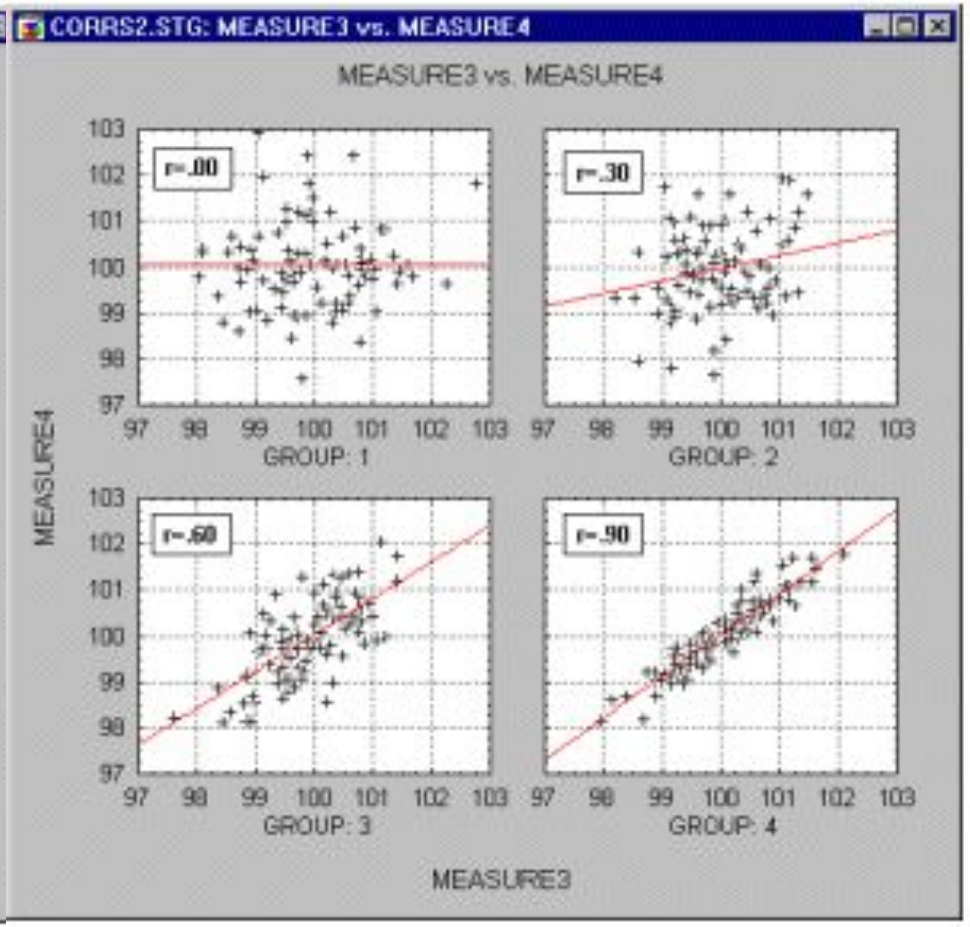
Разведочный анализ корреляционных матриц

Разведочный анализ корреляционных матриц



Разведочный анализ корреляционных матриц





Разведочный анализ корреляционных матриц

Проведенная прямая называется ***прямой регрессии*** или прямой, построенной *методом наименьших квадратов*. Последний термин связан с тем, что сумма *квадратов* расстояний (вычисленных по оси Y) от наблюдаемых точек до прямой является минимальной. Заметим, что использование *квадратов* расстояний приводит к тому, что оценки параметров прямой сильно реагируют на выбросы.

Разведочный анализ корреляционных матриц

Выбросы. По определению, выбросы являются нетипичными, резко выделяющимися наблюдениями. Так как при построении прямой регрессии используется сумма *квадратов* расстояний наблюдаемых точек до прямой, то выбросы могут существенно повлиять на наклон прямой и, следовательно, на значение коэффициента корреляции. Поэтому единичный выброс (значение которого возводится в квадрат) способен существенно изменить наклон прямой и, следовательно, значение корреляции.

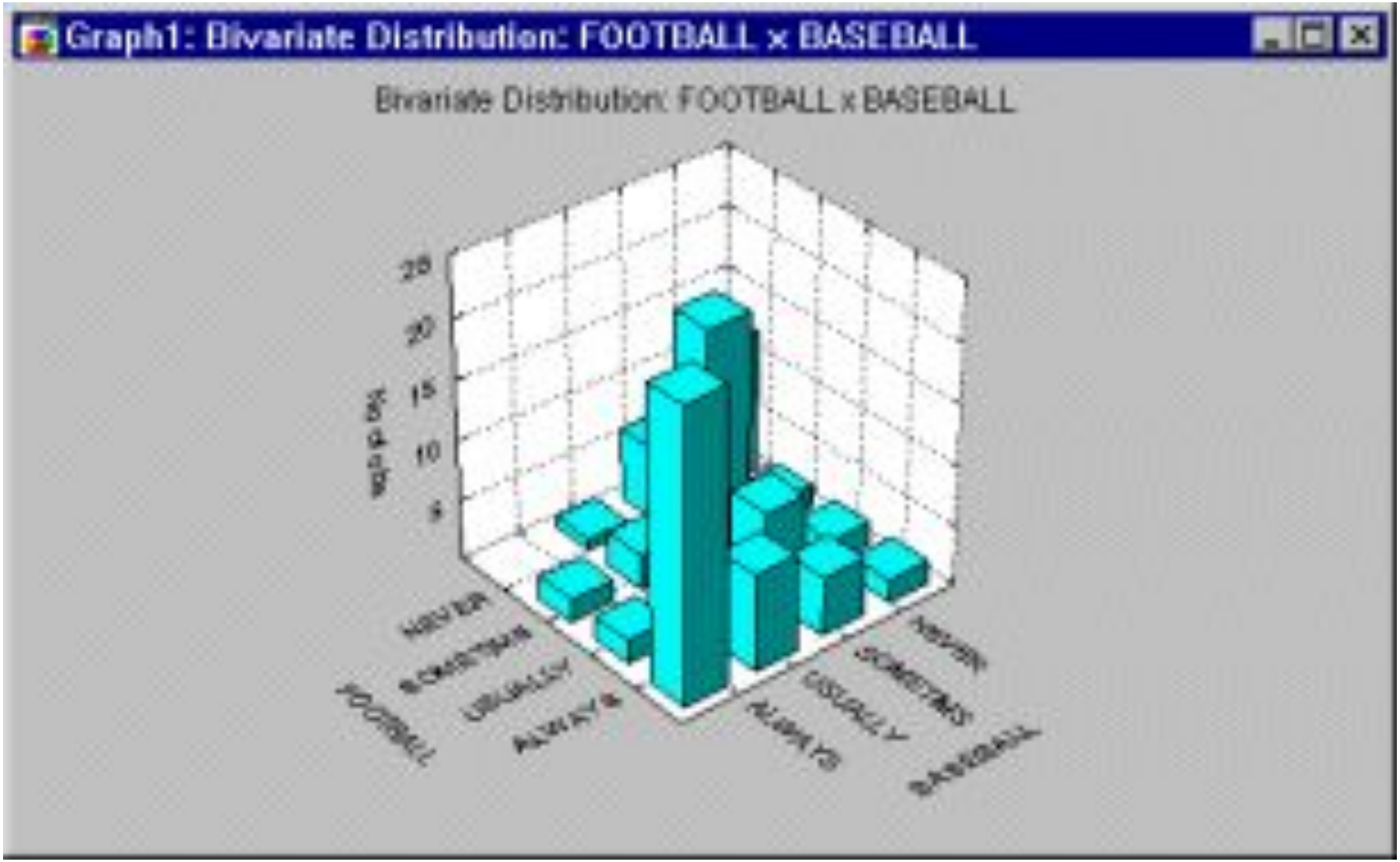
Анализ таблиц частот

Кросстабуляция - это процесс объединения двух (или нескольких) таблиц частот так, что каждая ячейка (клетка) в построенной таблице представляется единственной комбинацией значений или уровней табулированных переменных. Таким образом, кросстабуляция позволяет совместить частоты появления наблюдений на разных уровнях рассматриваемых факторов.

Анализ таблиц частот

В целях исследования отдельные строки и столбцы таблицы удобно представлять в виде графиков. Полезно также отобразить целую таблицу на отдельном графике. Таблицы с двумя входами можно изобразить на 3-мерной гистограмме.

Анализ таблиц частот



Анализ таблиц частот

Другой способ визуализации таблиц сопряженности - построение категоризованной гистограммы, в которой каждая переменная представлена индивидуальными гистограммами на каждом уровне другой переменной. Преимущество 3М гистограммы в том, что она позволяет представить на одном графике таблицу целиком. Достоинство категоризованного графика в том, что он дает возможность точно оценить отдельные частоты в каждой ячейке.

Методы многомерного разведочного анализа

Методы многомерного разведочного анализа специально разработаны для поиска закономерностей в многомерных данных (или последовательностях одномерных данных). К ним относятся: *кластерный анализ*, *факторный анализ*, *анализ дискриминантных функций*, *многомерное шкалирование*, *логлинейный анализ*, *канонические корреляции*, *пошаговая линейная и нелинейная регрессия*, *анализ соответствий*, *анализ временных рядов* и *деревья классификации*.

Кластерный анализ

Кластерный анализ. Термин в действительности включает в себя **набор различных алгоритмов классификации**. Общий вопрос состоит в том, как организовать наблюдаемые данные в наглядные структуры, т.е. **развернуть таксономии**.

Кластерный анализ является не столько обычным статистическим методом, сколько набором различных алгоритмов распределения объектов по кластерам.

Кластерный анализ

Независимо от предмета изучения применение кластерного анализа предполагает *следующие этапы*: — *Отбор выборки для кластеризации. — Определение множества переменных, по которым будут оцениваться объекты в выборке. — Вычисление значений той или иной меры сходства между объектами. — Применение метода кластерного анализа для создания групп сходных объектов. — Проверка достоверности результатов кластерного решения.*

Основные задачи решаемые Кластерным анализом

- Разработка типологии или классификации
- Исследование полезных концептуальных схем группирования объектов
- Порождение гипотез на основе исследования данных
- Проверка гипотез или исследования для определения, действительно ли типы (группы), выделенные тем или иным способом, присутствуют в имеющихся данных

Формальная постановка задачи кластеризации

Пусть X – множество объектов Y — множество номеров (имён, меток) кластеров. Задана функция расстояния между объектами $\rho(x, x')$. Имеется конечная обучающая выборка объектов $X^m = \{x_1, \ldots, x_m\} \subset X$. Требуется разбить выборку на непересекающиеся подмножества, называемые *кластерами*, так, чтобы каждый кластер состоял из объектов, близких по метрике ρ, а объекты разных кластеров существенно отличались. При этом каждому объекту $x_i \in X^m$ приписывается номер кластера y_i.

Алгоритм кластеризации — это функция $a: X \to Y$, которая любому объекту $x \in X$ ставит в соответствие номер кластера $y \in Y$. Множество Y в некоторых случаях известно заранее, однако чаще ставится задача определить оптимальное число кластеров, с точки зрения того или иного *критерия качества* кластеризации.

Кластерный анализ

Основные недостатки

Объединение (древовидная кластеризация)

Объединение (древовидная кластеризация). Назначение этого алгоритма состоит в объединении объектов в достаточно большие кластеры, используя некоторую меру сходства или расстояние между объектами. Типичным результатом такой кластеризации является иерархическое дерево.

Объединение (древовидная кластеризация)

Объединение (древовидная кластеризация)

Кластерный **анализ содержит эффективную двухвходовую процедуру объединения. Однако двухвходовое объединение используется (относительно редко) в обстоятельствах, когда ожидается, что и наблюдения и переменные одновременно вносят вклад в обнаружение осмысленных кластеров.**

Объединение (древовидная кластеризация)

Можно предположить, что медицинскому исследователю требуется выделить **кластеры пациентов, сходных по отношению к определенным кластерам характеристик физического состояния**. Трудность с интерпретацией полученных результатов возникает вследствие того, что сходства между различными кластерами могут происходить из (или быть причиной) некоторого различия подмножеств переменных. Поэтому **получающиеся кластеры являются по своей природе неоднородными.**

Объединение (древовидная кластеризация)

В сравнении с другими описанными методами кластерного анализа, **двухвходовое объединение является, наименее часто используемым методом**. Однако некоторые исследователи полагают, что он предлагает мощное средство разведочного анализа.

Метод К-средних (C-means)

Алгоритм разделительной кластеризации, основанный на разбиении множества элементов векторного пространства на заранее определенное число кластеров ***k***. Алгоритм представляет собой итерационную процедуру, в которой выполняются следующие шаги:

Метод К-средних (C-means)

1. Выбирается число кластеров *k*.
2. Из исходного множества данных случайным образом выбираются *k* записей, которые будут служить начальными центрами кластеров.
3. Для каждой записи исходной выборки определяется ближайший к ней центр кластера. При этом записи, «притянутые» определенным центром, образуют начальные кластеры.

Метод К-средних (C-means)

4. Вычисляются *центроиды* – центры тяжести кластеров. Каждый центроид – это вектор, элементы которого представляют собой средние значения признаков, вычисленные по всем записям кластера. Затем центр кластера смещается в его центроид.

Метод К-средних (C-means)

Затем 3-й и 4-й шаги итеративно повторяются. Очевидно, что на каждой итерации происходит изменение границ кластеров и смещение их центров. В результате минимизируется расстояние между элементами внутри кластеров. Остановка алгоритма производится тогда, когда границы кластеров и расположения центроидов не перестанут изменяться от итерации к итерации, т.е. на каждой итерации в каждом кластере будет оставаться один и тот же набор записей. Алгоритм обычно находит набор стабильных кластеров за несколько десятков итераций.

Метод К-средних (C-means)

Преимуществом алгоритма являются быстрота и простота реализации. К его **недостаткам** можно отнести неопределенность выбора начальных центров кластеров, а также то, что число кластеров должно быть задано изначально, что может потребовать некоторой априорной информации об исходных данных.