

1. Понятие информационной системы, информационной технологии, проектирования, информационной модели. 2	
2. Подходы к построению и проектированию информационных систем.....	3
3. Понятие метода проектирования ИС, их классификация.....	4
4. Основные принципы системного подхода к созданию ИС.....	5
5. Понятие технологии проектирования. Требования к технологии проектирования.....	6
6. Классификация технологий проектирования ИС. Выбор технологии проектирования ИС.....	7
7. Классификация средств проектирования ИС.....	8
8. Основные стадии жизненного цикла проектирования ИС. ....	9
9. Модели жизненного цикла ИС (основные). ....	10
10. Модели жизненного цикла ИС (модифицированные). ....	12
11. Соответствие информационных процессов, информационных технологий и средств их проектирования.	13
12. Понятия системной инженерии, программной инженерии: связи, отличия, особенности. ....	14
13. Принципы проектирования базовых информационных технологий.....	15
14. Классификация стандартов на проектирование и разработку информационных систем.....	16
15.Международный стандарт ISO/IEC 12207:1995-08-01 (ГОСТ Р ИСО/МЭК 12207). ....	17
16.Международный стандарт ISO/IEC 15288:2002 (ГОСТ Р ИСО/МЭК 15288-2005).....	18
17.Комплекс стандартов ГОСТ 34.....	20
18.Фирменные стандарты (Oracle CDM, MSF и др.).....	21
19.Понятие методологии проектирования информационных систем. ....	22
20.Методологии структурного подхода (анализа). ....	24
21.Методология моделирования функциональной структуры объектов –SADT. Методология моделирования данных –ERD (Entity-Relationship Diagrams) (case-метод Баркера). ....	27
22.Методология моделирования работы в реальном времени–STD.....	29
23.Методология функционального моделирования процессов –IDEF0. Характеристика диаграмм. Типы взаимосвязей между блоками. ....	31
24.Методология функционального моделирования процессов –IDEF0. Последовательность создания функциональных моделей. ....	32
25.Методология анализа взаимосвязей между информационными потоками –IDEF1(IDEF1X).....	34
26.Методология описания (документирования) и моделирования процессов –IDEF3. ....	37
27.Методология моделирования, анализа бизнес-процессов BPMN. ....	39
28.Реинжиниринг бизнес-процессов в ИС (реорганизации бизнес-процессов –BPMN).....	41
29.Методологии объектно-ориентированного подхода (анализа). ....	42
30.Средства объектно-ориентированного подхода (анализа). ....	43
31.Методологии модельно-ориентированного подхода (анализа). ....	44
32.Средства модельно-ориентированного подхода (анализа). ....	45
33.Гибкие методологии проектирования (agile-методы).....	46
34.Понятие CASE-технологии проектирования ИС. Основные принципы Case-технологии.....	47
35.Классификация CASE-средств, стратегия их выбора.....	48
36.Проблемы и перспективы проектирования ИС и ИТ. ....	49

## 1. Понятие информационной системы, информационной технологии, проектирования, информационной модели.

	ФЗ № 149-ФЗ	ГОСТ 34.321-96	ISO/IEC 2382-1:1993
<b>Информационная система</b>	совокупность содержащейся в базах данных информации и обеспечивающих её обработку информационных технологий и технических средств	это система, которая организует хранение и манипулирование информацией о предметной области	это система обработки информации и соответствующие организационные ресурсы, которые обеспечивают и распространяют информацию.
	ФЗ № 149-ФЗ	ГОСТ 34.003-90	ISO/IEC 38500:2015
<b>Информационная технология</b>	процессы, методы поиска, сбора, хранения, обработки, предоставления, распространения информации и способы осуществления таких процессов и методов.	это приемы, способы и методы применения средств вычислительной техники при выполнении функций сбора, хранения, обработки, передачи и использования данных.	это ресурсы, необходимые для сбора, обработки, хранения и распространения информации.

**Информационная система** - это взаимосвязанная совокупность информационных, технических, программных, математических, организационных, правовых, эргономических, лингвистических, технологических и других средств, а также персонала, предназначенная для сбора, обработки, хранения и выдачи информации и принятия решений.

**Проектирование** (лат. «projectus» - брошенный вперед) - процесс создания прототипа, прообраза предполагаемого или возможного объекта, явления. Результатом проектирования является научно-теоретически и практически обоснованное определение версий или вариантов развития или изменения того или иного объекта, явления.

**Проектирование** - это поиск способа (выбор технологии) создания ИС, который удовлетворяет требованиям функциональности системы средствами имеющихся технологий с учетом заданных ограничений.

**Информационная модель** - это модель объекта, процесса или явления, в которой представлены информационные аспекты моделируемого объекта, процесса или явления. Она является основой разработки моделей ИС

<b>Описательные информационные модели</b>	<b>Формальные информационные модели</b>
<i>информационные модели - это модели, созданные на естественном языке (т.е. на любом языке общения между людьми: английском, русском, китайском, мальтийском и т.п.) в устной или письменной форме.</i>	<b>Формальные информационные модели</b> - это модели, созданные на формальном языке (т.е. научном, профессиональном или специализированном). Примеры формальных моделей: все виды формул, таблицы, графы, карты, схемы и т.д.

## 2. Подходы к построению и проектированию информационных систем.

Модели, применяемые на стадии конструирования, образуют метафору проектирования или подход к проектированию.

При проектировании информационных систем используют локальный или системный подходы.

Сущность **локального подхода** к проектированию состоит в последовательном наращивании задач, решаемых в системе. В этом случае проектирование информационной системы состоит из решения задач, ориентированных на удовлетворение потребностей конкретных подразделений или требований, связанных с реализацией конкретных условий. При этом данные организуют в отдельные логически структурированные файлы. Этот подход имеет серьезные недостатки:

- избыточность информации;
- противоречивость;
- недостаточная скорость обработки;
- отсутствие гибкости;
- низкая стандартизация программного обеспечения.

**Системный подход**, будучи общей методологической базой проектирования информационных систем, основан на концепции интеграции данных, которые описывают все сферы деятельности объекта информатизации. Этот подход предусматривает рассмотрение все элементов и составляющих процесса проектирования в их взаимосвязи, взаимозависимости и взаимном влиянии в интересах оптимального достижения как отдельных, так и общих целей создания информационной системы. Данный подход исходит из обязательной необходимости анализа элементов и составляющих процесса проектирования в их взаимосвязи на основе широкого использования современных методов исследования.

В зависимости от принципа рассмотрения, анализа и моделирования предметной области возможна конкретизация системного подхода. В этом случае на современном этапе можно выделить четыре основных подхода к проектированию:

- **структурный подход** (функционально-ориентированное проектирование), который использует структурные методы для построения функциональной, информационной и других моделей информационной системы;

- **блочнo-иерархический подход** к проектированию использует идеи декомпозиции сложных описаний объектов и соответственно средств их создания на иерархические уровни и аспекты, вводит понятие стиля проектирования (восходящее, нисходящее, смешанное), устанавливает связь между параметрами соседних иерархических уровней;

- **объектный подход** (объектно-ориентированное проектирование) предлагает набор объектных моделей для описания предметной области;

- **модельный подход (модельно-ориентированное проектирование)** основан на настройке и доработке типовой конфигурации информационной системы в среде специализированных инструментальных систем.

### 3. Понятие метода проектирования ИС, их классификация.

Класс и подкласс технологии проектирования	Методы проектирования АИС		
	По степени автоматизации проектных решений	По степени типизации проектных решений	По степени адаптивности проектных решений
<b>Каноническое проектирование</b>	Ручное проектирование	Оригинальное проектирование	Реконструкция
<b>Индустриальное типовое проектирование</b>	Ручное проектирование; автоматизированное проектирование	Типовое проектирование	Параметризация; реструктуризация модели (конфигурации АИС)
<b>Индустриальное автоматизированное проектирование</b>	Автоматизированное проектирование	Оригинальное проектирование; типовое проектирование	Реструктуризация модели (генерация АИС)

Метод проектирования в общем случае включает совокупность трёх составляющих:

- 1) пошаговая процедура, определяющая последовательность технологических операций проектирования;
- 2) критерии и правила, используемые для оценки результатов выполнения технологических операций;
- 3) нотации (графические и текстовые средства), используемые для описания проектируемой системы.

По подходу к автоматизации объекта методы проектирования различают:

- **метод «снизу - вверх»** - разработка подсистем (процедур, функций), в то время когда проработка общей схемы не закончилась, то есть разработка ведется от отдельных задач ко всей системе;

- **метод «сверху - вниз»** - разработка начинается с определения целей решения проблемы, после чего идет последовательная детализация. При нисходящем проектировании задача анализируется с целью определения возможности разбиения ее на ряд подзадач. Затем каждая из полученных подзадач также анализируется для возможного разбиения на подзадачи. Процесс заканчивается, когда подзадачу невозможно или нецелесообразно далее разбивать на подзадачи;

- **принципы «дуализма» и многокомпонентности** - подход к проектированию ИС заключается в сбалансированном сочетании двух предыдущих.

## 4. Основные принципы системного подхода к созданию ИС.

**Системный подход**, будучи общей методологической базой проектирования информационных систем, основан на концепции интеграции данных, которые описывают все сферы деятельности объекта информатизации. Этот подход предусматривает рассмотрение все элементов и составляющих процесса проектирования в их взаимосвязи, взаимозависимости и взаимном влиянии в интересах оптимального достижения как отдельных, так и общих целей создания информационной системы. Данный подход исходит из обязательной необходимости анализа элементов и составляющих процесса проектирования в их взаимосвязи на основе широкого использования современных методов исследования.

ГОСТ Р ИСО/МЭК 15288-2005, подробно рассматриваемый в разделе стандартов и посвященный процессам ЖЦ систем, не только представляет из себя хороший пример единого системного подхода, рассматривая процессы соглашения, предприятия, проекта и технические процессы с точки зрения предприятия, а не разработчиков, но и сам в приложении содержит крайне наглядные представления понятий «системы» и «среды использования». Для успешного применения подобного системного подхода необходимо уметь рассматривать системы не только «изнутри» или «снаружи», но и «сбоку».

В зависимости от принципа рассмотрения, анализа и моделирования предметной области возможна конкретизация системного подхода. В этом случае на современном этапе можно выделить четыре основных подхода к проектированию:

- **структурный подход** (функционально-ориентированное проектирование), который использует структурные методы для построения функциональной, информационной и других моделей информационной системы;

- **блочно-иерархический подход** к проектированию использует идеи декомпозиции сложных описаний объектов и соответственно средств их создания на иерархические уровни и аспекты, вводит понятие стиля проектирования (восходящее, нисходящее, смешанное), устанавливает связь между параметрами соседних иерархических уровней;

- **объектный подход** (объектно-ориентированное проектирование) предлагает набор объектных моделей для описания предметной области;

## 5. Понятие технологии проектирования. Требования к технологии проектирования.

**Технология проектирования** - совокупность концептуальных методов проектирования ИС, а также методов и средств организации (проектирования), управление процессом проектирования, а также модернизации и реинжиниринга ИС.

Выбор технологии проектирования осуществляется с учетом следующих **требований**:

1. возможность обеспечения соответствия, создаваемого с помощью конкретной технологии проекта требованиям заказчика;
2. способность выбираемой технологии обеспечивать минимальные трудовые и стоимостные затраты на проектирование и сопровождение проекта;
3. создание условий для повышения производительности труда проектировщика;
4. обеспечение надежности процесса проектирования и эксплуатации проекта;
5. простота ведения проектной документации.

Для конкретных классов технологии проектирования свойственно применение соответствующих средств проектирования информационных систем. Так, использование технологии канонического проектирования предполагает применение различных средств на традиционных носителях, в том числе нормативно-правовых документов (положений о структурном подразделении, должностных инструкций и т. п.), нормативно-технических документов (стандартов, руководящих документов и т. п.), систем классификации и кодирования информации, систем документации, моделей входных и выходных потоков информации и методик их анализа и др. При автоматизированном проектировании, наряду с вышеназванными средствами, могут быть использованы **разнообразные программные средства**, которые подразделяют на четыре группы:

- **операционные средства** (алгоритмические языки, макрогенераторы, генераторы программ типовых операций обработки данных, утилиты, библиотеки стандартных подпрограмм и классов объектов и др.);
- **прикладные программные средства общего назначения** (СУБД, текстовые и графические редакторы, табличные процессоры, методо-ориентированные пакеты прикладных программ, оболочки экспертных систем, интегрированные пакеты прикладных программ);
- **функциональные средства проектирования** (функциональные пакеты прикладных программ, типовые проекты и проектные решения);
- **средства автоматизации проектирования** (CASE-средства).

## 6. Классификация технологий проектирования ИС. Выбор технологии проектирования ИС.

Среди технологий проектирования ИС выделяют два основных класса:

- каноническая технология;
- индустриальная технология.

**Технология канонического проектирования** предполагает использование инструментальных средств универсальной компьютерной поддержки и предназначена для создания индивидуальных (оригинальных) проектов локальных ИС. При этом адаптация проектных решений возможна лишь путем перепрограммирования соответствующих программных модулей.

Организация канонического проектирования ИС ориентирована на использование главным образом каскадной модели жизненного цикла ИС. Стадии и этапы работы описаны в ГОСТ 34.601-90.

**Технологии индустриального проектирования** использует инструментальные средства специальной компьютерной поддержки для разработки проектов сложных интегрированных (корпоративных) ИС.

Индустриальная технология проектирования, в свою очередь, разбивается на два подкласса:

- **типовое** (параметрически-ориентированное или модельно-ориентированное) проектирование.

- **автоматизированное проектирование** (использование CASE- технологий).

Замечание: использование индустриальных технологий не исключает использования в отдельных случаях канонических

Для конкретных классов технологии проектирования свойственно применение соответствующих средств проектирования информационных систем. Так, использование технологии канонического проектирования предполагает применение различных средств на традиционных носителях, в том числе нормативно-правовых документов (положений о структурном подразделении, должностных инструкций и т. п.), нормативно-технических документов (стандартов, руководящих документов и т. п.), систем классификации и кодирования информации, систем документации, моделей входных и выходных потоков информации и методик их анализа и др. При автоматизированном проектировании, наряду с вышеназванными средствами, могут быть использованы **разнообразные программные средства**, которые подразделяют на четыре группы:

- **операционные средства** (алгоритмические языки, макрогенераторы, генераторы программ типовых операций обработки данных, утилиты, библиотеки стандартных подпрограмм и классов объектов и др.);
- **прикладные программные средства общего назначения** (СУБД, текстовые и графические редакторы, табличные процессоры, методо-ориентированные пакеты прикладных программ, оболочки экспертных систем, интегрированные пакеты прикладных программ);
- **функциональные средства проектирования** (функциональные пакеты прикладных программ, типовые проекты и проектные решения);
- **средства автоматизации проектирования** (CASE-средства).

## 7. Классификация средств проектирования ИС.

Под *средствами проектирования информационных систем* (СП ИС) будем понимать комплекс инструментальных средств, обеспечивающих в рамках выбранной методологии проектирования поддержку полного жизненного цикла (ЖЦ) ИС. Каждый этап ЖЦ характеризуется определенными задачами и методами их решения, исходными данными, полученными на предыдущем этапе, и результатами.

Основные признаки средств проектирования ИС:

- Инвариантность к объекту проектирования
- Охват всех этапов жизненного цикла
- Техническая, программная и информационная совместимость средств
- Простота применения и обучения
- Экономическая целесообразность

Два класса средств проектирования:

- Стандарты, регламентирующие процесс создания, документация
- Компьютерные средства:
  - Проектирование операций обработки информации
    - Алгоритмические языки
    - Библиотеки
    - Отладчики
    - Тесты
  - Средства проектирования отдельных комплексов ИС
    - Специальные пакеты мат. статистики
    - Графические редакторы
    - Редакторы текста
    - СУБД
    - Математического программирования
  - Средства автоматизированной (case) разработки этапов



## 8. Основные стадии жизненного цикла проектирования ИС.

Сами стадии жизненного цикла фиксированы и для различных отраслей человеческой деятельности, по сути, одинаковы:

- Замысел (планирование проекта).
- Анализ и постановка задачи.
- Проектирование.
- Разработка.
- Развертывание и внедрение.
- Эксплуатация.
- Поддержка.
- Модернизация.
- Утилизация.

Стадии жизненного цикла на основании ГОСТ 34.601-90

1. ФТ - Формирование требований к АС.	1.1. Обследование объекта и обоснование необходимости создания АС 1.2. Формирование требований пользователя к АС 1.3. Оформление отчета о выполненной работе и заявки на разработку АС (тактико-технического задания)
2. РК - Разработка концепции АС.	2.1. Изучение объекта 2.2. Проведение необходимых научно-исследовательских работ 2.3. Разработка вариантов концепции АС удовлетворяющей требованиям пользователя 2.4. Оформление отчета о выполненной работе
3. ТЗ - Техническое задание АС.	3.1. Разработка и утверждение технического задания на задание
4. ЭП - Эскизный проект.	4.1. Разработка предварительных проектных решений по системе и ее частям 4.2. Разработка документации на АС и ее части
5. ТП - Технический проект.	5.1. Разработка проектных решений по системе и ее частям 5.2. Разработка документации на АС и ее части 5.3. Разработка и оформление документации на поставку изделий для комплектования АС и/или технических требований (технических заданий) на их разработку 5.4. Разработка заданий на проектирование в смежных частях проекта объекта автоматизации
6. РД - Рабочая документация.	6.1. Разработка рабочей документации на систему и ее части 6.2. Разработка или адаптация программ
7. ВД - Ввод в действие.	7.1. Подготовка объекта автоматизации к вводу АС в действие; 7.2. Подготовка персонала 7.3. Комплектация АС поставляемыми изделиями (программными и техническими средствами, программно-техническими комплексами, информационными изделиями) 7.4. Строительно-монтажные работы 7.5. Пуско-наладочные работы 7.6. Проведение предварительных испытаний 7.7. Проведение опытной эксплуатации 7.8. Проведение приемочных испытаний
8. Сп - Сопровождение АС.	8.1. Выполнение работ в соответствии с гарантийными обязательствами; 8.2. Послегарантийное обслуживание.

## 9. Модели жизненного цикла ИС (основные).

Под моделью ЖЦ понимается структура, определяющая последовательность выполнения и взаимосвязи процессов, действий и задач, выполняемых на протяжении ЖЦ. Модель ЖЦ зависит от специфики ИС и специфики условий, в которых последняя создается и функционирует.

Наименование модели	Сущность	Условия применения	Недостатки
<b>Каскадная модель</b>	Предусматривает последовательный переход на каждый следующий этап после завершения предыдущего	Решение отдельных, не связанных задач, не требующее выполнения информационной интеграции и совместимости обеспечивающих подсистем	Большая длительность процесса проектирования применительно к крупным и сложным проектам
<b>Итерационная модель</b>	Предусматривает возможность итерационных возвратов на предыдущие этапы после выполнения очередного этапа	Решение задач, связанных с пересмотром ранее сформулированных требований, обуславливающих необходимость итерационных возвратов	Угроза рассогласования в выполненных проектных решениях и документации; может стать причиной необходимости перепроектирования всей системы
<b>Спиральная модель</b>	Предусматривает возможность последовательного углубления и конкретизации проектных решений имеющегося прототипа АИС на основе применения прототипной технологии (RAD-технологии)	Последовательная разработка общесистемных вопросов, а затем технологии решения конкретных задач на каждом витке спирали	Необходимость активного участия на всех этапах разработки конечных пользователей создаваемой системы Необходимость активного участия на всех этапах разработки конечных пользователей создаваемой системы

### 1) Модель Code&Fix

- 1я версия    a->b->b->c->d
- Модернизация о удовлетворения клиента
- Сопровождена
- Замена

Модель кодирования и устранения ошибок («проб и ошибок») (Build-and-fix, Gode and fix) Модель кодирования и устранения ошибок - самая простая модель жизненного цикла.

Эта модель не актуальна при профессиональной разработке программного обеспечения. Модель Build-and-fix хороша для небольших проектов, которые не требуют сопровождения, но абсолютно непригодна для корпоративных или вообще нетривиальных проектов объемом более 1000 строк.

### 2) Каскадная модель («водопад» - waterfall)

Первой моделью, характерной для 70-85 г.г. прошлого века, получившей широкую известность и структурирующей процесс разработки, является каскадная (однократный проход, водопадная или классическая модель). Она была создана после прошедшей в 1968 г. конференции NATO по вопросам науки и техники, где рассматривались подобные вопросы.

- Анализ a->b->c->d->e
- Проектирование
- Реализация
- Внедрение
- Сопровождение

### 3) Спиральная модель

- a. Анализ
- b. Требования
- c. Проектирование (по версиям)
- d. Разработка
- e. Тестирование

Спиральная стратегия (эволюционная или итерационная модель, автор Барри Боэм, 1988 г.) подразумевает разработку в виде последовательности версий, но в начале проекта определены не все требования. Требования уточняются в результате разработки версий.

Данная модель жизненного цикла характерна при разработке новаторских (нетиповых) систем. В начале работы над проектом у заказчика и разработчика нет четкого видения итогового продукта (требования не могут быть четко определены) или стопроцентной уверенности в успешной реализации проекта (риски очень велики). В связи с этим принимается решение разработки системы по частям с возможностью изменения требований или отказа от ее дальнейшего развития.

### 4) V-образная (шарнирная) модель

- a. Планирование и определение требований
- b. Анализ требований
- c. Проектирование архитектуры верхнего уровня
- d. Детальное проектирование
- e. Реализация
- f. Модульное тестирование
- g. Интеграция и тестирование
- h. Тестирование и проверки системы в целом
- i. Ввод в эксплуатацию и поддержка

V-образная модель была предложена для того, чтобы устранить недостатки каскадной модели, а название - V-образная, или шарнирная.

V-образная модель дала возможность значительно повысить качество ПО за счет своей ориентации на тестирование, а также во многом разрешила проблему соответствия созданного продукта выдвигаемым требованиям благодаря процедурам верификации и аттестации на ранних стадиях разработки (пунктирные линии на рисунке указывают на зависимость этапов планирования/постановки задачи и тестирования/приемки).

### 5) Инкрементная (поступательная) модель

- a. Анализ требований и предварительное тестирование
- b. Проектирование
- c. Реализация
- d. Тестирование
- e. Версия

Другими словами, инкрементная разработка представляет собой процесс частичной реализации всей системы и постепенного наращивания функциональных возможностей.

Данная модель жизненного цикла характерна при разработке сложных и комплексных систем, для которых имеется четкое видение (как со стороны заказчика, так и со стороны разработчика) того, что собой должен представлять конечный результат (информационная система). Разработка версиями ведется в силу разного рода причин:

- отсутствия у заказчика возможности сразу профинансировать весь дорогостоящий проект;
  - отсутствия у разработчика необходимых ресурсов для реализации сложного проекта в сжатые сроки;
  - требований поэтапного внедрения и освоения продукта конечными пользователями.
- Внедрение всей системы сразу может вызвать у ее пользователей неприятие и только «затормозить» процесс перехода на новые технологии.

## 10. Модели жизненного цикла ИС (модифицированные).

### 1) Поэтапная модель с промежуточным контролем (водоворот)

- a) Определение требований
- b) Проектирование системного и программного обеспечения
- c) Реализация и тестирование модулей
- d) Интеграция и тестирование модулей
- e) Эксплуатация и сопровождение

В 1970 г. каскадная модель была доработана Уинстом Ройсом с учетом взаимозависимости этапов и необходимости возврата на предыдущие ступени, что может быть вызвано, например, неполнотой требований или ошибками в формировании задания.

Практическое использование данной модели выявило множество ее недостатков, главный из которых состоял в том, что она больше подходит для традиционных видов инженерной деятельности, чем для разработки ПО. В частности, одной из самых больших проблем оказалась ее «предрасположенность» к возможным несоответствиям полученного в результате продукта и требований, которые к нему предъявлялись.

### 2) Структурная эволюционная модель быстрого прототипирования (Rapid Prototype Model)

Имеется первоначальная версия, которая используется для апробирования реализованных возможностей

- a) Разработка плана проекта
- b) Быстрый анализ (сопоставления прототипа с планом)
- c) Отдельные элементы на основании прототипа

**Достоинства:** Управление рисками, гибкое проектирование, участия заказчика на всех этапах жизненного цикла

**Недостатки:** Проблема прототипирования, заикливание, большое количество операций

### 3) Модель быстрой разработки приложений RAD (Rapid Application Development)

- a) Анализ и планирование требований
- b) Проектирование
- c) Построение
- d) Внедрение

Малая команда программистов, от 2 до 10 человек, короткий, четко разработанный график, повторяющийся цикл создания ПО, пока приложение не станет приемлемым.

### 4) Модифицированная спиральная модель (Win-Win)

Больше промежуточных фаз и более тесное участия заказчика с разработкой. Теория W.

### 5) Объектная модель ROP (Rational Objectory Process) методологии RUP (Rational Unified Process)

- a) Начало
- b) Уточнение
- c) Построение
- d) Внедрение

На прямую связано с UML. Итеративный процесс последовательного уточнения результатов.

Направлен на создание моделей. RUP - одна из спиральных методологий разработки программного обеспечения.

## 11. Соответствие информационных процессов, информационных технологий и средств их проектирования.

Процессы, связанные с поиском, хранением, передачей, обработкой и использованием информации, называются **информационными процессами**.

### **Хранение информации:**

- Носители информации.
- Виды памяти.
- Хранилища информации.
- Основные свойства хранилищ информации.

### **Обработка информации:**

- Общая схема процесса обработки информации.
- Постановка задачи обработки.
- Исполнитель обработки.
- Алгоритм обработки.
- Типовые задачи обработки информации.

### **Передача информации:**

- Источник и приемник информации.
- Информационные каналы.
- Роль органов чувств в процессе восприятия информации человеком.
- Структура технических систем связи.
- Что такое кодирование и декодирование.
- Понятие шума; приемы защиты от шума.
- Скорость передачи информации и пропускная способность канала.

### **Технические средства реализации информационных процессов.**

#### *Хранение информации.*

##### **Носители информации:**

- ОЗУ компьютера (оперативная память)
- Гибкие диски 3,5"
- Оптические диски CD, DVD и др.
- Жёсткие диски
- Переносные запоминающие устройства - *flash* и др.

#### *Передача информации:* источник, приёмник, канал

#### *Обработка информации:* компьютер и др.

## 12. Понятия системной инженерии, программной инженерии: связи, отличия, особенности.

**системная инженерия** — междисциплинарный подход, определяющий полный набор технических и управленческих усилий, которые требуются для того, чтобы преобразовать совокупность потребностей и ожиданий заказчика и имеющихся ограничений и эффективные решения и поддержать эти решения в течении их ЖЦ. ISO/IEC 24765

Рассматривает как деловые, так и технические потребности всех заинтересованных сторон с целью их всестороннего учета, и создания в результате качественной продукции и услуг, отвечающих потребностям заказчика.

**Программная инженерия** (англ. *software engineering*) — приложение систематического, дисциплинированного, измеримого подхода к развитию, функционированию и сопровождению программного обеспечения, а также исследованию этих подходов; то есть, приложение дисциплины инженерии к программному обеспечению (*ISO/IEC/IEEE 24765-2010*)

Программная инженерия:

- Определение и анализ требований к ПО
- Проектирование ПО
- Конструирование ПО
- Тестирование ПО
- Эксплуатация и сопровождение ПО
- Управления инженерной деятельностью по созданию ПО
- Процессы программной инженерии
- Методы и средства программной инженерии
- Качество ПО

Современный подход к созданию сложных систем анализирует **системную и программную инженерию** в неразрывном единстве. Именно **системная инженерия** рассматривается как дисциплина, которая закладывает и описывает ключевые принципы организации и управления деятельностью по созданию систем любого масштаба и назначения, развивает методологию и средства управления ЖЦ сложных систем, создает принципы и инструменты их разработки. При этом за **программной инженерией** оставляется ключевая роль в комплексной реализации системных решений более высокого уровня.

## 13. Принципы проектирования базовых информационных технологий.

Базовая информационная технология - это технология, ориентированная на определенную область применения. Предметом изучения излагаемого курса являются автоматизированные информационные технологии в управлении организационно-экономическими системами, создаваемыми при производстве материальных благ и услуг. Любая информационная технология складывается из взаимосвязанных информационных процессов, каждый из которых содержит определенный набор процедур, реализуемых с помощью информационных операций. Информационная технология выступает как система, функционирование каждого элемента которой подчиняется общей цели функционирования системы - получению качественного информационного продукта из исходного информационного ресурса в соответствии с поставленной задачей.

Как базовая информационная технология в целом, так и отдельные информационные процессы могут быть представлены тремя уровнями: концептуальным, логическим и физическим. Концептуальный уровень определяет содержательный аспект информационной технологии или процесса, логический - отображается формализованным (модельным) описанием, а физический уровень раскрывает программно-аппаратную реализацию информационных процессов и технологии.

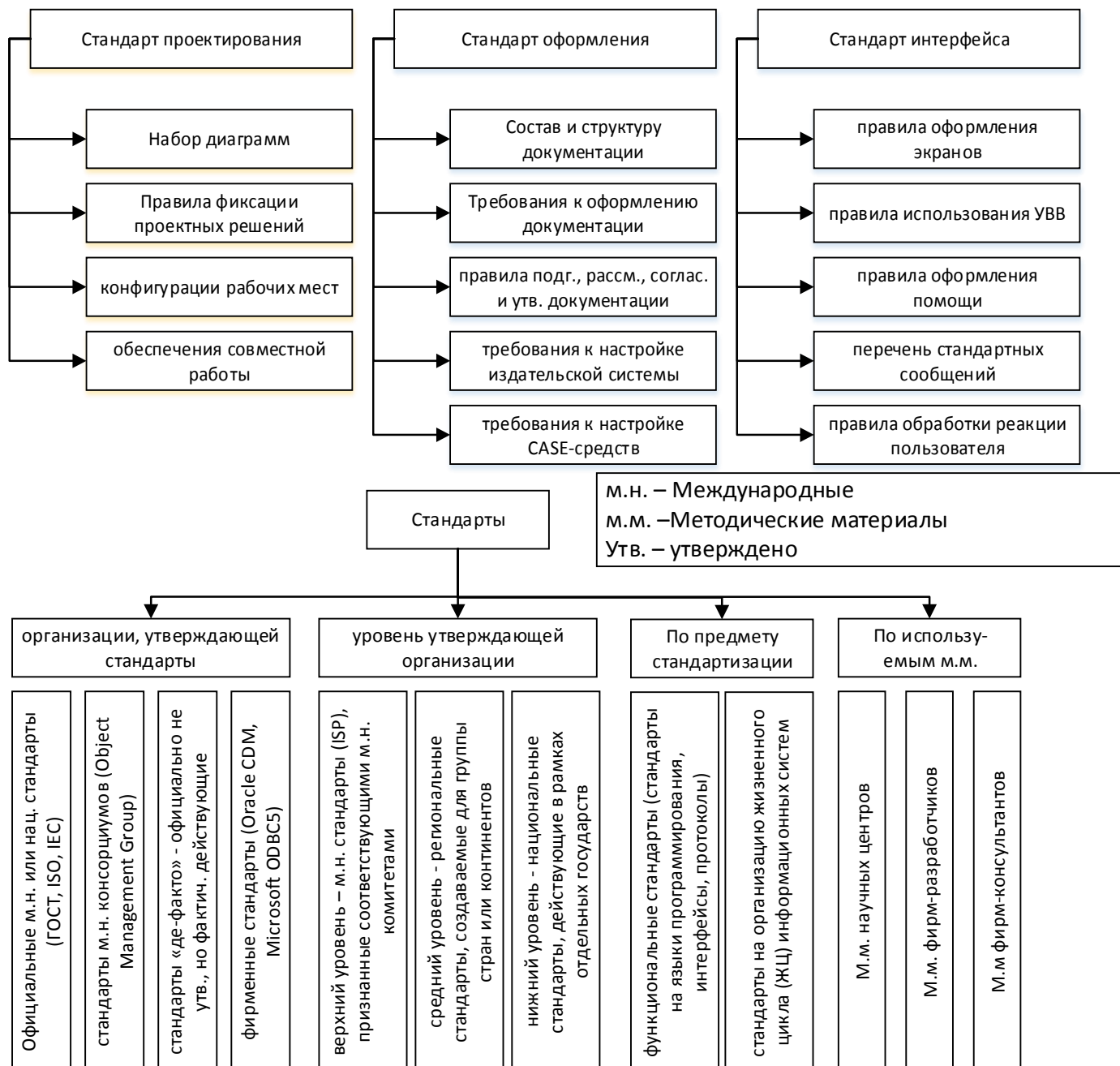
Базовые информационные технологии включают в себя следующие технологии:

- Мультимедийные технологии
- Автоматизация офиса
- Информационные технологии автоматизированного проектирования
- Информационные технологии в промышленности и экономике
- Технологии искусственного интеллекта
- CASE-технологии
- Геоинформационные технологии
- Статистические информационные технологии
- Информационная технология управления
- Информационные технологии в образовании
- Информационные технологии организационного управления (корпоративные информационные технологии)
- Телекоммуникационные технологии
- Технологии защиты информации
- Бухгалтерские информационные системы (БУИС)

## 14. Классификация стандартов на проектирование и разработку информационных систем.

Реальное применение любой технологии проектирования, разработки и сопровождения ИС в конкретной организации и конкретном проекте происходит при соблюдении всеми участниками проекта правил и соглашений, касающихся:

- проектирования; (создание и сопровождение ИС, быстрого проектирования приложений)
- оформления проектной документации;
- пользовательского интерфейса.



**Объектами стандартизации** при проектировании информационных систем являются:

- процесс (что делать?): ISO12207 [8], ISO15288 [11], IEEE1220 [12];
- практика (как следует делать?): ISO15504 [13];
- источник (какие данные использовать): ISO10303 [14];
- описание: IDEF [15], UML [16], IEEE 1471 [17]. Стандартизация информационных технологий и систем повышает их прибыльность за счет снижения затрат на создание и особенно модификацию.

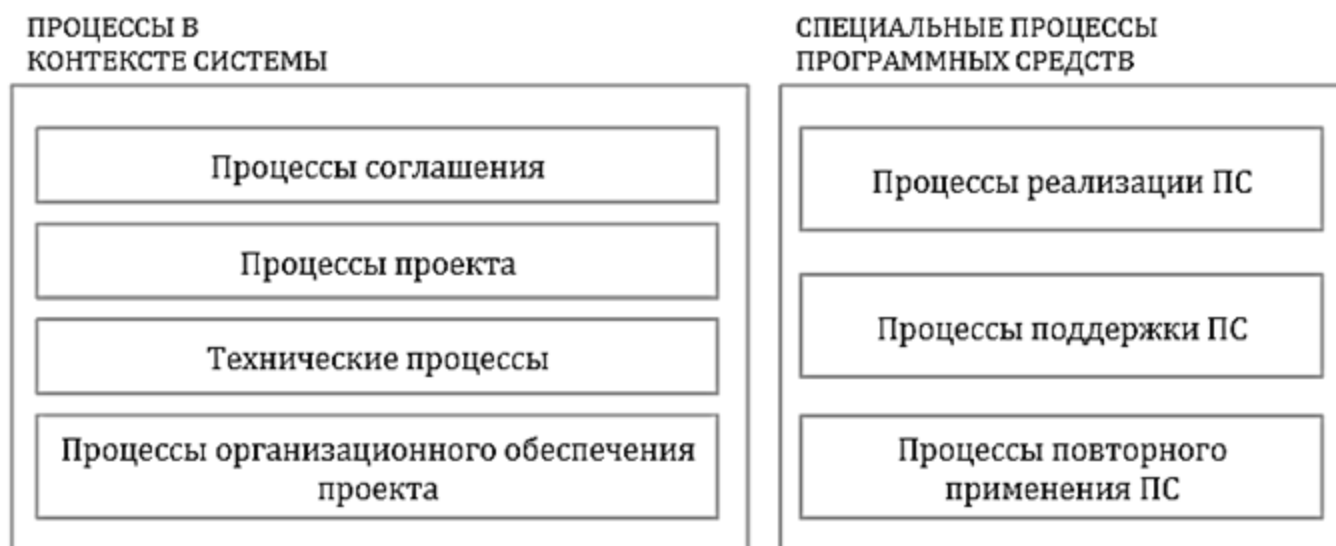


## 15.Международный стандарт ISO/IEC 12207:1995-08-01 (ГОСТ Р ИСО/МЭК 12207).

Международный стандарт: ISO/IEC 12207:2008 Information technology - Software life cycle processes (Информационные технологии. Процессы жизненного цикла программного обеспечения).

Российский аналог: ГОСТ Р ИСО/МЭК 12207-2010 «Информационная технология. Системная и программная инженерия. Процессы жизненного цикла программных средств».

Базируясь на процессном подходе, ISO 12207 определяет необходимость документирования основных результатов процесса, но не ограничивает их содержание и тем более последовательность, а также не противоречит применению итераций в разработке. Данный стандарт стал основой для дальнейшей детализации в некоторых методологиях разработки ПО (в частности, Rational Unified Process), однако сам по себе лишь устанавливает структуру основных, вспомогательных и организационных процессов ЖЦ программных средств, определяя необходимые в их рамках работы и задачи. Таким образом формируется единое понимание жизненного цикла (и единая терминология) между заказчиком, разработчиком / подрядчиком и другими стейкхолдерами. С другой стороны, ISO 12207:2008 рассматривает лишь программные средства и соответствующие организационные процессы, не рассматривая аппаратную составляющую.



В РФ был разработан и принят идентичный ISO 12207 стандарт ГОСТ Р ИСО/МЭК 12207-2010. Основной идеей разработчиков ГОСТ 12207 являлось создание единого общекорпоративного стандарта, которым было бы возможно воспользоваться при возникновении любой задачи из тех, которые описаны в документе (будь это обучение пользователей, поставка ПО или любая другая активность в рамках ЖЦ).

Стандарт предполагает, что процессы состоят из работ, для которых определены задачи (а также цели и результаты). Тем не менее, допускается адаптация процессов к особенностям организации (например, при больших масштабах проекта изменять состав определенных задач или работ). Как правило, это возможно сделать в рамках существующих на предприятии процессов.

Приложения стандарта содержат (помимо эталонной модели процессов, их описаний и видов, а также истории разработки) отдельно выделенный процесс адаптации. Приведенные в нем рекомендации по переходу от стандарта к реалиям определенного предприятия в основном концентрируются на выборе из всего приведенного множества процессов тех работ, которые необходимы для реализации конкретного программного проекта. Однако практические рекомендации по организации внедрения ГОСТ 12207-2010 остаются за границами самого документа.

## 16.Международный стандарт ISO/IEC 15288:2002 (ГОСТ Р ИСО/МЭК 15288-2005).

Международный стандарт: ISO/IEC 15288:2005 Systems engineering. System life cycle processes (Системотехника. Процессы жизненного цикла системы).

Российский аналог: ГОСТ Р ИСО/МЭК 15288-2005 Информационная технология. Системная инженерия. Процессы жизненного цикла систем.

Достаточно «молодой» стандарт системной инженерии (впервые представленный в 2002 году), ISO/IEC 15288 фокусируется на вопросах жизненного цикла системного уровня, в особенности тейлоринге (tailoring) - по сути, настройке и адаптации ЖЦ к конкретным требованиям и ограничениям.

В отличие от рассмотренного ранее стандарта, ISO 15288 распространяется на системы в целом, охватывая такие их элементы, как: «технические средства, программные средства, люди, процессы (например, процесс оценки), процедуры (например, инструкции оператора), основные средства и природные ресурсы (например, вода, объекты живой природы, минералы)» [50, с. 4]. Согласно данному стандарту, любой процесс ЖЦ может быть начат в любой момент, без ограничения порядка использования и последовательности (в том числе, параллельном выполнении нескольких процессов).

Важно также отметить высокий уровень абстракции ISO 15288 в сравнении с ISO 12207, так как данный стандарт не приводит ролей, конечных результатов в виде списка выходных документов, либо же состава работ, лишь оставаясь на уровне концепции.



Очевидно, что формулировки «... обеспечивать поддержку инфраструктуры ресурсов...» или «проявлять заботу о персонале.» не являются очень конкретными и могут толковаться по-разному. В связи с этим, важно иметь поддержку высшего руководства для использования данного стандарта в проекте создания и эксплуатации системы. При этом особенность ISO 15288 в том, что использоваться он может как со стороны заказчика, так и со стороны исполнителя.

Стандарт содержит четыре основные группы процессов (предприятия, соглашения, проекта и технические), описывающие соответственно вспомогательные корпоративные процессы, взаимодействие с контрагентами, управление проектом и саму реализацию системы.

Важной положительной чертой стандарта является его связь с бизнесстороной проекта создания системы за счет групп процессов предприятия и соглашения. Благодаря наличию подобных

разделов стандарта появляется связь с соответствующими корпоративными функциями и для бизнеса становится более понятным место процессов ЖЦ в процессах организации в целом.

## 17.Комплекс стандартов ГОСТ 34.

Группа 0	ГОСТ 34.003-90	Автоматизированные системы. Термины и определения
Группа 2	ГОСТ 34.201-89	Виды, комплектность, обозначения документов при создании АС
Группа 6	ГОСТ 34.601-90	Стадии создания АС
	ГОСТ 34.602-89	ТЗ на создание АС
Руководящий документ	РД 50-34.698-90	Требования к содержанию документов

По ГОСТ 34.601-90

1. ФТ - Формирование требований к АС.	1.1. Обследование объекта и обоснование необходимости создания АС 1.2. Формирование требований пользователя к АС 1.3. Оформление отчета о выполненной работе и заявки на разработку АС (тактико-технического задания)
2. РК - Разработка концепции АС.	2.1. Изучение объекта 2.2. Проведение необходимых научно-исследовательских работ 2.3. Разработка вариантов концепции АС удовлетворяющей требованиям пользователя 2.4. Оформление отчета о выполненной работе
3. ТЗ - Техническое задание АС.	3.1. Разработка и утверждение технического задания на задание
4. ЭП - Эскизный проект.	4.1. Разработка предварительных проектных решений по системе и ее частям 4.2. Разработка документации на АС и ее части
5. ТП - Технический проект.	5.1. Разработка проектных решений по системе и ее частям 5.2. Разработка документации на АС и ее части 5.3. Разработка и оформление документации на поставку изделий для комплектования АС и/или технических требований (технических заданий) на их разработку 5.4. Разработка заданий на проектирование в смежных частях проекта объекта автоматизации
6. РД - Рабочая документация.	6.1. Разработка рабочей документации на систему и ее части 6.2. Разработка или адаптация программ
7. ВД - Ввод в действие.	7.1. Подготовка объекта автоматизации к вводу АС в действие; 7.2. Подготовка персонала 7.3. Комплектация АС поставляемыми изделиями (программными и техническими средствами, программно-техническими комплексами, информационными изделиями) 7.4. Строительно-монтажные работы 7.5. Пуско-наладочные работы 7.6. Проведение предварительных испытаний 7.7. Проведение опытной эксплуатации 7.8. Проведение приемочных испытаний
8. Сп - Сопровождение АС.	8.1. Выполнение работ в соответствии с гарантийными обязательствами; 8.2. Послегарантийное обслуживание.

## 18.Фирменные стандарты (Oracle CDM, MSF и др.).

- **IBM (Rational Unified Process, RUP)**-ориентирован на пользователя, предполагает итеративную разработку, большое внимание архитектуре, к потенциальным рискам. Управление требованиями, компонентный подход. Визуально моделирование. Поддержка уровня качества. Мониторинг изменений.
- **Microsoft (Microsoft Solution Framework, MSF)**-итеративность. Формирование в качестве результата IT-решение. Спиралевидный цикл с вехами.
- **Microsoft Dynamics Sure Step и Microsoft Business Solutions Partner Methodology- MS** Dynamics Sure Step представляет собой комплексную методологию внедрения, оптимальную для программных продуктов Dynamics CRM, AX, NAV за счет подробных шаблонов и схем процессов, а также средств редактирования для адаптации методологии к специфике предприятия и отрасли. Методология определяет необходимые шаги каждого этапа внедрения продуктов MS Dynamics как в виде иерархической структуры работ, так и в виде графических схем процессов MS Visio.

### Oracle (Oracle Method)

комплекс методов, охватывающий большинство процессов ЖЦ ПО [40]. В состав комплекса входят:

- CDM (Custom Development Method) - разработка прикладного ПО;
- PJM (Project Management Method) - управление проектом;
- AIM (Application Implementation Method) - внедрение прикладного ПО;
- BPR (Business Process Reengineering) - реинжиниринг бизнес-процессов;
- OCM (Organizational Change Management) - управление изменениями, и др.

Custom Development Method (методика Oracle) по разработке прикладных информационных систем - технологический материал, детализированный до уровня заготовок проектных документов, рассчитанных на использование в проектах с применением Oracle.

В соответствии с этими факторами в CDM выделяются два основных подхода к разработке:

- классический подход (Classic);
- подход быстрой разработки (Fast Track).

**Классический подход (Classic)** применяется для наиболее сложных и масштабных проектов, он предусматривает последовательный и детерминированный порядок выполнения задач. Для таких проектов характерно большое количество реализуемых бизнес-правил, распределенная архитектура, критичность приложения. Применение классического подхода также рекомендуется при нехватке опыта у разработчиков, неподготовленности пользователей, нечетко определенной задаче. Продолжительность таких проектов от 8 до 36 месяцев.

**Подход быстрой разработки (Fast Track)** в отличие от каскадного классического, является итерационным и основан на методе DSDM (Dynamic Systems Development Method). В этом подходе четыре этапа - стратегия, моделирование требований, проектирование и генерация системы и внедрение в эксплуатацию. Подход используется для реализации небольших и средних проектов с несложной архитектурой системы, гибкими сроками и четкой постановкой задач. Продолжительность проекта от 4 до 16 месяцев.

### MSF

Microsoft Solutions Framework (модель разработки приложений Microsoft) - это набор концепций и рекомендуемых моделей, которые позволяют разрабатывать и внедрять информационные системы на основе технологий и инструментальных средств Microsoft. Многие концепции MSF хорошо известны. MSF является одной из интерпретаций спиральной (циклической) модели разработки приложений и базируется на практических результатах организации распределенных вычислений и применения технологий «клиент-сервер» компании Microsoft, ее партнеров и заказчиков [38].

Процесс MSF ориентирован на «вехи» (milestones) - ключевые точки проекта, характеризующие достижение в его рамках какого-либо существенного (промежуточного либо конечного) результата. Этот результат может быть оценен и проанализирован, что подразумевает ответы на вопросы: «Пришла ли проектная группа к однозначному пониманию целей и рамок проекта?», «В достаточной степени ли готов план действий?», «Соответствует ли продукт утвержденной спецификации?», «Удовлетворяет ли решение нужды заказчика?» и т. д.

Особенностями модели процессов MSF являются:

- подход, основанный на фазах и вехах;
- итеративный подход;
- интегрированный подход к созданию и внедрению решений.



Кроме этого, существует большое количество промежуточных вех, которые показывают достижение в ходе проекта определенного прогресса и расчлняют большие сегменты работы на меньшие, обозримые участки. Для каждой фазы модели процессов MSF определяет:

- что (какие артефакты) является результатом этой фазы;
- над чем работает каждый из ролевых кластеров на этой фазе

## 19. Понятие методологии проектирования информационных систем.

**Методология проектирования информационных систем** описывает процесс создания и сопровождения систем в виде жизненного цикла ИС, представляя его как некоторую последовательность стадий и выполняемых на них процессов.

В настоящее время существует ряд общих методологий разработки ИС. Главное в них - единая дисциплина работы на всех этапах жизненного цикла системы, учет критических задач и контроль их решения, применение развитых инструментальных средств поддержки процессов анализа, проектирования и реализации ИС.

Основными задачами, решение которых должна обеспечивать методология создания информационных систем, являются следующие:

- обеспечение создания информационных систем, отвечающих целям и задачам предприятия и соответствующих предъявляемым к ним требованиям;

- гарантия создания системы с заданными параметрами в течение заданного времени в рамках оговоренного заранее бюджета;
- простота сопровождения, модификации и расширения системы с целью обеспечения ее соответствия изменяющимся условиям работы предприятия;
- обеспечение создания информационных систем, отвечающих требованиям открытости, переносимости и масштабируемости;
- возможность использования в создаваемой системе разработанных ранее средств информационных технологий (программного обеспечения, баз данных, средств вычислительной техники, телекоммуникаций).

***Замечание: именно методология определяет, какие средства будут применяться для разработки программного обеспечения и, во многом, рекомендует, какой технологический подход будет при этом использован.***



## 20.Методологии структурного подхода (анализа).

Все наиболее распространенные методологии структурного подхода базируются на ряде **общих принципов**. В качестве двух базовых принципов используются следующие:

- принцип «разделяй и властвуй» - принцип решения сложных проблем путем их разбиения на множество меньших независимых задач, легких для понимания и решения;
- принцип иерархического упорядочивания - принцип организации составных частей проблемы в иерархические древовидные структуры с добавлением новых деталей на каждом уровне.
- Выделение двух базовых принципов не означает, что остальные принципы являются второстепенными, поскольку игнорирование любого из них может привести к непредсказуемым последствиям (в том числе и к провалу всего проекта). Основными из этих принципов являются следующие:
- принцип абстрагирования - заключается в выделении существенных аспектов системы и отвлечения от несущественных;
- принцип формализации - заключается в необходимости строгого методического подхода к решению проблемы;
- принцип непротиворечивости - заключается в обоснованности и согласованности элементов;
- принцип структурирования данных - заключается в том, что данные должны быть структурированы и иерархически организованы.

В структурном анализе используются в основном две группы средств, иллюстрирующих функции, выполняемые системой и отношения между данными. Каждой группе средств соответствуют определенные виды моделей (диаграмм), наиболее распространенными среди которых являются следующие:

- SADT (Structured Analysis and Design Technique) модели и соответствующие функциональные диаграммы;
- DFD (Data Flow Diagrams) диаграммы потоков данных;
- ERD (Entity-Relationship Diagrams) диаграммы "сущность-связь".

Одними из самых известных и широко используемых методологий в области моделирования бизнес-процессов являются методологии семейства IDEF. Семейство IDEF появилось в конце 60-х гг. XX в. под названием SADT (Structured Analysis and Design Technique). В настоящее время оно включает следующие стандарты:

- **IDEF0** - методология функционального моделирования. Используется для создания функциональной модели, отображающей структуру и функции системы, а также потоки информации и материальных объектов, связывающих эти функции;
- **IDEF1** - методология моделирования информационных потоков внутри систем, позволяющая отображать их структуру и взаимосвязи. Методология применяется для построения информационной модели, отображающей структуру и содержание информационных потоков, необходимых для поддержки функций системы;
- **IDEF1X (IDEF1X Extended)** - методология построения реляционных информационных структур. IDEF относится к типу методологий «сущность-связь» и, как правило, используется для моделирования реляционных баз данных, имеющих отношение к рассматриваемой системе;
- **IDEF2** - методология динамического моделирования развития систем, которая позволяет построить динамическую модель меняющихся во времени поведения функций, информации и ресурсов системы. В настоящее время известны алгоритмы и их компьютерные реализации, позволяющие превращать набор статических диаграмм IDEF0 в динамические модели, построенные на базе «раскрашенных сетей Петри» (CPN - Color Petri Nets);
- **IDEF3** - методология документирования процессов, происходящих в системе. С помощью IDEF3 описываются сценарий и последовательность операций для каждого процесса. Функция в диаграмме IDEF3 может быть представлена в виде отдельного процесса средствами **IDEF3**;
- **IDEF4** - методология построения объектно-ориентированных систем. Средства IDEF4 позволяют наглядно отображать структуру объектов и заложенные принципы их взаимодействия,

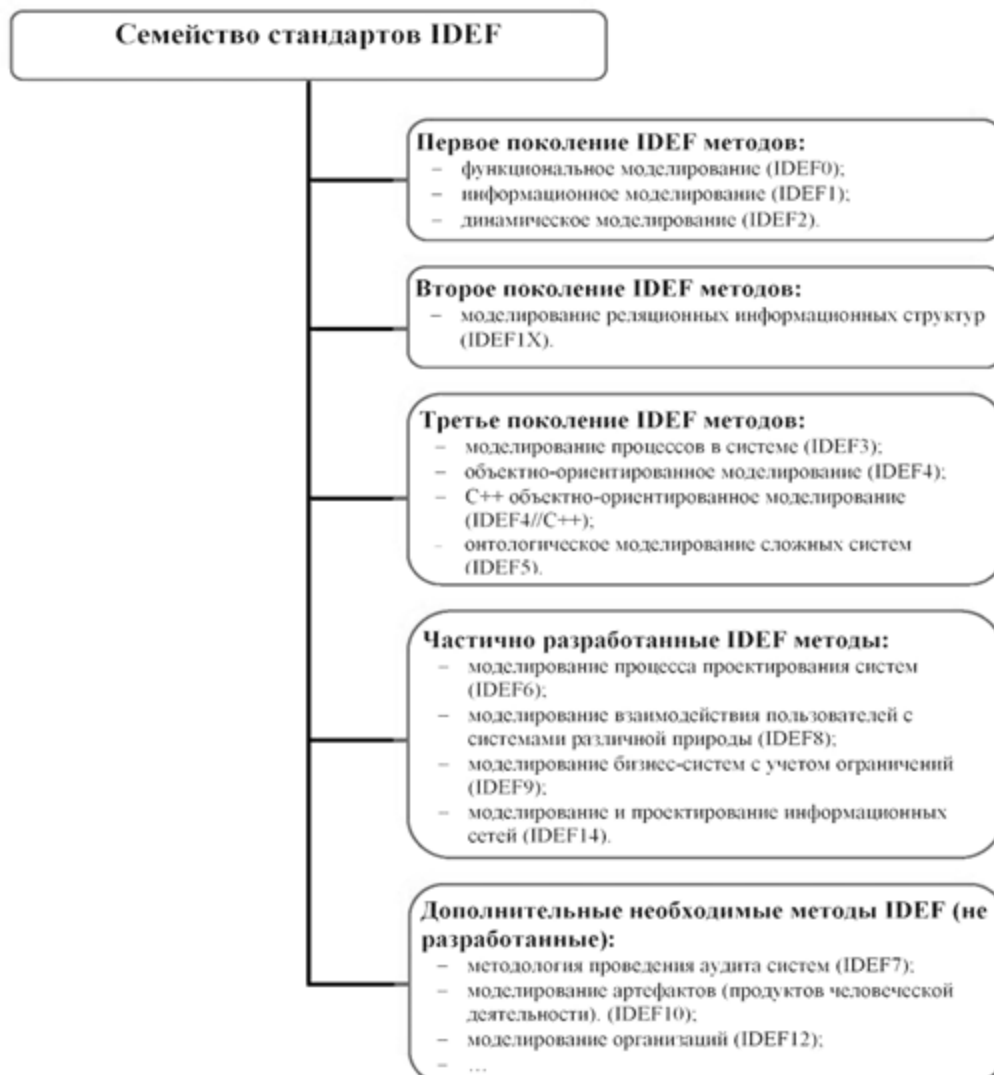


позволяя тем самым анализировать и оптимизировать сложные объектно-ориентированные системы;

- **IDEF5** - методология онтологического исследования сложных систем. С помощью этой методологии онтология системы описывается при помощи определенного словаря терминов и правил, на основе которых могут быть сформированы достоверные утверждения о состоянии рассматриваемой системы в некоторый момент времени. На основе этих утверждений формируются выводы о дальнейшем развитии системы и производится ее оптимизация;
  - **IDEF6 (Design Rational Capture)** - метод рационального представления процесса проектирования информационных систем, позволяющий обосновать необходимость проектируемых моделей, выявить причинно-следственные связи и отразить это в итоговой документации системы;
  - **IDEF8 (User Interface Modeling)** - Human - System Interaction Design Method - метод проектирования взаимодействия пользователей с системами различной природы (не обязательно информационно-вычислительными).
  - **IDEF9 (Business Constraint Discovery Method)** - метод изучения и анализа бизнес-систем в терминах «ограничений». Ограничения инициируют результат, руководят и ограничивают поведение объектов и агентов (автономных программных модулей) для выполнения целей или намерений системы.
  - **IDEF14 (Network Design Method)** - метод проектирования вычислительных сетей, позволяющий устанавливать требования, определять сетевые компоненты, анализировать существующие сетевые конфигурации и формулировать желаемые характеристики сети.
- Анонсированные корпорацией KBSI (Knowledge Based System Inc.) методы IDEF7 (Information System Audit Method), IDEF10 (Information Artifact Modeling) и IDEF12 (Organization Design) не получили дальнейшего развития.

С помощью методологий семейства IDEF можно эффективно отображать и анализировать модели

широкого  
сложных  
времени



деятельности  
спектра  
систем. К  
настоящему  
наибольшее

распространение и применение 15 имеют методологии IDEF0 и IDEF1 (IDEF<sup>^</sup>), получившие в США статус федеральных стандартов.

## 21.Методология моделирования функциональной структуры объектов – SADT. Методология моделирования данных –ERD (Entity-Relationship Diagrams) (case-метод Баркера).

Наиболее распространенной методологией моделирования данных являются диаграммы «сущность-связь» - ERD (Entity-Relationship Diagrams). С их помощью определяются важные для предметной области объекты (сущности), их свойства (атрибуты) и отношения друг с другом (связи). ERD непосредственно используются для проектирования реляционных баз данных.

Нотация ERD была впервые введена П. Ченом (Chen) и получила дальнейшее развитие в работах Баркера. Она включает в себя:

- сущности, которые являются существительными и представляет собой множество экземпляров реальных или абстрактных объектов (людей, событий, состояний, идей, предметов и т. п.), обладающих общими атрибутами или характеристиками;
- отношения, в самом общем виде представляют собой связи между двумя и более сущностями. Именованное отношения осуществляется с помощью грамматического оборота глагола;
- связи, каждая из которых соединяет сущность и отношение и может быть направлена только от отношения к сущности. Значение связи характеризует ее тип (1:1 - один-к-одному; 1:n - один-к-многим; m:n - многие-к-многим). Связь соединяется с ассоциируемыми сущностями линиями. Возле каждой сущности на линии, соединяющей ее со связью, цифрами указывается класс принадлежности;
- атрибуты, которые являются прилагательными или модификаторами. Каждая сущность обладает одним или несколькими атрибутами, которые однозначно идентифицируют каждый экземпляр сущности. При этом любой атрибут может быть определен как ключевой. Детализация сущности осуществляется с использованием диаграмм атрибутов, которые раскрывают ассоциированные сущностью атрибуты.

ERD-диаграмма, рис.1.18, позволяет рассмотреть систему целиком и выяснить требования, необходимые для ее разработки, касающиеся хранения информации.

ERD-диаграммы можно подразделить на отдельные части, соответствующие отдельным задачам, решаемым проектируемой системой. Это позволяет рассматривать систему с точки зрения функциональных возможностей, делая процесс проектирования управляемым.

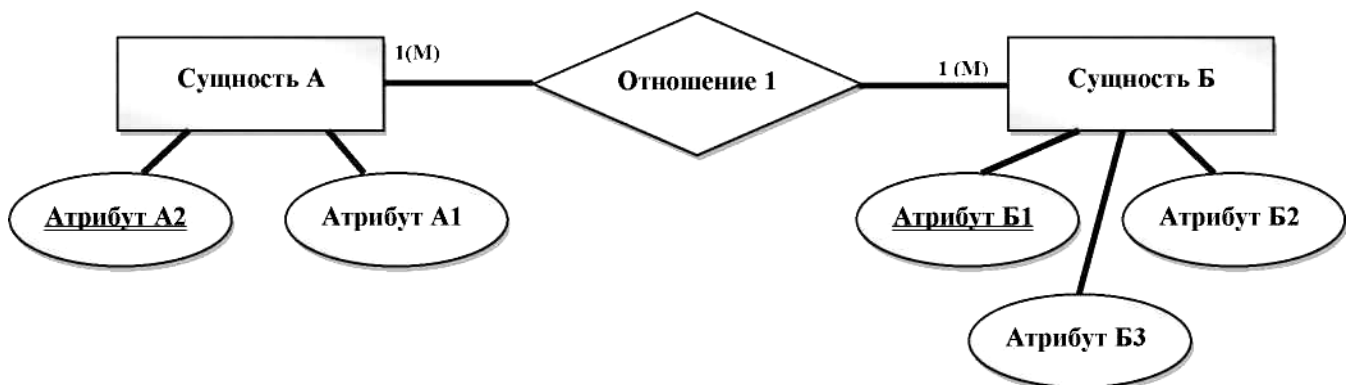
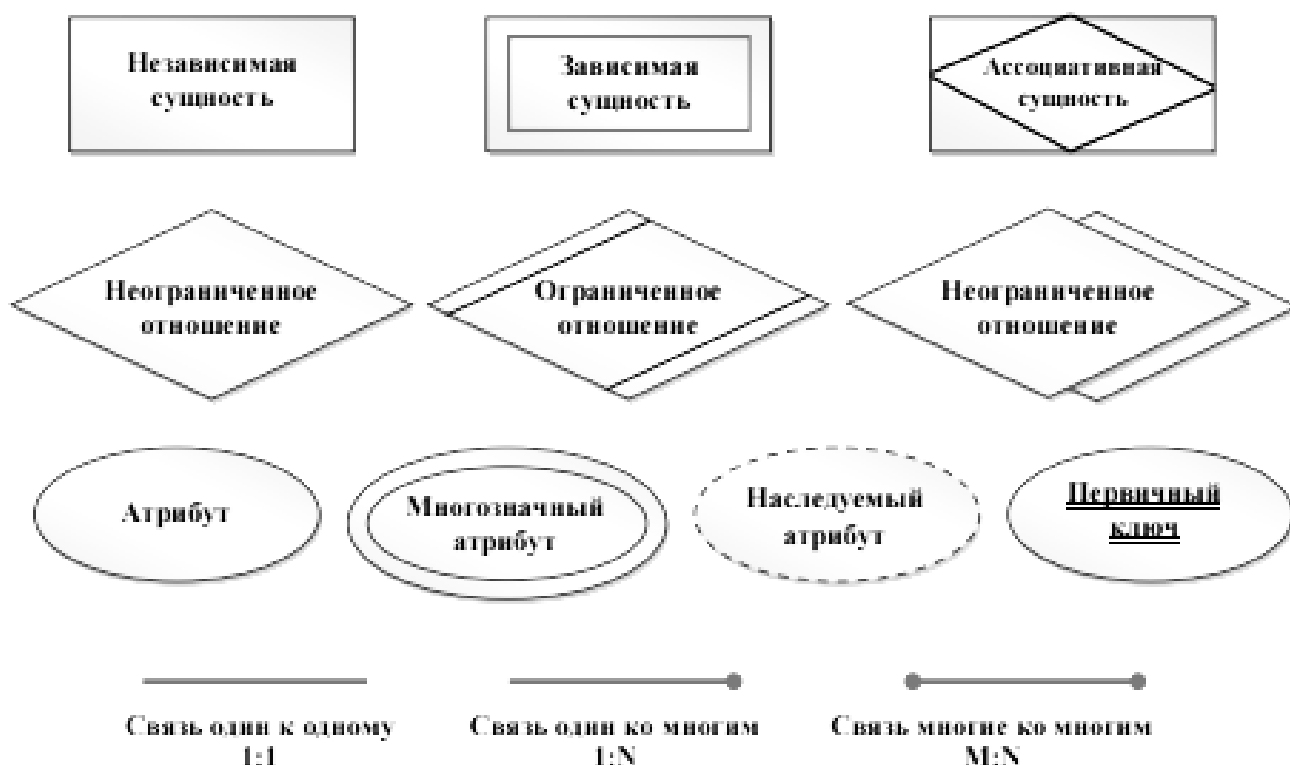


Рис.1.18. Пример ER-диаграммы в нотации П. Чена.

Существуют и другие нотации для представления ER-диаграмм: нотация Мартина, нотация IDEF1X (см. п. 1.8.1.6), нотация Баркера.



### Этапы построения ERD-диаграммы

На первом шаге производится определение сущностей. Для построения ER-диаграммы «с нуля» производится анализ предметной области и выделяются информационные объекты проектируемой системы, другими словами составляется список (пул) потенциальных сущностей (как правило, выделяются все существительные в текстовом описании предметной области). В случае если имеется DFD-диаграмма, то в этом случае в качестве сущностей можно взять названия интерфейсных дуг.

На втором шаге необходимо описать каждый информационный объект набором характеристик (атрибутов), которые представляют важность с точки зрения выполняемых системой функций, то есть из списка потенциальных сущностей выделяются сущности, а остальное преобразуется в атрибуты сущностей.

На третьем шаге устанавливаются отношения и связи между сущностями по описанию предметной области на естественном языке. Определяются виды отношений и типы связей.

На четвертом шаге из списка атрибутов выделить атрибуты, способные однозначно идентифицировать экземпляры сущности, то есть определить первичные ключи. Корректируются

На пятом шаге строится ER-диаграмма,

## 22. Методология моделирования работы в реальном времени—STD.

Методология STD (State Transition Diagram) предназначена для моделирования аспектов функционирования системы, зависящих от времени или реакции на события (так называемая работа в реальном времени).

Диаграммы переходов состояний STD обычно используются для описания отношения между входными и выходными управляющими потоками на управляющем процессе-предке и позволяют осуществлять декомпозицию управляющих процессов.

Такие диаграммы позволяют осуществить декомпозицию управляющих процессов, происходящих в системе, и описать отношение между управляющими потоками. С помощью диаграмм перехода состояний можно моделировать последующее функционирование системы исходя из предыдущих и текущего состояний.

Моделируемая система в любой заданный момент времени находится точно в одном из конечного множества состояний. С течением времени она может изменить свое состояние, при этом переходы между состояниями должны быть точно определены. Для перехода в состояние нужно какое-либо условие - условие перехода. Оно может быть информационным (условие появления информации) или временным. STD состоит из следующих объектов.

**Состояние** - рассматривается как устойчивое значение некоторого свойства в течение определенного времени. Начальное состояние - узел STD, являющийся стартовой точкой для начального системного перехода. STD имеет ровно одно начальное состояние, соответствующее состоянию системы после ее инсталляции, но перед началом реальной обработки, а также любое (конечное) число завершающих состояний;

**Переход** определяет перемещение моделируемой системы из одного состояния в другое. При этом имя перехода идентифицирует событие, являющееся причиной перехода и управляющее им. Это событие обычно состоит из управляющего потока (сигнала), возникающего как во внешнем мире, так и внутри моделируемой системы при выполнении некоторого условия. Управляющий поток - это «трубопровод», через который проходит управляющая информация.

Имеются следующие типы управляющих потоков:

- Т-поток (trigger flow) - поток управления процессом, который может вызвать выполнение процесса. При этом процесс включается одной короткой операцией;
- А-поток (activator flow) - поток управления процессом, который может изменять выполнение отдельного процесса. Используется для обеспечения непрерывности выполнения процесса до тех пор, пока поток «включен», с «выключением» потока выполнение процесса завершается;
- ED-поток (enable/disable flow) - поток управления процессом, который может переключать выполнение отдельного процесса. Течение по Е-линии вызывает выполнение процесса, которое продолжается до тех пор, пока не возбуждается течение по D-линии. Можно использовать 3 типа таких потоков: Е-поток, D-поток, Е/D-поток.

**Условие** представляет собой событие (или события), вызывающее переход и идентифицируемое именем перехода. Если в условии участвует входной управляющий поток управляющего процесса-предка, то имя потока должно быть заключено в кавычки, например. Кроме условия с переходом может связываться действие или ряд действий, выполняющихся, когда переход имеет место.

**Действие** - это операция, которая может иметь место при выполнении перехода. Если действие необходимо для выбора выходного управляющего потока, то имя этого потока должно заключаться в кавычки. Для спецификации А-, Т-, Е/D-потоков имя запускаемого или переключаемого процесса также должно заключаться в кавычки.

Фактически **условие** есть некоторое внешнее или внутреннее событие, которое система способна обнаружить и на которое она должна отреагировать определенным образом, изменяя свое **состояние**. При изменении состояния система обычно выполняет одно или более **действий** (производит вывод, выдает сообщение на терминал, выполняет вычисления). Таким образом, действие представляет собой отклик, посылаемый во внешнее окружение, или вычисление, результаты которого запоминаются в системе (обычно в хранилищах данных на DFD), для того, чтобы обеспечить реакцию на некоторые из планируемых в будущем событий.

На STD **состояния** представляются узлами, а **переходы** - дугами. **Условия** (по-другому называемые стимулирующими событиями) идентифицируются именем перехода и возбуждают выполнение перехода. **Действия** или отклики на события привязываются к переходам и записываются под соответствующим условием.

Начальное состояние на диаграмме должно иметь входной переход, изображаемый потоком из подразумеваемого стартового узла (иногда этот стартовый узел изображается небольшим квадратом и привязывается к входному состоянию). При построении STD рекомендуется следовать нижеперечисленным правилам:

- строить STD на как можно более высоком уровне детализации DFD;
- строить как можно более простые STD;
- по возможности детализировать STD;
- использовать те же принципы именований состояний, событий и действий, что и при именовании процессов и потоков.

Применяются **два способа построения STD**.

**Первый способ** заключается в идентификации всех возможных состояний и дальнейшем исследовании всех не бессмысленных связей (переходов) между ними.

По **второму способу** сначала строится начальное состояние, затем следующие за ним и т.д.

Результат (оба способа) - предварительная STD, для которой затем осуществляется контроль состоятельности, заключающийся в ответе на следующие вопросы:

- все ли состояния определены и имеют уникальное имя?
- все ли состояния достижимы?
- все ли состояния имеют выход?
- (для каждого состояния) реагирует ли система соответствующим образом на все возможные условия (особенно на ненормальные)?
- все ли входные (выходные) потоки управляющего процесса отражены в условиях (действиях) на STD?

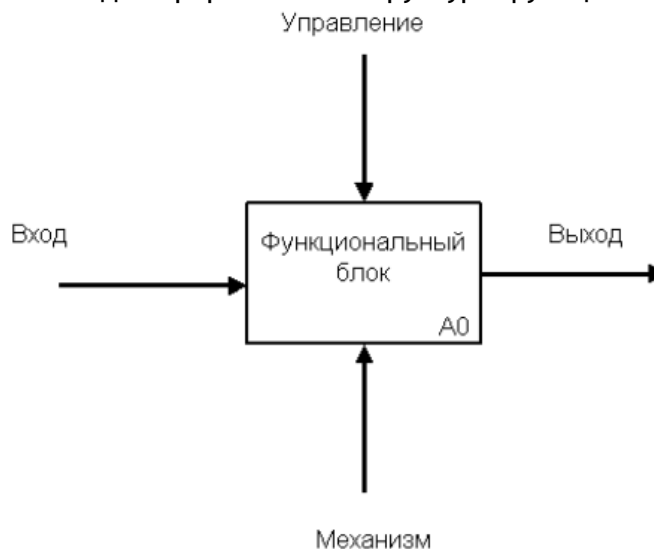
В ситуации, когда число состояний и/или переходов велико, для проектирования спецификаций управления могут использоваться **таблицы** и **матрицы переходов состояний**. Обе эти нотации позволяют зафиксировать ту же самую информацию, что и диаграммы переходов состояний, но в другом формате.

Первая колонка таблицы содержит список всех состояний проектируемой системы, во второй колонке для каждого состояния приведены все условия, вызывающие переходы в другие состояния, а в третьей колонке - совершаемые при этих переходах действия. Четвертая колонка содержит соответствующие имена состояний, в которые осуществляется переход из рассматриваемого состояния при выполнении определенного условия.

## 23.Методология функционального моделирования процессов –IDEF0.

### Характеристика диаграмм. Типы взаимосвязей между блоками.

IDEF0 представляет процесс в виде иерархической структуры функциональных блоков



Функциональный блок представляет собой функцию (например, «Производить услуги», «Разработать план производства», «Выставить счет»), к которой относятся интерфейсные дуги четырех типов (Вход, Выход, Управление, Механизм). Входом и выходом функции могут являться как различные реальные объекты (детали, материалы), так и потоки информации (данные о заказах, аналитика продаж). Управление определяет все стандарты, методологии, распоряжения и прочие регламентирующие документы, а Механизм-поддерживающие выполнение функции информационные системы, сотрудники и ресурсы. IDEF0 в обязательном порядке требует указания как минимум одной управляющей и одной исходящей интерфейсной дуги, что объясняется с точки зрения необходимости для каждой функции иметь определенные правила выполнения и определенный результат, иначе смысла в функциональном блоке просто не будет.

IDEF0 используется не только для иллюстрации процессов предприятия, но и в целом для описания схемы его функционирования, с какими ресурсами компания работает, от чего зависит и какой итоговый результат производит. Однако в некоторой степени содержание функциональных блоков IDEF0 остается «черным ящиком», в связи с чем и проводится построение дополнительных схем процессов, уже в нотации IDEF3

Результатом моделирования процессов является модель, которая относится к одному из трех типов:

- модель AS-IS (как есть) - модель текущей организации процессов;
- модель TO-BE (как будет) - модель идеальной организации процессов;
- модель SHOULD-BE(как должно бы быть) - идеализированная модель, не отражающая реальную организацию процессов.

Модель (AS-IS, TO-BE или SHOULD-BE) может содержать 4 типа диаграмм:

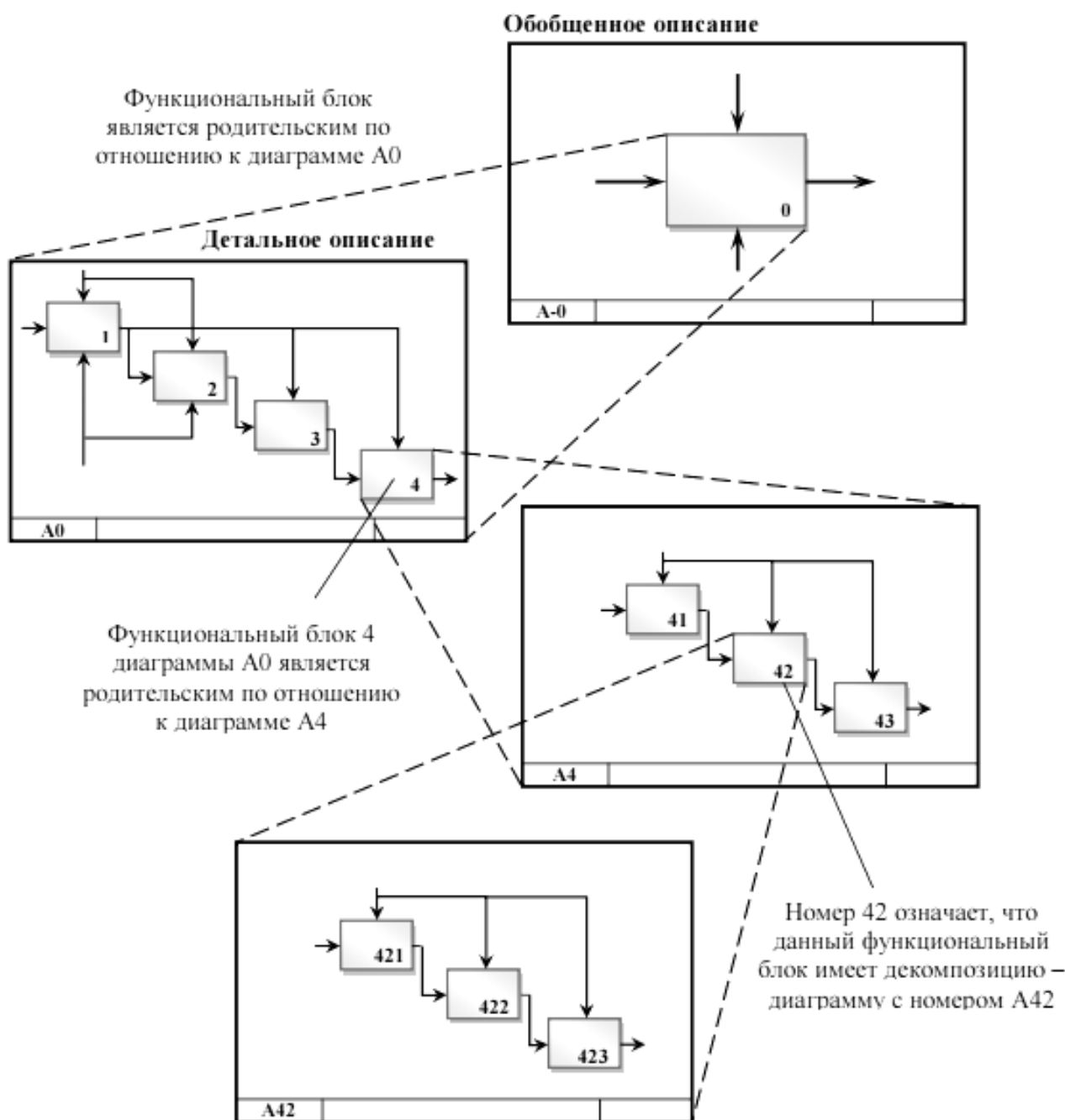
- контекстную диаграмму;
- диаграммы декомпозиции;
- диаграммы дерева узлов;
- презентационные диаграммы (диаграммы только для экспозиции) (for exposition only, FEO).

## 24.Методология функционального моделирования процессов –IDEF0.

### Последовательность создания функциональных моделей.

Первый шаг построения IDEF0 - построение основной контекстной диаграммы, то есть с представления всей системы в виде простейшей компоненты (контекстной диаграммы). Данная диаграмма отображает назначение (основную функцию) системы, необходимые входные и выходные данные, управляющую и регламентирующую информацию, а также механизмы.

То есть модель IDEF0 всегда начинается с представления системы как единого целого - одного функционального блока, с интерфейсными дугами, простирающимися за пределы рассматриваемой области. Такая диаграмма называется контекстной диаграммой, и обозначается идентификатором «А-0»



Декомпозиция функциональных блоков.

На втором шаге контекстная диаграмма детализируется с помощью диаграммы декомпозиции первого уровня. На этой диаграмме отображаются функции системы, которые должны быть реализованы в рамках основной функции.



Как правило, при построении диаграммы декомпозиции исходная функция (декомпозируемая) разбивается на 3-8 подфункций (блоков).

После построения диаграммы декомпозиции первого уровня для указанных на ней функций строятся отдельные диаграммы (диаграммы декомпозиции второго уровня). Затем процесс декомпозиции (построения диаграмм) продолжается до тех пор, пока дальнейшая детализация функций не теряет смысла. Для каждой атомарной функции, описывающей элементарную операцию (т.е. функции, не имеющей диаграммы декомпозиции), составляется подробная спецификация, определяющая ее особенности и алгоритм реализации.

Стрелки, входящие в блок и выходящие из него на диаграмме верхнего уровня, являются теми же самыми, что и стрелки, входящие в диаграмму нижнего уровня и выходящие из нее.

После определения состава функций и взаимосвязей между ними, возникает вопрос о правильной их композиции (объединении) в модули (подсистемы). При этом подразумевается, что каждая отдельная функция должна решать одну, строго определенную задачу. В противном случае необходима дальнейшая декомпозиция или разделение функций.

При объединении функций в подсистемы необходимо стремиться, чтобы внутренняя связность (между функциями внутри модуля) была как можно сильнее, а внешняя (между функциями, входящими в разные модули), как можно слабее.

- иерархическая связь имеет место между функцией и подфункциями, из которых она состоит;
- регламентирующая связь отражает зависимость одной функции от другой, когда выход одной работы направляется на управление другой.
- функциональная связь имеет место, когда выход одной функции служит входными данными для следующей функции. С точки зрения потока материальных объектов данная связь показывает технологию (последовательность работ) обработки этих объектов.
- потребительская связь имеет место, когда выход одной функции служит механизмом для следующей функции.

Из приведенных выше типов связей наиболее сильной является иерархическая связь, которая, по сути, и определяет объединение функций в модули (подсистемы). Несколько слабее являются регламентирующие, функциональные и потребительские связи. Функции с этими связями обычно реализуются в одной подсистеме.

Третий шаг построение диаграммы дерева узлов. Это обзорная диаграмма, показывающая структуру всей модели. Обычно вершина дерева соответствует контекстному блоку, под вершиной выстраивается вся иерархия блоков модели. Обзор модели с использованием дерева помогает сконцентрироваться на функциональной декомпозиции модели.

Четвёртый шаг построение презентационных FEO-диаграмм для иллюстрации других точек зрения или деталей, выходящих за рамки традиционного синтаксиса IDEF0. Диаграммы FEO допускают нарушение любых правил построения диаграмм IDEF0 в целях выделения важных с точки зрения аналитика частей модели.

Один из способов использования FEO-диаграмм состоит в отделении функционального блока от его окружения посредством создания диаграммы с единственным блоком и всеми относящимися к нему стрелками наподобие контекстной диаграммы.

## 25.Методология анализа взаимосвязей между информационными потоками –IDEF1(IDEF1X).

Логическая модель позволяет понять суть проектируемой системы, отражая логические взаимосвязи между сущностями. Различают три уровня логической модели, отличающихся по глубине представления информации о данных:

- диаграмма сущность-связь (Entity Relationship Diagram, ERD);
- модель данных, основанная на ключах (Key Based model, KB);
- полная атрибутивная модель (Fully Attributed model, FA).

Диаграмма сущность-связь представляет собой модель данных верхнего уровня. Она включает сущности и взаимосвязи, отражающие основные бизнес правила предметной области. Диаграмма сущность-связь может включать связи «многие-ко-многим» и не включать описание ключей.

Базовыми понятиями ERD являются:

Сущность (Entity) - множество экземпляров реальных или абстрактных объектов (людей, событий, состояний, идей, предметов и др.), обладающих общими атрибутами или характеристиками. Любой объект системы может быть представлен только одной сущностью, которая должна быть уникально идентифицирована.

Связь (Relationship) - поименованная ассоциация между двумя сущностями, значимая для рассматриваемой предметной области. Связь - это ассоциация между сущностями, при которой каждый экземпляр одной сущности ассоциирован с произвольным (в том числе нулевым) количеством экземпляров второй сущности, и наоборот.

Каждая связь может иметь один из следующих типов:

- связь типа один-к-одному (1:1) означает, что один экземпляр первой сущности (левой) связан с одним экземпляром второй сущности (правой).
- связь типа один-ко-многим означает, что один экземпляр первой сущности (левой) связан с несколькими экземплярами второй сущности (правой). Левая сущность (со стороны «один») называется родительской, правая (со стороны «много») - дочерней (1:n);
- связь типа много-ко-многим. Тип связи много-ко-многим является временным типом связи, допустимым на ранних этапах разработки модели. В дальнейшем этот тип связи должен быть заменен двумя связями типа один-ко-многим путем создания промежуточной сущности (m:n).

Каждая связь может иметь одну из двух модальностей связи:

- модальность «может» означает, что экземпляр одной сущности может быть связан с одним или несколькими экземплярами другой сущности, а может быть, и не связан ни с одним экземпляром (например: каждый пользователь может подать несколько заявок);
- модальность «должен» означает, что экземпляр одной сущности обязан быть связан не менее чем с одним экземпляром другой сущности (например: логин должен принадлежать только одному пользователю). Связь может иметь разную модальность с разных концов.

Методология IDEF1X - язык для семантического моделирования данных, основанный на концепции «сущность-связь» (ERD). Графическая нотация IDEF1 применяется для построения информационной модели, эквивалентной реляционной модели в третьей нормальной форме. На основе совершенствования метода IDEF1 создана его новая версия - метод IDEF1X, разработанный с учетом таких требований, как простота для изучения и возможность автоматизации.

В IDEF1X все сущности делятся на зависимые и независимые от идентификаторов. Сущность является независимой от идентификаторов, если каждый экземпляр сущности может быть однозначно идентифицирован без определения его отношений с другими сущностями. Сущность называется зависимой от идентификаторов, если однозначная идентификация экземпляра сущности зависит от его отношения к другой сущности.

Если экземпляр сущности-потомка однозначно определяется своей связью с сущностью-родителем, то связь называется идентифицирующей, в противном случае - не идентифицирующей. Идентифицирующая связь между сущностью-родителем и сущностью-потомком изображается сплошной линией. Сущность-потомок в идентифицирующей связи является зависимой от идентификатора сущностью.

Пунктирная линия изображает не идентифицирующую связь. Сущность-потомок в не идентифицирующей связи будет независимой от идентификатора, если она не является также сущностью-потомком в какой-либо идентифицирующей связи.

Связь может дополнительно определяться с помощью указания степени или мощности

- каждый экземпляр сущности-родителя может иметь ноль, один или более одного связанного с ним экземпляра сущности-потомка;
- каждый экземпляр сущности-родителя должен иметь не менее одного связанного с ним экземпляра сущности-потомка;
- каждый экземпляр сущности-родителя должен иметь не более одного связанного с ним экземпляра сущности-потомка;
- каждый экземпляр сущности-родителя связан с некоторым фиксированным числом экземпляров сущности-потомка

Связь изображается линией, проводимой между сущностью-родителем и сущностью-потомком, с точкой на конце линии у сущности-потомка. Мощность связей может принимать следующие значения:

- N - ноль, один или более;
- Z - ноль или один;
- P - один или более;
- цифра - заранее заданное число экземпляров сущности-потомка;

По умолчанию мощность связей принимается равной N

Атрибуты изображаются в виде списка имен внутри блока сущности. Атрибуты, определяющие первичный ключ, размещаются наверху списка и отделяются от других атрибутов горизонтальной чертой.

Сущности могут иметь также внешние ключи (Foreign Key), которые могут использоваться в качестве части или целого первичного ключа или не ключевого атрибута.

В качестве первичных ключей могут быть использованы несколько атрибутов или групп атрибутов. Атрибуты, которые могут быть выбраны первичными ключами, называются кандидатами в ключевые атрибуты (потенциальные атрибуты). Кандидаты в ключи должны уникально идентифицировать каждую запись сущности.

Правила устанавливают, что атрибуты и группы атрибутов должны:

- уникальным образом идентифицировать экземпляр сущности;
- не использовать NULL значений;
- не изменяться со временем.
- быть как можно более короткими для использования индексирования и получения данных.

Если необходимо использовать ключ, являющийся комбинацией ключей из других сущностей, то нужно убедиться в том, что каждая из частей ключа соответствует правилам. При выборе первичного ключа для сущности, разработчики модели часто используют дополнительный (суррогатный) ключ, т.е. произвольный номер, который уникальным образом определяет запись, в сущности.

Потенциальные ключи, которые не выбраны первичными, могут быть использованы в качестве вторичных или альтернативных ключей. С помощью альтернативных ключей часто отображают различные индексы доступа к данным в конечной реализации реляционной базы.

Если сущности в IDEF1X диаграмме связаны, связь передает ключ (или набор ключевых атрибутов) дочерней сущности. Эти атрибуты называются внешними ключами. Внешние ключи определяются как атрибуты первичных ключей родительского объекта, переданные дочернему объекту через их связь. Передаваемые атрибуты называются мигрирующими.



## 26.Методология описания (документирования) и моделирования процессов –IDEF3.

IDEF3 - способ описания процессов, основной целью которого является обеспечение структурированного метода, используя который эксперт в предметной области может описать положение вещей как упорядоченную последовательность событий с одновременным описанием объектов, имеющих непосредственное отношение к процессу.




Технология IDEF3 хорошо приспособлена для сбора данных, требующихся для проведения структурного анализа системы. В отличие от большинства технологий моделирования бизнес-процессов, IDEF3 не имеет жестких синтаксических или семантических ограничений, делающих неудобным описание неполных или нецелостных систем.

Любая IDEF3-диаграмма может содержать

- работы;
- связи;
- перекрестки (соединения);
- объекты ссылок.

Работа (Unit of Work, activity) изображается прямоугольником с прямыми углами, и имеет имя, выраженное отглагольным существительным, обозначающим процесс действия, одиночным или в составе фразы, и номер (идентификатор); другое имя существительное в составе той же фразы обычно отображает основной выход (результат) работы.


Связи предназначены для выделения существенных взаимоотношений между действиями. Все связи в IDEF3 являются однонаправленными.





Изображение	Название	Назначение
	Временное предшествование	Исходное действие должно завершиться прежде, чем конечное действие сможет начаться
	Объектный поток	Выход исходного действия является входом конечного действия. Из этого, в частности, следует, что исходное действие должно завершиться прежде, чем конечное действие сможет начаться
	Нечеткое отношение	Вид взаимодействия между исходным и конечным действиями задается аналитиком отдельно для каждого случая использования такого отношения

Связь типа «Объектный поток». Одной из наиболее часто встречающихся причин использования этого типа связи состоит в том, что некоторый объект, являющийся результатом выполнения исходного действия, необходим для выполнения конечного действия.

Связь типа «Нечеткое отношение». Используется для выделения отношений между действиями, которые невозможно описать с использованием предшественных или объектных связей. Значение каждой такой связи должно быть определено, поскольку связи данного типа сами по себе не предполагают никаких ограничений. Одно из применений нечетких отношений - отображение взаимоотношений между параллельно выполняющимися действиями.

Перекрестки (соединения). Завершение одного действия может инициировать начало выполнения сразу нескольких других действий, или, наоборот, определенное действие может требовать завершения нескольких других действий для начала своего выполнения.

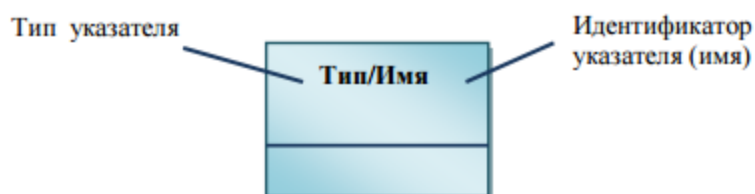
Обозначение	Название	Смысл в случае слияния стрелок	Смысл в случае разветвления стрелок
	Asynchronous AND	Все предшествующие процессы должны быть завершены	Все следующие процессы должны быть запущены

	Synchronous AND	Все предшествующие процессы завершены одновременно	Все следующие процессы запускаются одновременно
	Asynchronous OR	Один или несколько предшествующих процессов должны быть завершены	Один или несколько следующих процессов должны быть запущены
	Synchronous OR	Один или несколько предшествующих процессов завершаются одновременно	Один или несколько следующих процессов запускаются одновременно
	XOR (Exclusive OR)	Только один предшествующий процесс завершен	Только один следующий процесс запускается

**Указатели** - специальные символы, которые ссылаются на другие разделы описания процесса. Они выносятся на диаграмму для привлечения внимания читателя к каким-либо важным аспектам модели. В табл. 3.16 представлены типы указателей модели IDEF3.

Тип указателя	Назначение
ОБЪЕКТ (OBJECT)	Для описания того, что в действии принимает участие какой-либо заслуживающий отдельного внимания объект.
ССЫЛКА (GOTO)	Для реализации цикличности выполнения действий. Указатель ССЫЛКА может относиться и к соединению.
ЕДИНИЦА ДЕЙСТВИЯ (Unit of Behavior – UOB)	Для помещения на диаграмму дополнительного экземпляра уже существующего действия без зацикливания.
ЗАМЕТКА (NOTE)	Для документирования любой важной информации общего характера, относящейся к изображенному на диаграммах (в этом смысле ССЫЛКА служит альтернативой методу помещения текстовых заметок непосредственно на диаграммах.
УТОЧНЕНИЕ (Elaboration – ELAB)	Для уточнения или более подробного описания изображенного на диаграмме. Обычно используются для описания логики ветвления у соединений

Указатель изображается на диаграмме в виде прямоугольника, рис.3.8, похожего на изображение действия. Имя указателя обычно включает его тип (например, OBJECT, UOB и т.п.) и идентификатор (имя).



Изображение и нумерация Указателя в диаграмме IDEF3.

Действия в IDEF3 могут быть декомпозированы, или разложены на составляющие, для более детального анализа. Декомпозировать действие можно несколько раз. Это позволяет документировать альтернативные потоки процесса в одной модели.

Для корректной идентификации действий в модели с множественными декомпозициями схема нумерации действий расширяется и наряду с номерами действия и его родителя включает в себя порядковый номер декомпозиции. Например, в номере действия 1.2.5: 1 - номер родительского действия, 2 - номер декомпозиции, 5 - номер действия.

## 27. Методология моделирования, анализа бизнес-процессов BPMN.

Модель и нотация бизнес-процессов - методология моделирования, анализа и реорганизации бизнес-процессов.

Основной целью BPMN является обеспечение доступной нотацией описания бизнес-процессов всех пользователей: от аналитиков и разработчиков до руководителей и обычных пользователей. Таким образом, BPMN нацелена на устранение расхождения между моделями бизнес-процессов и их реализацией.

Диаграмма процесса в нотации BPMN представляет собой алгоритм выполнения процесса. На диаграмме могут быть определены события, исполнители, материальные и документальные потоки, сопровождающие выполнение процесса. Каждый процесс может быть декомпозирован на более низкие уровни.

В нотации BPMN выделяют пять основных категорий элементов:

- элементы потока (Flow Objects) (события, процессы и шлюзы);
- данные (объекты данных и базы данных) (товарно-материальные ценности или информация);
- соединяющие элементы (Connecting Objects) (потоки управления, потоки сообщений и ассоциации);
- зоны ответственности (Swimlanes) (пулы и дорожки);
- артефакты (сноски) (Artifacts).

Элементы потока являются важнейшими графическими элементами, определяющими ход бизнес-процесса. Элементы потока, в свою очередь, делятся на:

- события (Events);
- действия (Activities);
- шлюзы (Gateways).

Выделяют три вида соединяющих элемента потока, связывающихся друг с другом и с другими элементами:

- поток операций (Sequence Flow);
- поток сообщений (Message Flow);
- ассоциация (Association).

Существуют два способа группировки основных элементов моделирования с помощью зон ответственности:

- группировка с помощью Пула (Pool);
- группировка с помощью Дорожки (Lane)

Артефакты используются для добавления дополнительной информации о Процессе.

Выделяют три типовых Артефакта, что, однако, не запрещает разработчикам моделей бизнес-процессов либо программам моделирования добавлять необходимое количество Артефактов. Текущий перечень Артефактов включает в себя следующие элементы:

- объект данных (Data object);
- группа (Group);
- аннотация (Annotation).

Разновидности диаграмм (типы процессов) BPMN:

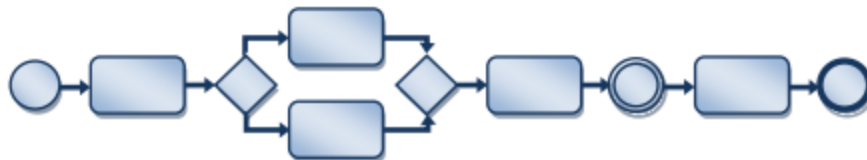


Рис. 1.24. Примерный вид BPMN диаграммы частного (внутреннего) бизнес-процесса.

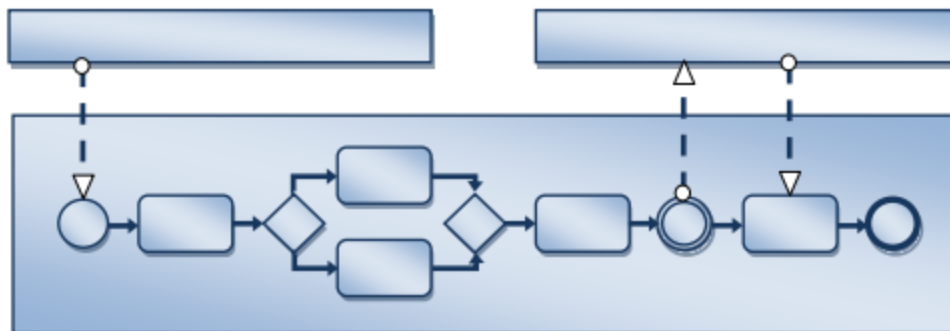


Рис.1.25. Примерный вид BPMN диаграммы публичного (открытого) процесса.

Процесс моделирования процессов с помощью BPMN подчиняется классическим принципам моделирования: декомпозиции и иерархического упорядочивания. Декомпозиция, с отображением на отдельных диаграммах, выполняется для участников (пулов) и отдельных под процессов, подобно работам на диаграммах IDEF0 или predetermined process on block-schemes.



Рис.1.26.Примерный вид BPMN диаграммы хореографии.

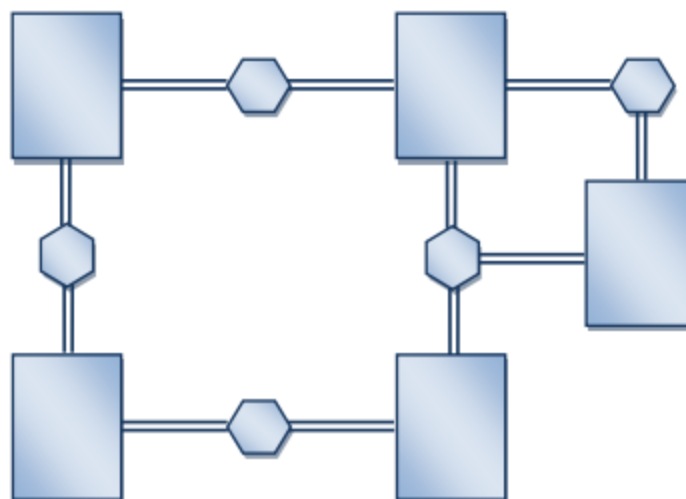
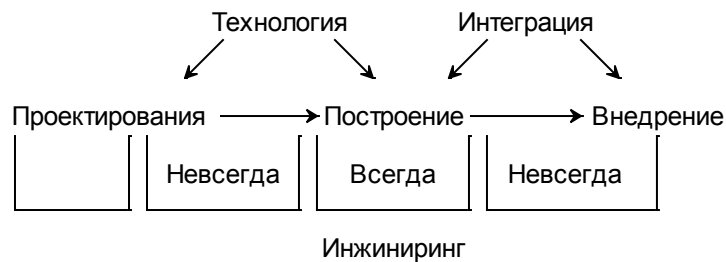


Рис.1.28.Примерный вид BPMN диаграммы посредством обмена сообщениями.



## 28. Реинжиниринг бизнес-процессов в ИС (реорганизации бизнес-процессов –BPMN).

Под бизнес процессом подразумевается совокупность взаимосвязанных работ по изготовлению конечного продукта или выполнению услуг. Преобразование бизнес процессов с точки зрения реорганизации процессов производства и управления называют реинжинирингом.



Бизнес процессы - все материальные физические и информационные потоки в системе, которая рассматривается совместно.

Первоначально под реинжинирингом бизнес процессов была связано с организацией организованного управления.

Содержание реинжиниринга БП системная реорганизация основных потоков. Упорядочивание и оптимизация организационной структуры. Реструктуризация производства.

Автоматизация процессов управления и производства.

Реинжиниринг ПО (задача тестирования, развития, связаны с расширением алгоритмов и функций).

Реинжиниринг информационных процессов в ИС это:

- Сбор информации
- Анализ информации
- Хранение информации

Цель Реинжиниринга БП - упрощение организационной структуры, минимизация ресурсов, сокращение сроков реализации задач клиентов и повышение качества.

Сокращение затрат не является целью реинжиниринга БП.

Основные задачи реинжиниринга БП:

Определение оптимальной последовательности

- Выполняемой функции ИС. Следствие - ускорение оборачиваемости средств организации.
- Оптимизация использования ресурсов для минимизации издержек. Средство сокращение издержек.
- Поиск оптимального сочетания видов деятельности.
- Построения адаптивных бизнес процессов.
- Определение рациональных схем взаимодействия субъектов бизнес процессов.

## 29.Методологии объектно-ориентированного подхода (анализа).

### **Объектно-ориентированный анализ и проектирование** (ООАП, *Object-Oriented Analysis/Design*)

- технология разработки программных систем, в основу которых положена объектно-ориентированная методология представления *предметной области* в виде *объектов*, являющихся экземплярами соответствующих *классов* .

В роли унифицированного языка визуализации был разработан язык моделирования UML

В рамках ООАП исторически рассматривались три графических нотации:

- диаграммы "сущность-связь" (*Entity-Relationship Diagrams, ERD*),
- диаграммы *функционального моделирования* (*Structured Analysis and Design Technique, SADT*),
- диаграммы потоков данных (*Data Flow Diagrams, DFD*).
- 

Основной недостаток данной методологии связан с отсутствием явных средств для объектно-ориентированного представления моделей сложных систем. Некоторые аналитики отмечают важность знания и применения нотации IDEF0, однако отсутствие возможности реализации соответствующих графических моделей в объектно-ориентированном программном коде существенно сужают диапазон решаемых с ее помощью задач.

В основе графического моделирования информационных систем с помощью диаграмм потоков данных лежит специальная технология построения диаграмм потоков данных DFD. В разработке методологии DFD приняли участие многие аналитики, среди которых следует отметить Э. Йордона. Он автор одной из первых графических нотаций DFD.

Недостаток рассмотренных нотаций связан с отсутствием явных средств для объектно-ориентированного представления моделей сложных систем, а также сложных алгоритмов обработки данных. Поскольку на рассмотренных типах диаграмм не указываются характеристики времени выполнения отдельных процессов и передачи данных между процессами, то модели систем, реализующих синхронную обработку данных, не могут быть адекватно представлены в этих нотациях. Все эти особенности методов структурного системного анализа ограничили возможности широкого применения соответствующих нотаций и послужили основой для разработки унифицированного языка моделирования UML.

### 30. Средства объектно-ориентированного подхода (анализа).

На рынке CASE-средств представлены десятки программных инструментов, поддерживающих нотацию языка *UML* 1.4-1.5 и обеспечивающих интеграцию, включая прямую и обратную генерацию кода программ, с наиболее распространенными языками и средами программирования, такими как *MS Visual C++*, *Java*, *Object Pascal/Delphi*, *Power Builder*, *MS Visual Basic*, *Forte*, *Ada*, *Smalltalk*.

С каждым годом интерес к языку *UML* со стороны специалистов неуклонно возрастает. Язык *UML* повсеместно становится не только основой для разработки и реализации во многих перспективных инструментальных RAD-средствах, но и в CASE-средствах визуального и имитационного моделирования. Более того, заложенные в языке *UML* потенциальные возможности широко используются как для объектно-ориентированного моделирования систем, так и для документирования бизнес-процессов, а в более широком контексте - для представления знаний в интеллектуальных системах, которыми, по существу, станут перспективные сложные программно-технологические комплексы.

### 31. Методологии модельно-ориентированного подхода (анализа).

Модельно-ориентированное проектирование (МОП) - эффективный и экономически выгодный способ разработки систем управления и создания встраиваемых систем. Вместо физических прототипов и текстовых спецификаций в модельно-ориентированном проектировании применяется исполняемая модель.

При таком подходе можно разрабатывать и проводить имитационное моделирование как всей системы целиком, так и ее компонентов. Автоматическая генерация программного кода позволяет избежать большинства ошибок, связанных с человеческим фактором и уменьшить время разработки более чем в два раза.

Типовая ИС в специальной базе метаданных - репозитории - содержит модель объекта автоматизации, на основе которой осуществляется конфигурирование программного обеспечения. Таким образом, модельно ориентированное проектирование ИС предполагает, прежде всего, построение модели объекта автоматизации с использованием специального программного инструментария. Возможно также создание системы на базе типовой модели ИС из репозитория, который поставляется вместе с программным продуктом и расширяется по мере накопления опыта проектирования информационных систем для различных отраслей и типов производства.

Репозиторий содержит базовую модель ИС, типовые модели определенных классов ИС, модели конкретных ИС предприятий.

Базовая модель ИС в репозитории содержит описание бизнес-функций, бизнес-процессов, бизнес-объектов, бизнес-правил, организационной структуры, которые поддерживаются программными модулями типовой ИС.

Модель конкретной системы строится либо путем выбора фрагментов основной или типовой модели в соответствии со специфическими особенностями объекта автоматизации, либо путем автоматизированной адаптации этих моделей в результате экспертного опроса.

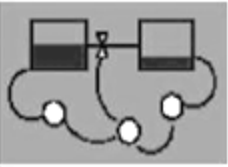
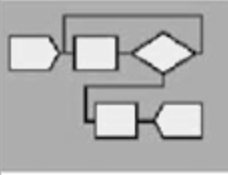

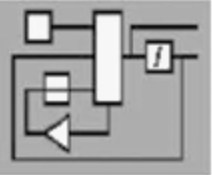
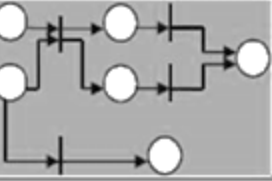
Построенная модель в виде метаописания хранится в репозитории и при необходимости может быть откорректирована.

Принципы МОП существенно отличаются от традиционной методологии проектирования. Вместо создания программных кодов разработчики могут применять МОП для улучшения характеристик модели, используя стандартные функциональные блоки. Построенные таким образом модели вместе с использованием инструментов для моделирования, могут привести к созданию прототипа системы управления.

Преимущества МОП перед традиционным подходом проектирования:

- МОП предоставляет общую среду разработки, что способствует взаимодействию группы разработчиков в процессе анализа данных и проверки системы;
- инженеры могут найти и исправить ошибки на ранних стадиях проектирования системы, когда затраты времени и финансовые последствия изменения системы сводятся к минимуму;
- МОП способствует повторному использованию моделей для улучшения системы и создания производных систем с расширенными возможностями.

### 32. Средства модельно-ориентированного подхода (анализа).

Системная динамика	Дискретные системы	Агентное моделирование	Динамические системы	Сети
				
Vensim, iThink, Powersim, AnyLogic	GPSS, Simula, Arena, AutoMod, AnyLogic, Extend, ProModel, QUEST, SIMFACTORY II.5, SIMPLE++, eM-Plant, Taylor ED, WITNESS	AnyLogic	MATLAB	ARIS

### 33. Гибкие методологии проектирования (agile-методы).

Agile - это семейство методологий разработки программного обеспечения, для которых характерны:

- итеративный процесс разработки;
- динамичное формирование требований и их реализация;
- тесная взаимосвязь, активное взаимодействие, как между абсолютно всеми участниками команды разработчиков, так и между командой и заказчиком.

Agile Manifesto содержит 4 основные идеи и 12 принципов.

Основные идеи: Люди и взаимодействие **важнее** процессов и инструментов Работаящий продукт **важнее** исчерпывающей документации

Сотрудничество с заказчиком **важнее** согласования условий контракта

Готовность к изменениям **важнее** следования первоначальному плану.

То есть, не отрицая важности того, что справа, мы всё-таки больше ценим то, что слева.

Основные принципы:

- наивысшим приоритетом для нас является удовлетворение потребностей заказчика;
- изменение требований приветствуется, даже на поздних стадиях разработки (Agile-процессы используют изменения для обеспечения заказчику конкурентного преимущества);
- работающий продукт следует выпускать как можно чаще;
- на протяжении всего проекта разработчики и представители бизнеса должны ежедневно работать вместе;
- над проектом должны работать **мотивированные** профессионалы;
- непосредственное общение является наиболее практичным и эффективным способом обмена информацией как с самой командой, так и внутри команды;
- работающий продукт - основной показатель прогресса;
- инвесторы, разработчики и пользователи должны иметь возможность поддерживать постоянный ритм бесконечно (Agile помогает наладить такой устойчивый процесс разработки);
- постоянное внимание к техническому совершенству и качеству проектирования повышает гибкость проекта;
- простота - искусство минимизации лишней работы - крайне необходима;
- самые лучшие требования, архитектурные и технические решения рождаются у самоорганизующихся команд;
- команда должна систематически анализировать способы улучшения эффективности.

Agile Modeling (AM) - это набор понятий, принципов и приемов (практик), позволяющих быстро и просто выполнять моделирование и документирование в проектах разработки программного обеспечения (ПО).

AM описывает стиль моделирования, который позволит повысить качество и сократить сроки. AM не является технологическим процессом. Это не детальная инструкция по проектированию. AM сосредоточен на эффективном моделировании и документировании. Он не охватывает программирование и тестирование. AM также не включает вопросы управления проектом, развертывания и сопровождения системы.

AM должен рассматриваться как дополнение к существующим методам, а не самостоятельная технология. Этот метод должен использоваться для повышения эффективности труда разработчиков.

## 34. Понятие CASE-технологии проектирования ИС. Основные принципы Case-технологии.

Проектирование ИС с применением компьютерной поддержки, называется CASE - технологии проектирования. CASE - технологии применяются не только для автоматизации проектирования ИС, но и для разработки моделей бизнес- процессов при проведении бизнес-анализа. CASE - технологии применяются в ситуациях, когда проблематика предметной области отличается большой сложностью.

Можно выделить следующие основные принципы создания ИС на основе CASE - технологий:

- принцип всесторонней компьютерной поддержки проектирования;
- принцип модельного подхода. CASE - система может поддерживать методологию функционально-ориентированного или объектно-ориентированного подхода;
- принцип иерархического представления модели предметной области. Данный принцип выражается в возможности последовательной детализации (декомпозиции) описания системы;
- принцип наглядности представления модели - означает наличие в составе CASE-технологий визуальных средств проектирования. Система графических изображения и правила, называются нотацией CASE-средства;
- принцип декомпозиции процесса проектирования ИС с применением CASE-технологий на стадии и этапы.

Большинство современных CASE-средств поддерживает методологии структурного и/или объектно-ориентированного анализа и проектирования информационных систем. Выбор того или иного подхода (парадигмы) подразумевает следование ему и на стадии кодирования. Их отличие друг от друга заключается в выборе способа декомпозиции системы. Если за основу принимается функциональная (алгоритмическая) декомпозиция, то речь идет о структурном подходе, если объектная - об объектно-ориентированном.

### 35.Классификация CASE-средств, стратегия их выбора.

Классификация по уровню проектирования в жизненном цикле создания ИС:

- средства верхнего уровня (Upper CASE) - анализ предметной области, определение места ИС в контуре бизнес-системы;
- средства среднего уровня (Middle CASE) - разработка архитектуры ИС, создание проектных спецификаций;
- средства нижнего уровня (Lower CASE) - поддержка разработки программного обеспечения.

Классификация по типам отражает функциональную ориентацию CASE - средств на те или иные процессы ЖЦ и включает следующие типы:

- средства анализа (соответствуют Upper CASE), предназначенные для построения и анализа моделей предметной области - (CA ERwin Process Modeler ARIS);
- средства анализа и проектирования (соответствуют Middle CASE), поддерживающие наиболее распространенные методологии проектирования, которые используются для создания проектных спецификаций. (CASE- аналитик, ARIS);
- средства проектирования баз данных (соответствуют Middle CASE), обеспечивающие моделирование данных и генерацию схем баз данных для наиболее распространенных СУБД: ER-win, Data Base Designer, ARIS Tooleset;
- средства разработки приложений (соответствуют Lower CASE) - средства 4GL - SQL Windows (Gupta), Delphi (Borland) и др. и генераторы кода, входящие в состав Vantage Team Builder, PROIV и частично - в Silverrun;
- средства реинжиниринга, обеспечивающие анализ программных кодов и схем БД и формирование на их основе различных моделей и проектных спецификаций: ER-win, ARIS Tooleset, Designer/2000 и т.д. В области анализа программных кодов наибольшее распространение получают объектно-ориентированные CASE-средства.

Классификация по степени интегрированности выделяет:

- локальные CASE-средства - применяются для анализа системы и разработки автоматизированных рабочих мест, поддерживают 1, 2 типа моделей и методов (CASE- аналитик, Design/IDEF);
- малые интегрированные CASE-средства, используются для создания небольших ИС, поддерживают несколько типов моделей и методов (ER-win, BPwin, Silverran);
- средние интегрированные CASE-средства - поддерживают от 4-15 моделей и методов. В этой категории Rational Rose, Designer/2000;
- крупные интегрированные CASE-средства, поддерживаются свыше 15 типов моделей и методов (семейство программных продуктов ARIS).



### 36. Проблемы и перспективы проектирования ИС и ИТ.

Самой первой проблемой является проблема проектирования. Нельзя начинать техническую разработку, не имея тщательно проработанного проекта. Если начинать с решения наиболее очевидных задач, не обращая внимания на потенциально существующие, то такая система будет непрерывно находиться в стадии разработки и переделки.

Первой стадией проектирования должен быть анализ требований корпорации. Для этого на основе экспертных запросов необходимо выявить все актуальные и потенциальные потребности корпорации, которые должны удовлетворяться проектируемой информационной системой, понять какие потоки данных существуют внутри корпорации, оценить объемы информации, которые должны поддерживаться и обрабатываться информационной системой. Эта стадия, как правило, носит неформальный характер, хотя, конечно, очень важно сохранить полученную информацию, поскольку она должна входить в документацию системы.

Следующая стадия проектирования - выработка концептуальной схемы базы данных, которая будет лежать в основе информационной системы. Сначала придется выбрать систему *нотаций*, в которой будет представляться концептуальная схема. Таких нотаций существует великое множество, и хотя практически все они диаграммные (в духе ER-модели), они отличаются одна от другой. Заметим, что даже в случае использования некоторых CASE-средств вам все равно предлагается на выбор несколько нотаций.

Поэтому, с большой вероятностью, на следующей стадии проектирования понадобится на основе имеющейся концептуальной схемы произвести набор определений схемы базы данных.

#### Перспективы

К настоящему времени на международном уровне сформировалась мощная кооперация организаций, разрабатывающих стандарты в области ИТ. Среди этих организаций в первую очередь следует назвать Международную организацию по стандартизации — ИСО и Международную электротехническую комиссию — МЭК.

ИСО и МЭК объединили свою деятельность по стандартизации в области информатизации, создав Совместный технический комитет № 1 — СТК1 ИСО/МЭК «Информационные технологии» основной задачей которого является разработка базовых стандартов информационных технологий вне зависимости от их конкретных приложений. В последние годы СТК1 ИСО/МЭК активно взаимодействует с техническими комитетами ИСО.

Находятся в стадии разработки стандарты ГОСТ Р ИСО/МЭК по программной инженерии: по измерению и оценке производительности ПО автоматизированных систем; на процесс создания документации пользователя ПС; по сопровождению ПС; по оценке уровня целостности систем, и ПС; руководство по применению ГОСТ Р ИСО/МЭК 12207 при управлении проектами.

За последние годы получили значительное развитие работы по созданию нормативной базы в области CALS-технологий. Стандарты CALS-технологий направлены прежде всего на обеспечение качественно нового уровня организации процессов проектирования, производства и эксплуатации сложной продукции.