

1. Автоматизированные информационные системы

Автоматизированная информационная система (АИС) — совокупность программно-аппаратных средств, предназначенных для автоматизации деятельности, связанной с хранением, передачей и обработкой информации.

АИС может быть определена как комплекс автоматизированных информационных технологий, предназначенных для информационного обслуживания – организованного непрерывного технологического процесса подготовки и выдачи потребителям научной, управленческой и др. информации, используемой для принятия решений, в соответствии с нуждами для поддержания эффективной деятельности.

Классическими примерами автоматизированных информационных систем являются банковские системы, автоматизированные системы управления предприятиями, системы резервирования авиационных или железнодорожных билетов и т. д.

Назначение АИС

Основной причиной создания и развития АИС является необходимость ведения учета информации о состоянии и динамике объекта, которому посвящена система. На основании информационной картины, создаваемой системой, руководители различного звена могут принимать решения об управляющих воздействиях с целью решения текущих проблем.

Учетные данные системы могут быть подвергнуты автоматической обработке для последующего тактического и стратегического анализа с целью принятия управленческих решений большего горизонта действия.

Побочными, возможными, но не гарантированными эффектами от использования системы могут выступать:

- повышение производительности работы персонала;
- улучшение качества обслуживания клиентов;
- снижение трудоемкости и напряженности труда персонала;
- снижение количества ошибок в его действиях.

Типы автоматизированных информационных систем

Какая-либо однозначная и общепринятая классификация АИС отсутствует, однако в науке и индустрии по крайней мере выделяют следующие типы систем по назначению:

- [АСУ](#) — Автоматизированные системы управления
- [АСУП](#) — Автоматизированные системы управления предприятия
- [АСКУЭ](#) — Автоматизированная система контроля и учёта энергоресурсов
- [АСУ ТП](#) — Автоматизированные системы управления технологическими процессами

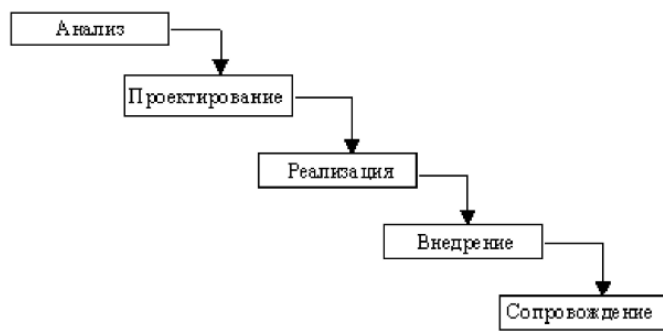
- [ГИС](#) — Геоинформационные системы
- [ИУС](#) — Информационно-управляющие системы
- [ИИС](#) — Информационно-измерительные системы
- [ИИС](#) — Интеллектуальные информационные системы
- [ИПС](#) — Информационно-поисковые системы
- [ИАС](#) — Информационно-аналитические системы
- [ИСС](#) — Информационно-справочные системы;
- [ЛИС](#) — Лабораторная информационная система
- [СИИ](#) — [Системы искусственного интеллекта](#)
- [СКД](#), [СКУД](#) — Система контроля (и управления) доступом

2. Жизненный цикл ИС. Модели жизненного цикла

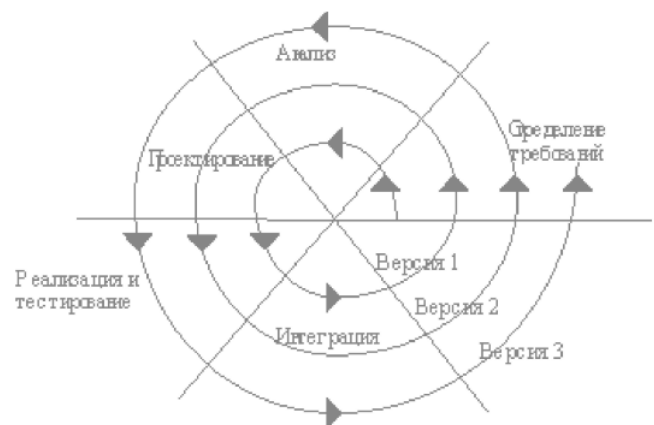
Жизненный цикл – это модель создания и использования ИС, отражающая ее различные состояния, начиная с момента возникновения необходимости в данном комплексе средств и заканчивая моментом его полного выхода из употребления пользователей.

Этапы жизненного цикла: анализ, проектирование, разработка, тестирование, внедрение, сопровождение.

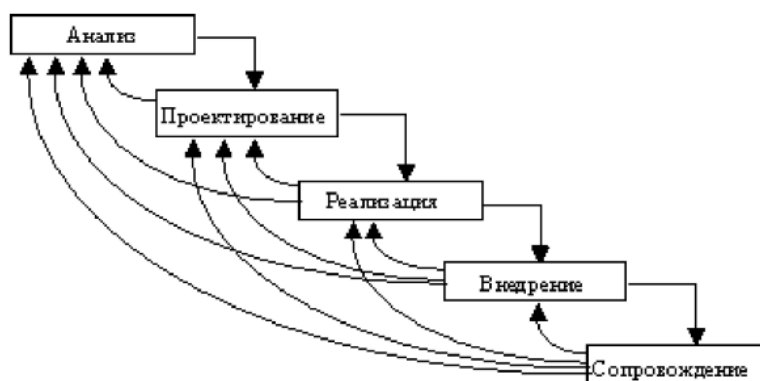
Модели:



Каскадная модель



Спиральная модель



Поэтапная итерационная модель

3. Эффективность ИС

Для оценки эффективности существует ряд критериев, которые количественно определяют степень соответствия системы целям ее создания.

Критерий эффективности должен быть наглядным и напрямую зависеть от работы системы, допускать приближенную оценку по результатам экспериментов.

Оценивают как ИС в целом, так и ее компоненты.

Одновременное достижение всех целей невозможно, поэтому один из критериев оптимизируется, а остальные служат в качестве ограничений.

Таблица: Цели и критерии создания ИС

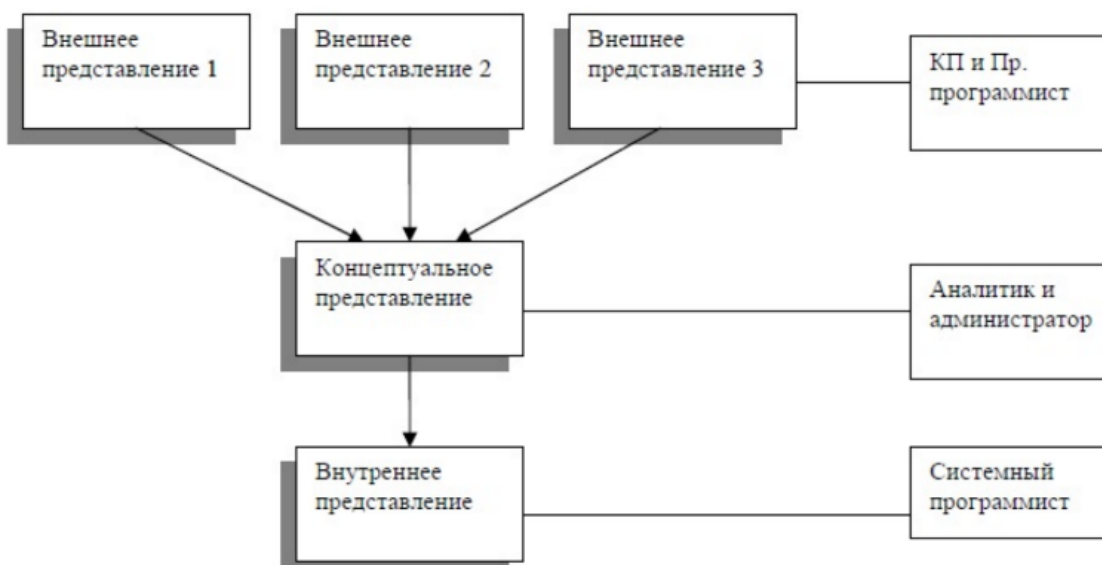
№	Цели	Критерии
1	Максимальная полнота отображения информации	Отношение объема информации в системе к объему информации на объекте
2	Максимальная скорость предоставления информации	Время обработки данных Время ответа на запрос
3	Максимальное удобство пользователя	Время на формирование запроса и понимание ответа
4	Минимальные расходы	Капитальные вложения + Текущие затраты
5	Максимальное извлечение полезной информации	Отношение объема входной информации к объему выходной информации
6	Минимальная избыточность базы данных	Отношение объема избыточной информации к объему хранимой информации

4. Пользователи ИС. Трехуровневое представление данных

Группы пользователей: *случайный*; *конечный* (потребитель информации) – в интересах которого работает ИС; *коллектив специалистов* (персонал ИС) – включающий:

- администратора (отвечает за определение, загрузку и эффективность работы банка данных),
- системного аналитика (строит мат модель предметной области исходя из информационных потребностей конечных пользователей, ставит задачи для прикладных программистов),
- программистов (системные – разрабатывают и сопровождают базовое математическое обеспечение ЭВМ, работают с ОС, СУБД, трансляторами и т. д.; прикладные – разрабатывают программы для реализации запросов к БД).

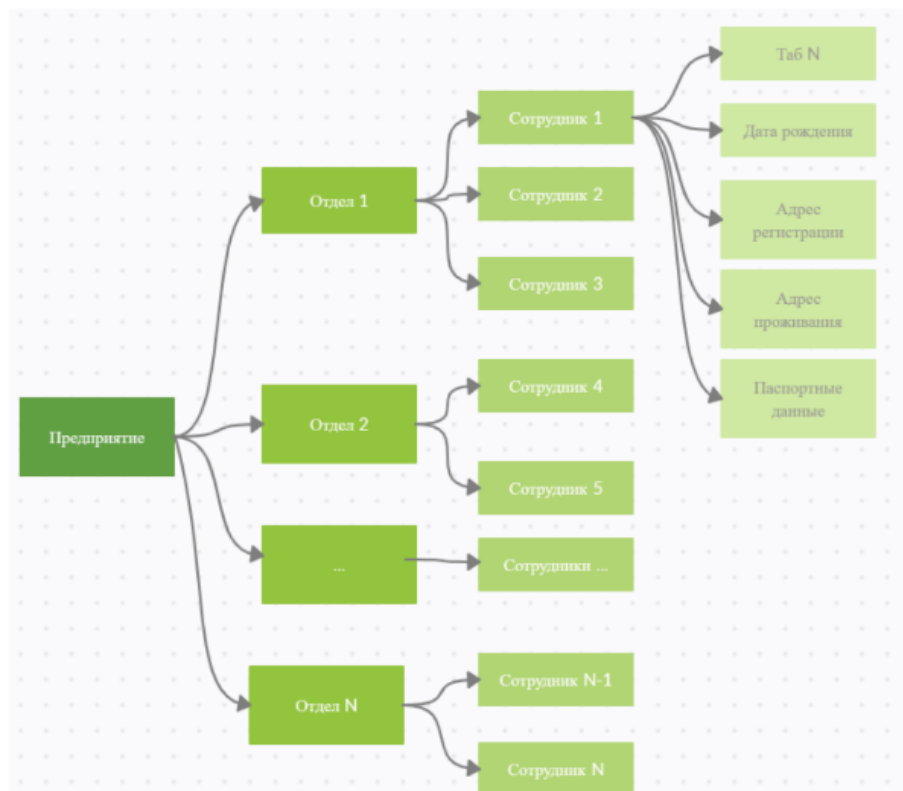
Трехуровневое представление данных



5. Классификация ИС

- По режиму работы: **пакетные** – работают в пакетном режиме (накапливают пакет данных и потом этот пакет последовательно обрабатывается), **диалоговые** – работают в режиме обмена сообщениями между пользователями и системой (например, продажа авиабилетов), **смешанные**.
- По способу распределения вычислительных ресурсов: **локальные** – используют одну ЭВМ, **распределенные** – несколько ЭВМ, связанные сетью (решают отдельные задачи, но используют общую информационную базу).
- По функциям: **системы обработки данных (СОД)** – для решения задач типа расчета зп или статистической отчетности (выполняют расчеты без оптимизации), **автоматизированные системы управления (АСУ)** – сама выполняет управленческие функции к объекту, включает в себя прикладные программы для принятия и автоматизации управленческих решений (например, система оптимального управления запасами на складе), **информационно-поисковые системы (ИПС)**: документографическая ИПС, фактографическая ИПС.
- По степени автоматизации: **неавтоматизированные** (запрос обрабатывается человеком), **автоматизированные** (обработка с помощью специальных программ), **автоматические**.
- По концепции построения: **файловые системы** – ОС берет на себя распределение внешней памяти, ПО ИС напрямую использует функции ОС для работы с файлами (хранит слабо структурированную информацию, сложно решить проблемы согласования данных в разных файлах, коллективного доступа к данным и модификации структуры данных), **базы данных** – структура меньше зависит от прикладных программ, все функции по работе с БД сосредоточены в СУБД, сведения о структуре БД сосредоточены в репозитории (использование метainформации – автоматизируется с помощью словарей данных), **интеллектуальные базы данных (базы знаний)** – способ построения ИС, при котором информация условно делится между двумя базами (база данных содержит сведения о количественных и качественных характеристиках объектов, база знаний содержит сведения о закономерностях в ПО, обеспечивают понимание запроса и синтез ответа), **хранилища данных** – автономная БД, в котором БД разделена на два компонента (оперативная БД, которая хранит текущую информацию, квазипостоянная БД, которая содержит исторические данные).

6. Структура хранения информации в ИС



Пример структуры хранения данных

Структура данных в ИС обычно сложна (сложность не столько в объеме, сколько в количестве взаимосвязей). Задачи по обработке данных однотипны для разных предметных областей (создание, поиск, ввод и вывод, сортировка), поэтому все типовые функции по работе с данными выделены в специальную систему (СУБД – система управления БД – комплекс программных и языковых средств создания, ведения и манипулирования данными).

7. Этапы разработки автоматизированных информационных систем

Согласно ГОСТ 34.601–90 «Автоматизированные системы. Стадии создания» выделяют следующие основные стадии создания и этапы разработки автоматизированной системы (АС):

- Формирование требований к АС.
- Разработка концепции АС.
- Техническое задание.
- Эскизный проект.
- Технический проект.
- Рабочая документация.
- Ввод в действие.
- Сопровождение АС.

8. Основные понятия электронного документооборота. ECM-системы

ECM (Enterprise Content Management) – стратегическая инфраструктура и техническая архитектура для поддержки единого жизненного цикла неструктурированной информации (контента) различных типов и форматов.

ECM представляет собой совокупность методов и инструментов, предназначенных для захвата, систематизации, хранения и поиска контента и документов, связанных с бизнес-процессами организации.

Разделы ECM:

- Управление документами (document management) – экспорт/импорт, контроль версий и обеспечение безопасности для деловых документов;
- Управление образами документов (document imaging) — захват, преобразование бумажных документов в электронную форму;
- Управление записями (records management) — долгосрочное архивное хранение, разработка норм и политик хранения, обеспечение соответствия законодательным и отраслевым нормам управление почтовыми сообщениями (e-mail management) — систематизация и хранение информацией, поступающей в организацию по эл. почте;
- Управление социальными медиа (social media management, SMM) – управление информацией, поступающей в организацию в процессе взаимодействия с внешней средой через социальные сети;
- Управление веб-контентом — поддержка корпоративных веб-порталов, управление динамическим контентом и взаимодействием пользователей;
- Управление мультимедиа-контентом (digital assets management, DAM) — хранение и систематизация графических, аудио- и видеоматериалов;
- Управление потоками работ (workflow management) – поддержка бизнес-процессов, доставка контента по маршрутам, назначение рабочих задач и состояний.

Примеры ECM: *Oracle* (Oracle Enterprise Content Management OEM); *IBM* (IBM ECM); *EMC* (EMC Documentum); *OpenText* (OpenText ECM Suite).

Система электронного документооборота (СЭД) – это компьютерная программа, которая позволяет организовать работу с электронными документами (создание, изменение, поиск), а также взаимодействие между сотрудниками (передачу документов, выдачу заданий, отправку

уведомлений и т. п.). Примеры: Directum (Directum), DocsVision (DocsVision), Globus Professional (Проминфосистемы), PayDox (Paybot), 1С:Документооборот (1С).



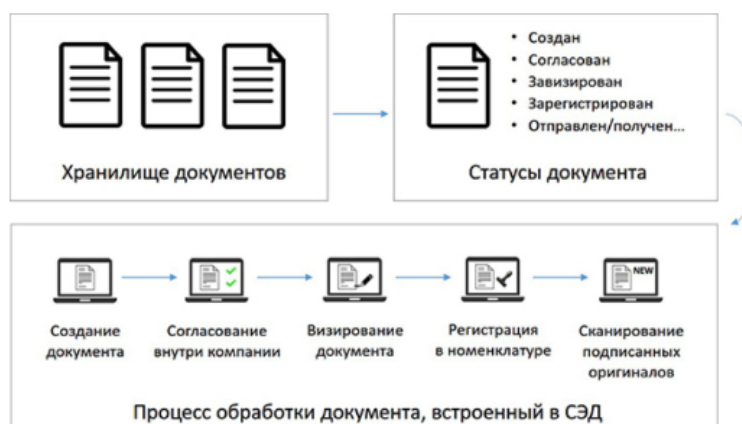
Логическая структура ECM

Система ввода (Capture): *созданное людьми* (офисные доки, формы, мультимедиа), *распознанное* (OCR, HCR, ICR, OMR, Штрих-код), *созданные приложениями* (ERP, XML, финансовые приложения, электронные платежи), *обработка форм* (E-Forms/Web-Forms), *агрегирование*, COLD/ERM.

Система управления (Management). Компоненты управления предназначены для управления, обработки и использования информации, которые включают в себя: БД для администрирования и выборки, системы авторизации доступа для защиты информации.

Система хранения (Store). Включает функциональности и компоненты для временного хранения информации, которая не предназначена для архивирования. Система сохранения (Preserve). Включает функциональности и компоненты, работающие с долговременным безопасным хранением и резервными копиями данных.

Система доставки, распространения (Deliver). Планирование и разработка → Преобразование (COLD/ERM, персонализация, XML, PDF, сжатие, конверторы) → Безопасность (открытые ключи, цифровые подписи) → Распространение (интернет, email, факсы, бумага)



Работа СЭД ->

9. Модель информационного пространства предприятия

Единая информационная система включает возможности: удаленной работы; доступа к информации (разные пользователи должны иметь доступ к одним и тем же данным без потерь производительности и независимо от своего местоположения в сети); средств коммуникации (электронная почта, факс, печать доков); сохранения целостности данных в общей БД; полнотекстового и реквизитного поиска информации; открытость системы (доступ пользователей к привычным средствам создания документов и к уже существующим документам); защищенность информации; удобства настройки на конкретные задачи пользователей; масштабируемость системы для поддержки роста организации и т. д.

Начальным этапом создания такой системы является построение модели предметной области (модели документооборота). Составить по трехмерному пространству свойств:



Рис. 1.1: Модель информационного пространства предприятия

Ось (F) – уровень организации хранения фактографической информации, которая привязана к специфике конкретного рода деятельности, компании или организации (например, при закупке мат. ценностей происходит оформление товарно-сопроводительных доков).

Ось (D) – отражает необходимость взаимодействия при формировании и передаче товаров, услуг или информации как внутри корпорации, так и вне ее.

Ось (R) – отражает регламент процессов прохождения документов (описание процедур: какие, как и когда должны выполняться).

Точка в пространстве (F, D, R) определяет состояние системы документооборота. Положение этой точки зависит от уровня развития и стадии внедрения системы, документооборота на предприятии, а также от его специфики и масштабов бизнеса.

10. Системы класса MRP, MRP II, ERP, CSRP

Система MRP (Material Requirements Planning) – система, работающая по алгоритму, регламентированному MRP методологией, позволяющую оптимально регулировать поставки комплектующих в производственный процесс, контролируя запасы на складе и саму технологию производства.

Главная задача MRP – обеспечение гарантии наличия необходимого количества требуемых материалов-комплектующих в любой момент времени в рамках срока планирования, наряду с возможным уменьшением постоянных запасов → разгрузкой склада.

Принцип работы MRP-модуля:

1. Для каждого отрезка времени создаётся полная потребность в материалах. Она представляет собой интегрированную таблицу, выражающую потребность в каждом материале, в каждый конкретный момент времени.
2. Вычисляется чистая потребность (какое количество материалов нужно заказать (или произвести, в случае внутреннего производства комплектующих) в каждый конкретный момент времени.
3. Чистая потребность в материалах конвертируется в соответствующий план заказов на требуемые материалы и, в случае необходимости, вносятся поправки в уже действующие планы.

Результатами работы MRP-модуля являются:

- План Заказов (Planned Order Schedule) – какое количество каждого материала должно быть заказано в каждый рассматриваемый период времени в течение срока планирования.
- Изменения к плану заказов (Changes in planned orders) – модификации к ранее спланированным заказам.

В концепции MRP есть серьезный недостаток. При расчете потребности в материалах не учитываются: производственные мощности, их загрузка; стоимость рабочей силы и т. д.

Поэтому в 80-х гг. MRP-система с замкнутым циклом была трансформирована в систему планирования производственных ресурсов (manufacture resource planning), которая получила название MRPII.

Система MRPII (manufactory resource planning) – это совместного планирования запасов и производственных ресурсов, характеризующаяся:

- бизнес-планированием;
- планированием продаж;
- планированием производства;
- планированием материальных потребностей;
- планированием производственных мощностей;
- различными системами управления.

Суть концепции MRPII: прогнозирование, планирование и контроль производства осуществляется по всему жизненному циклу продукции, начиная от закупки сырья и заканчивая отгрузкой продукции потребителю.

В результате применения MRPII-систем должны быть реализованы:

- оперативное получение информации о текущих результатах деятельности предприятия как в целом, так и с полной детализацией по отдельным заказам, видам ресурсов, выполнению планов;
- долгосрочное, оперативное и детальное планирование деятельности предприятия с возможностью корректировки плановых данных на основе оперативной информации;
- оптимизация производственных и материальных потоков со значительным сокращением непроизводственных затрат и реальным сокращением материальных ресурсов на складах;
- отражение финансовой деятельности предприятия в целом.

Примеры: Галактика 7.1, Concorde XAL, Platinum, Microsoft Dynamics, Scala.

Недостатки MRP-II: ориентация только на заказ, слабая интеграция конструирования и проектирования, слабая интеграция системы технологических процессов, слабая интеграция планирования кадров и управления финансами.

ERP (Enterprise Resource Planning) – это система, в основе которой лежит принцип создания единого хранилища данных (repository), содержащего всю деловую информацию, накопленную организацией в процессе ведения деловых операций. Это устраняет необходимость в передаче данных от системы к системе. Кроме того, любая часть информации, которой располагает данная организация, становится одновременно доступной для всех работников, обладающих соответствующими полномочиями.

Различие между концепциями MRP II и ERP заключается в том, что первая ориентирована на производство, а вторая – на бизнес. Например, такие вещи, как условия кредитования заказчика по отгрузке готовой продукции, попадают в поле зрения ERP, но не MRP II.

Основные функции ERP-системы:

- ведение конструкторских и технологических спецификаций. Такие спецификации определяют состав конечного изделия, а также материальные ресурсы и операции, необходимые для его изготовления (включая маршрутизацию);
- управление спросом и формирование планов продаж и производства. Эти функции предназначены для прогноза спроса и планирования выпуска продукции;
- планирование потребностей в материалах. Позволяют определить объемы различных видов материальных ресурсов (сырья, материалов, комплектующих), необходимых для выполнения производственного плана, а также сроки поставок, размеры партий и т. д.;
- управление запасами и закупочной деятельностью. Позволяют организовать ведение договоров, реализовать схему централизованных закупок, обеспечить учет и оптимизацию складских запасов и т. д.;
- планирование производственных мощностей. Эта функция позволяет контролировать наличие доступных мощностей и планировать их загрузку;
- финансовые функции. В эту группу входят функции финансового учета, управленческого учета, а также оперативного управления финансами. Обеспечивают планирование задач проекта и ресурсов, необходимых для их реализации.

CRM (Customer Relations Management) – это стратегия, основанная на применении таких управленческих и информационных технологий, с помощью которых компании аккумулируют знания о клиентах для выстраивания взаимовыгодных отношений с ними. Подобные отношения способствуют увеличению прибыли, т. к. привлекают новых клиентов и помогают удержать старых.

CSRP (customer synchronized resource planning) – это концепция, суть которой состоит в том, что при планировании и управлении компанией можно и нужно учитывать не только основные производственные и материальные ресурсы предприятия, но и все те, которые обычно рассматриваются как «вспомогательные» или «накладные». К таким ресурсам относят: ресурсы, потребляемые во время маркетинговой и «текущей» работы с клиентом, послепродажного обслуживания реализованных товаров, используемые для перевалочных и обслуживающих операций, а также внутрицеховые расходы.

Такой подход позволяет на порядок точнее управлять стоимостью товара, учитывая производство, продвижение и обслуживание товара данного типа, и учитывать все элементы его функционального жизненного цикла, а не только производства, как во всех стандартных системах предыдущих поколений.

Отличие ERP и CSRP в том, что в традиционном ERP происходит просто планирование ресурсов предприятия, а в CSRP планирование ресурсов, синхронизированное с покупателем.

11. Методология IDEF0. Основные характеристики и назначение. Нотация. Преимущества и недостатки применения

Функциональная модель предназначена для описания существующих бизнес-процессов, в котором используются как естественный, так и графический языки. Для передачи информации о конкретной системе источником графического языка является методология IDEF0.

Методология IDEF0 предписывает построение иерархической системы диаграмм – единичных описаний фрагментов системы.

Сначала проводится описание системы в целом и ее взаимодействия с окружающим миром (контекстная диаграмма), после чего проводится функциональная декомпозиция – система разбивается на подсистемы и каждая подсистема описывается отдельно (диаграммы декомпозиции). Затем каждая подсистема разбивается на более мелкие и так далее до достижения нужной степени подробности.

Каждая IDEF0-диаграмма содержит блоки и дуги:

- Блоки изображают функции моделируемой системы (на диаграммах изображаются прямоугольниками, обозначающими поименованные процессы, функции или задачи, которые происходят в течение определенного времени и имеют распознаваемые результаты). Имя работы должно быть выражено отглагольным существительным, обозначающим действие.
- Дуги связывают блоки вместе и отображают взаимодействия и взаимосвязи между ними.

IDEF0 требует, чтобы в диаграмме было не менее трех и не более шести блоков. Эти ограничения поддерживают сложность диаграмм и модели на уровне, доступном для чтения, понимания и использования.

Каждая сторона блока имеет вполне определенное назначение. Левая сторона блока предназначена для входов, верхняя – для управления, правая – для выходов, нижняя – для механизмов.

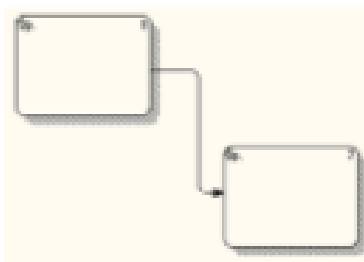
Блоки в IDEF0 размещаются по степени важности, как ее понимает автор диаграммы. Этот относительный порядок называется доминированием. Наиболее доминирующий блок обычно размещается в верхнем левом углу диаграммы, а наименее доминирующий – в правом углу.

Взаимодействие работ с внешним миром и между собой описывается в виде стрелок, изображаемых одинарными линиями со стрелками на концах. Стрелки представляют собой некую информацию и именуются существительными.

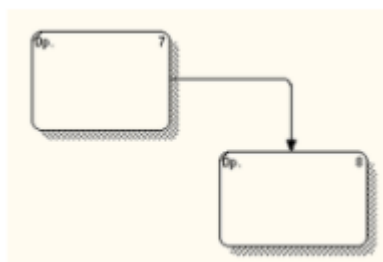
Типы стрелок:

- **Вход** – объекты, используемые и преобразуемые работой для получения результата (выхода). Допускается, что работа может не иметь ни одной стрелки входа. Стрелка входа рисуется как входящая в левую грань работы.
- **Управление** – информация, управляющая действиями работы. Обычно управляющие стрелки несут информацию, которая указывает, что должна выполнять работа. Каждая работа должна иметь хотя бы одну стрелку управления, которая изображается как входящая в верхнюю грань работы.
- **Выход** – объекты, в которые преобразуются входы. Каждая работа должна иметь хотя бы одну стрелку выхода, которая рисуется как исходящая из правой грани работы.
- **Механизм** – ресурсы, выполняющие работу. Стрелка механизма рисуется как входящая в нижнюю грань работы. По усмотрению аналитика стрелки механизма могут не изображаться на модели.
- **Вызов** – специальная стрелка, указывающая на другую модель работы. Стрелка вызова рисуется как исходящая из нижней части работы и используется для указания того, что некоторая работа выполняется за пределами моделируемой системы.

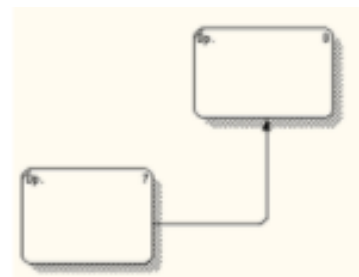
Типы взаимодействий между блоками:



Связь по входу

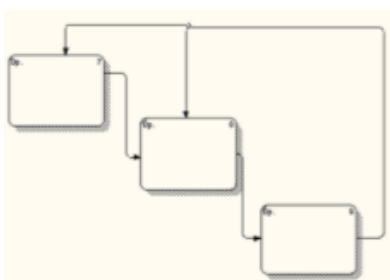


Связь по управлению



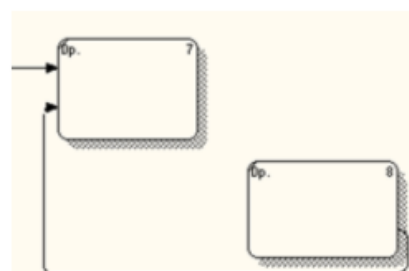
Связь выход-механизм

(распределение источников ресурсов)



Обратная связь по управлению

(когда выход некоторого блока влияет



Обратная связь по входу

(когда выходы из одной работы влияют на

на блок с большим доминированием)

будущее выполнения других работ)

Преимущества IDEF0:

- широкая известность стандарта среди аналитиков, консультантов и программистов.
- относительная простота изучения и применения стандарта при описании

бизнес-процессов.

- лаконичность и высокая информативность получаемых моделей.
- стандарт соответствует требованиям ISO 9000.

Недостатки IDEF0:

- сложность восприятия (большое количество стрелок).
- большое количество уровней декомпозиции.
- трудность увязки нескольких процессов, представленных в различных моделях одной и

той же организации.

- методология IDEF0 рассматривает систему как совокупность взаимосвязанных работ, что плохо отражает процессы обработки информации (в данном случае лучше использовать DFD)

12. Методология IDEF3. Основные характеристики и назначение. Нотация. Преимущества и недостатки применения

Диаграммы IDEF3 (Workflow diagramming) – методология моделирования, использующая графическое описание информационных потоков, взаимоотношений между процессами обработки информации и объектов, являющихся частью этих процессов.

Диаграммы Workflow используются для анализа процедур обработки информации.

Цель IDEF3 – дать аналитикам описание последовательности выполнения процессов, а также объектов, участвующих совместно в одном процессе. IDEF3 дополняет IDEF0 и содержит все необходимое для построения моделей, которые могут быть использованы для имитационного моделирования.

В IDEF3 работы изображаются прямоугольниками и имеют имя, обозначающее процесс действия и номер (идентификатор). В имя обычно включается основной результат работы (например, приготовление обеда).

Связи в IDEF3 показывают взаимоотношения работ и являются однонаправленными. Типы связей:

- Старшая (Precedence) линия – сплошная линия, связывающая единица работ. Рисуется слева направо или сверху вниз. Показывает, что работа-источник должна закончиться прежде, чем работа-цель начнется.
- Линия отношения (Relation Link) – пунктирная линия, использующаяся для изображения связей между единицами работ, а также между единицами работ и объектами ссылок.
- Потоки объектов (Object Flow) – стрелка с двумя наконечниками, применяется для описания использования объекта в двух или более единицах работы, например когда объект порождается в одной работе и используется в другой.
- Перекрестки (Junction) – используются для отображения логики взаимодействия стрелок при слиянии и разветвлении или для отображения множества событий, которые могут или должны быть завершены перед началом следующей работы. Различают перекрестки для слияния (Fan-in Junction) и разветвления (Fanout Junction) стрелок. Перекресток не может использоваться одновременно для слияния и для разветвления.

Типы перекрестков:

Обозначение	Наименование	Смысл в случае слияния стрелок	Смысл в случае разветвления стрелок
 AND	Asynchronous AND	Все предшествующие процессы должны быть завершены	Все следующие процессы должны быть запущены
 AND	Synchronous AND	Все предшествующие процессы завершены одновременно	Все следующие процессы запускаются одновременно
 OR	Asynchronous OR	Один или несколько предшествующих процессов должны быть завершены	Один или несколько следующих процессов должны быть запущены
 OR	Synchronous OR	Один или несколько предшествующих процессов завершены одновременно	Один или несколько следующих процессов запускаются одновременно
 XOR	XOR(Exclusive OR)	Только один процесс завершен	Только один следующий процесс запускается

Объекты-ссылки являются специальными символами, которые ссылаются на внешние части описания процесса. Они добавляются на диаграмму для того, чтобы обратить внимание редактора на что-либо важное, что невозможно связать со стрелкой, работой или перекрестком. Объекты-ссылки должны быть связаны с единицами работ или перекрестками пунктирными линиями. При внесении объектов-ссылок необходимо указать их тип.

Типы объектов-ссылок:





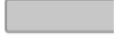
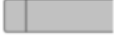


Тип объекта-ссылки	Цель описания
OBJECT	Описывает участие важного объекта в работе
GOTO	Инструмент циклического перехода (в повторяющейся последовательности работ), возможно на текущей диаграмме, но не обязательно. Если все работы цикла присутствуют на текущей диаграмме, цикл может также изображаться стрелкой, возвращающейся на стартовую работу. GOTO может ссылаться на перекресток
UQB (Unit of behavior)	Применяется, когда необходимо подчеркнуть множественное использование какой-либо работы, но без цикла. Например, работа «Контроль качества» может быть использована в процессе «Изготовления изделия» несколько раз, после каждой единичной операции. Обычно этот тип ссылки не используется для моделирования автоматически запускающихся работ
NOTE	Используется для документирования важной информации, относящейся к каким-либо графическим объектам на диаграмме. NOTE является альтернативой внесению текстового объекта в диаграмму
ELAB (Elaboration)	Используется для усовершенствования графиков или их более детального описания. Обычно употребляется для детального описания разветвления и слияния стрелок на перекрестках

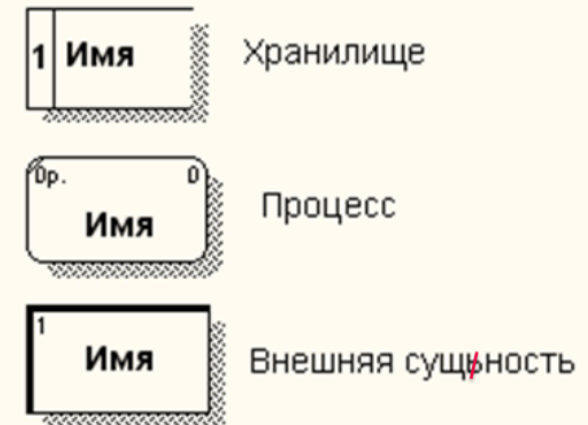
13. Методология DFD. Основные характеристики и назначение. Нотация. Преимущества и недостатки применения

DFD описывает:

- функции обработки информации (работы);
- документы (стрелки, arrows), объекты, сотрудников или отделы, которые участвуют в обработке информации;
- внешние ссылки (external reference), которые обеспечивают интерфейс с внешними объектами, находящимися за границами моделируемой системы;
- таблицы для хранения документов (хранилища данных, data store).

Для построения диаграмм DFD обычно используется нотация Гейна-Сарсона. Есть два варианта синтаксиса нотации:

Нотация	Юрдан и Коад	Гейн и Сарсон
Внешняя сущность		
Процесс		
Хранилище данных		
Поток данных		



Потоки данных являются механизмами, используемыми для моделирования передачи информации (или физических компонентов) из одной части системы в другую. Потоки изображаются на диаграмме именованными стрелками, ориентация которых указывает направление движения информации. Стрелки могут подходить к любой грани прямоугольника работы и могут быть двунаправленными для описания взаимодействия типа «команда-ответ».

Назначение процесса состоит в продуцировании выходных потоков из входных в соответствии с действием, задаваемым именем процесса. Каждый процесс должен иметь уникальный номер для ссылок на него внутри диаграммы.

Хранилище данных позволяет на определенных участках определять данные, которые будут сохраняться в памяти между процессами. Хранилище представляет «срезы» потоков данных во времени. Информация, которую оно содержит, может использоваться в любое время после ее определения, при этом данные могут выбираться в любом порядке. Имя хранилища

должно идентифицировать его содержимое. В случае, когда поток данных входит в хранилище или выходит из него и его структура соответствует структуре хранилища, он должен иметь то же самое имя, которое нет необходимости отражать на диаграмме.

Внешняя сущность представляет сущность вне контекста системы, являющуюся источником или приемником данных системы. Предполагается, что объекты, представленные такими узлами, не должны участвовать ни в какой обработке. Внешние сущности изображаются в виде прямоугольника с тенью и обычно располагаются по краям диаграммы. Одна внешняя сущность может быть использована многократно на одной или нескольких диаграммах.

14. Язык моделирования UML. Понятие объекта и класса. Визуальное представление объектов и классов в UML. Атрибуты и операции объектов и классов. Нотации отношений в UML

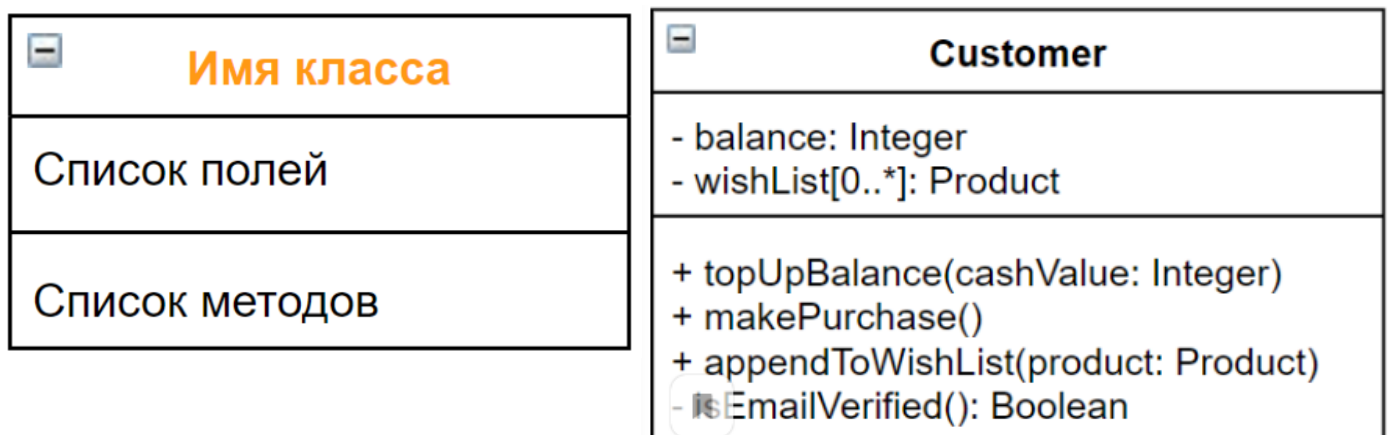
UML (Unified Modeling Language) – Унифицированный Язык Моделирования. Объектно-ориентированный графический язык для визуализации, специфицирования, конструирования и документирования систем, где большая роль отводится описанию бизнес-процессов в информационных системах. Разработан группой объектного проектирования OMG (Object Management Group). Получил статус отраслевого стандарта.

Цели создания UML:

- Предоставить пользователям готовый к использованию язык моделирования;
- Предоставить механизмы расширения и специализации;
- Быть независимым от определенного языка программирования и процесса разработки;
- Интегрировать лучший практический опыт разработок.

Класс (class) – категория вещей, которые имеют общие атрибуты и операции. Классы – это строительные блоки любой объектно-ориентированной системы. Они представляют собой описание совокупности объектов с общими атрибутами, операциями, отношениями и семантикой. При проектировании объектно-ориентированных систем диаграммы классов обязательны.

Изображается класс на диаграмме в виде прямоугольника, разделенного на три секции: Имя класса; Список полей класса; Список методов класса. Обычно в качестве имени класса выбирается существительное в единственном числе. Пример:



Объект (object) – экземпляр класса. Объект – конкретная материализация абстракции.

Как же обозначается объект в UML? А очень просто – объект, как и класс, обозначается прямоугольником, но его имя подчеркивается. Под словом имя здесь мы понимаем название объекта и наименование его класса, разделенные двоеточием. Для указания значений атрибутов объекта в его обозначении может быть предусмотрена специальная секция. Еще один нюанс состоит в том, что объект может быть анонимным: это нужно в том случае, если в данный момент не важно, какой именно объект данного класса принимает участие во взаимодействии. Примеры:



Операция (метод) – это реализация метода класса. Класс может иметь любое число операций либо не иметь ни одной. Часто вызов операции объекта изменяет его атрибуты. Графически операции представлены в нижнем блоке описания класса. Допускается указание только имен операций. Имя операции, как и имя класса, должно представлять собой текст. На практике для именования операции используются короткие глагольные конструкции, описывающие некое поведение класса, которому принадлежит операция. Обычно каждое слово в имени операции пишется с заглавной буквы, за исключением первого, например move (переместить) или isEmpty (проверка на пустоту). Можно специфицировать операцию, устанавливая ее сигнатуру, включающую имя, тип и значение по умолчанию всех параметров, а применительно к функциям – тип возвращаемого значения.

Атрибут (свойство) – это именованное свойство класса, описывающее диапазон значений, которые может принимать экземпляр атрибута. Класс может иметь любое число атрибутов или не иметь ни одного. В последнем случае блок атрибутов оставляют пустым. Атрибут представляет некоторое свойство моделируемой сущности, которым обладают все объекты данного класса. Имя атрибута, как и имя класса, может представлять собой текст. На практике для именования атрибута используются одно или несколько коротких существительных, выражающих некое свойство класса, к которому относится атрибут. Можно уточнить спецификацию атрибута, указав его тип, кратность (если атрибут представляет собой массив некоторых значений) и начальное значение по умолчанию. Статические атрибуты класса обозначаются подчеркиванием.

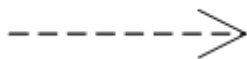
Отношения между классами

Существует четыре типа связей в UML:

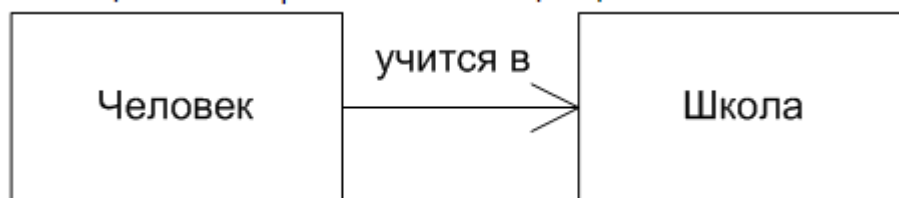
- Зависимость
- Ассоциация
- Обобщение
- Реализация

Эти связи представляют собой базовые строительные блоки для описания отношений в UML, используемые для разработки хорошо согласованных моделей.

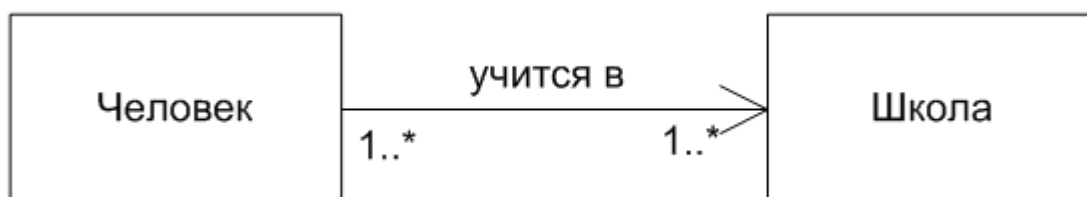
Первая из них — **зависимость** — семантически представляет собой связь между двумя элементами модели, в которой изменение одного элемента (независимого) может привести к изменению семантики другого элемента (зависимого). Графически представлена пунктирной линией, иногда со стрелкой, направленной к той сущности, от которой зависит еще одна; может быть снабжена меткой.



Ассоциация — это структурная связь между элементами модели, которая описывает набор связей, существующих между объектами. Ассоциация показывает, что объекты одной сущности (класса) связаны с объектами другой сущности таким образом, что можно перемещаться от объектов одного класса к другому. Например, класс Человек и класс Школа имеют ассоциацию, так как человек может учиться в школе. Ассоциации можно присвоить имя «учится в». В представлении однонаправленной ассоциации добавляется стрелка, указывающая на направление ассоциации.



(бывает множественной)



*****ДОПОЛНЕНИЕ К АССОЦИАЦИИ*****

Агрегация – особая разновидность ассоциации, представляющая структурную связь целого с его частями. Как тип ассоциации, агрегация может быть именованной. Одно отношение агрегации не может включать более двух классов (контейнер и содержимое). Агрегация встречается, когда один класс является коллекцией или контейнером других. Причём, по умолчанию агрегацией называют агрегацию по ссылке, то есть когда время существования содержащихся классов не зависит от времени существования содержащего их класса. Если контейнер будет уничтожен, то его содержимое — нет. Графически агрегация представляется пустым ромбом на блоке класса «целое», и линией, идущей от этого ромба к классу «часть».



Композиция — более строгий вариант агрегации. Известна также как агрегация по значению. Композиция – это форма агрегации с четко выраженными отношениями владения и совпадением времени жизни частей и целого. Композиция имеет жёсткую зависимость времени существования экземпляров класса контейнера и экземпляров содержащихся классов. Если контейнер будет уничтожен, то всё его содержимое будет также уничтожено. Графически представляется как и агрегация, но с закрашенным ромбиком.



Третья связь – **обобщение** – выражает специализацию или наследование, в котором специализированный элемент (потомок) строится по спецификациям обобщенного элемента (родителя). Потомок разделяет структуру и поведение родителя. Графически обобщение представлено в виде сплошной линии с пустой стрелкой, указывающей на родителя.



Четвертая – **реализация** – это семантическая связь между классами, когда один из них (поставщик) определяет соглашение, которого второй (клиент) обязан придерживаться. Это связи между интерфейсами и классами, которые реализуют эти интерфейсы. Это, своего рода, отношение «целое-часть». Поставщик, как правило, представлен абстрактным классом. В графическом исполнении связь реализации – это гибрид связей обобщения и зависимости: треугольник указывает на поставщика, а второй конец пунктирной линии – на клиента.



15. Виды и нотация диаграмм UML. Характеристики видов диаграмм. Цели их разработки

Диаграммы языка UML:

- вариантов использования или прецедентов (use case diagram)
- классов (class diagram)
- объектов (object diagram)
- состояний (statechart diagram)
- активности (activity diagram)
- последовательности (sequence)
- взаимодействия (collaboration)
- компонентов (component diagram)
- развертывания (deployment diagram)
- композитная структурная диаграмма
- обзорная диаграмма взаимодействия
- временная диаграмма
- диаграмма пакетов

1. **Диаграмма вариантов использования (use case diagram)** — диаграмма, на которой изображаются отношения между актерами и вариантами использования (прецедентами).

Диаграммы прецедентов описывают функциональное назначение системы (то, что система будет делать в процессе своего функционирования). Диаграммы прецедентов являются исходной концептуальной моделью системы в процессе ее проектирования и разработки.

Цели создания use case диаграммы:

- Определить общие границы и контекст моделируемой предметной области на начальных этапах проектирования системы;
- Сформулировать общие требования к функц-ному поведению проектируемой системы;
- Разработать исходную концептуальную модель системы для ее последующей детализации в форме логических и физических моделей;
- Подготовить исходную документацию для взаимодействия разработчиков системы с ее заказчиками и пользователями.

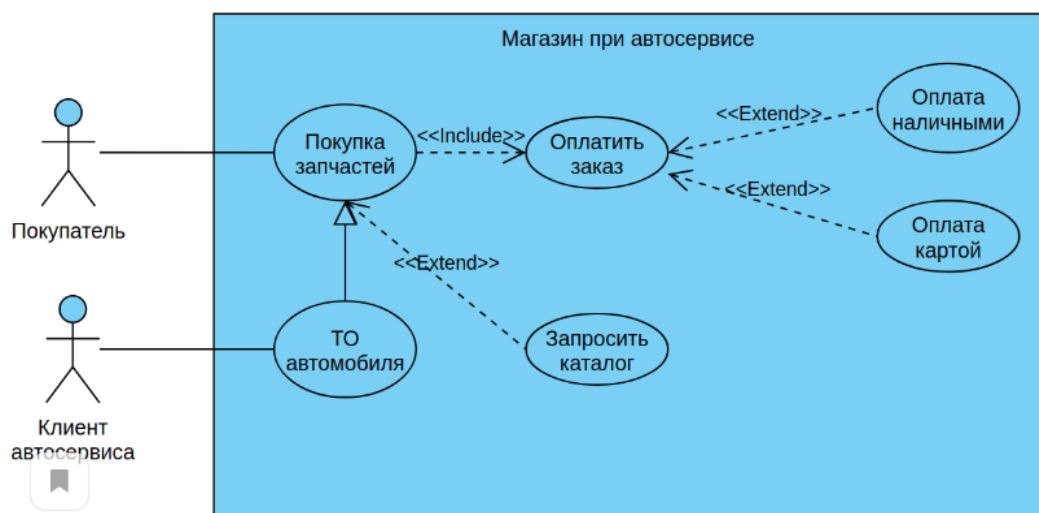
Назначение данной диаграммы состоит в следующем: проектируемая информационная система представляется в форме так называемых вариантов использования, с которыми взаимодействуют внешние сущности или акторы. При этом актером или действующим лицом называется любой объект, субъект или система, взаимодействующая с моделируемой бизнес-системой извне. Это может быть человек, техническое устройство, программа или

любая другая система, которая служит источником воздействия на моделируемую систему так, как определит разработчик. Вариант использования служит для описания сервисов, которые система предоставляет актеру. Другими словами каждый вариант использования определяет набор действий, совершаемый системой при диалоге с актером.

В этой диаграмме есть следующие элементы: вариант использования — последовательность действий, которые система или другая сущность могут выполнять в процессе взаимодействия с актерами (обозначается эллипсом, внутри которого содержится его краткое имя в форме отглагольного существительного), актер — согласованное множество ролей, которые играют внешние сущности по отношению к вариантам использования (обозначается человечком, под которым записывается имя актера с заглавной буквы).

Отношения на диаграмме вариантов использования: ассоциации — служит для обозначения специфической роли актера при его взаимодействии с отдельным вариантом использования (обозначается сплошной линией между актером и вариантом использования), включения (include) — устанавливается только между двумя вариантами использования и указывает на то, что заданное поведение для одного варианта использования включается в качестве составного фрагмента в последовательность поведения другого варианта использования (обозначается как отношение зависимости в форме пунктирной линии со стрелкой, направленной от базового варианта использования к включаемому варианту использования), расширения (extend) — является зависимостью, направленной к базовому варианту использования и соединенной с ним в так называемой точке расширения (обозначается как отношение зависимости в форме пунктирной линии со стрелкой, направленной от того варианта использования, который является расширением для базового варианта использования), обобщения — если актеры имеют общие свойства, т.е. взаимодействуют с одним и тем же множеством вариантов использования одинаковым образом (обозначается сплошной линией со стрелкой в форме незакрашенного треугольника, которая указывает на родительский вариант использования).

Пример:



2. **Диаграмма классов (class diagram)** предназначена для представления внутренней структуры программы в виде классов и связей между ними.

Целью создания диаграммы классов является графическое представление статической структуры декларативных элементов системы (классов, типов и т. п.) Она содержит в себе также некоторые элементы поведения (например — операции), однако их динамика должна быть отражена на диаграммах других видов (диаграммах коммуникации, диаграммах состояний).

3. **Диаграмма объектов (object diagram)** предназначена для демонстрации совокупности моделируемых объектов и связей между ними в фиксированный момент времени.

Цель диаграммы объектов-показать отношения между набором объектов (т. е. ссылки/указатели между ними)

4. **Диаграмма активности/диаграмма деятельности (activity diagram)** — диаграмма, на которой показаны действия, состояния которых описаны на диаграмме состояний. Такая диаграмма описывает процесс выполнения действий, т.е. логику и последовательность перехода от одного действия к другому. Диаграмма деятельности используется для моделирования бизнес-процессов.

Разработка диаграммы деятельности преследует цели:

- Детализировать особенности алгоритмической и логической реализации выполняемых системой операций и прецедентов;
- Выделить последовательные и параллельные потоки управления;
- Подготовить детальную документацию для взаимодействия разработчиков системы с ее заказчиками и проектировщиками.

5. **Диаграмма состояний (statechart diagram)** — это тип диаграммы, используемый в UML для описания поведения систем, который основан на концепции диаграмм состояний Дэвида Харела. Диаграммы состояний отображают разрешенные состояния и переходы, а также события, которые влияют на эти переходы. Она помогает визуализировать весь жизненный цикл объектов и, таким образом, помогает лучше понять системы, основанные на состоянии.

Цель: для абстрактного описания поведения системы.

6. **Диаграмма последовательности (sequence)** — это диаграмма, на которой для некоторого набора объектов на единой временной оси показан жизненный цикл объекта (создание- деятельность- уничтожение некой сущности) и взаимодействие актеров (действующих лиц) информационной системы в рамках прецедента.

Целью диаграммы последовательности в UML является визуализация последовательности потока сообщений в системе. Диаграмма последовательности показывает взаимодействие между двумя линиями жизни в виде упорядоченной по времени последовательности событий.

7. **Диаграмма взаимодействия (collaboration)** используется в UML для установления связи между объектами. Он не манипулирует данными, связанными с конкретным каналом связи. Диаграммы взаимодействия в основном сосредоточены на передаче сообщений и на том, как эти сообщения составляют одну функциональность системы. Диаграммы взаимодействия предназначены для отображения того, как объекты будут реализовывать определенные требования системы. Критическим компонентом в диаграмме взаимодействия является линия жизни и сообщения.

Целью диаграмм взаимодействия является визуализация интерактивного поведения системы.

8. **Диаграмма компонентов (component diagram)** используется для визуализации организации компонентов системы и зависимостей между ними. Такая диаграмма позволяет получить высокоуровневое представление о компонентах системы. Компонентами могут быть программные компоненты, такие как база данных или пользовательский интерфейс; или аппаратные компоненты, такие как схема, микросхема или устройство; или бизнес-подразделение, такое как поставщик, платежная ведомость или доставка.

Диаграмма компонентов разрабатывается для следующих целей:

- Визуализации общей структуры исходного кода программной системы;
- Спецификации исполняемого варианта программной системы;
- Обеспечения многократного использования отдельных фрагментов программного кода;
- Представления концептуальной и физической схем баз данных.

9. **Диаграмма развертывания (deployment diagram)** — это тип UML-диаграммы, которая показывает архитектуру исполнения системы, включая такие узлы, как аппаратные или программные среды исполнения, а также промежуточное программное обеспечение, соединяющее их.

При разработке диаграмм развертывания преследуются следующие цели:

- Специфицировать физические узлы, необходимые для размещения на них исполняемых компонентов программной системы;
- Показать физические связи между узлами реализации системы на этапе ее исполнения;
- Выявить узкие места системы и реконфигурировать ее топологию для достижения требуемой производительности.

10. **Композитная структурная диаграмма** — статическая структурная диаграмма, демонстрирует внутреннюю структуру классов и, по возможности, взаимодействие элементов (частей) внутренней структуры класса. Подвидом диаграмм композитной структуры являются диаграммы кооперации (Collaboration diagram, введены в UML 2.0), которые показывают роли и взаимодействие классов в рамках кооперации. Кооперации удобны при моделировании шаблонов проектирования.

11. **Обзорная диаграмма взаимодействия** является разновидностью диаграммы деятельности с расширенным синтаксисом: в качестве элементов обзорной диаграммы взаимодействия могут выступать ссылки на взаимодействия (interaction use) 1, определяемые диаграммами последовательности.

12. **Временная диаграмма** — это диаграмма взаимодействия UML, используемая для отображения взаимодействий, когда основная цель диаграммы — рассуждать о времени. Такая диаграмма сосредоточена на изменении условий внутри и между линиями жизни вдоль линейной оси времени.

13. **Диаграмма пакетов** используется для упрощения сложных диаграмм классов, вы можете группировать классы в пакеты. Пакет — это набор логически связанных элементов UML. Диаграмма пакетов следует иерархической структуре вложенных пакетов.

16. Основные компоненты сети. Сетевые устройства

Все устройства, подключаемые к сети, можно разделить на следующие функциональные группы с точки зрения их отношения к ресурсам сети:

- Сервер (server);
- Рабочая станция (клиентский компьютер, клиент);
- Терминал;
- Коммуникационное оборудование.

Сервер

Это специально выделенный высокопроизводительный компьютер, управляющий работой сети и/или предоставляющий другим компьютерам сети свои ресурсы (программное обеспечение, сервисы, файлы, устройства) и отвечающий на запросы клиентов.

Различают:

- Файловые серверы (file server) – компьютеры с большой емкостью памяти, предназначенные для хранения данных пользователей сети и обеспечения доступа к ним;
- Серверы баз данных (database server) – компьютеры с (СУБД), предназначенные для хранения и обработки огромных массивов данных;
- Сервер прикладных программ (application server) - обеспечивает выполнение прикладных программ для пользователей, работающих на своих рабочих станциях;
- Сервер резервного копирования данных (backup server) - обеспечивает создание, хранение и восстановление копий данных, расположенных на файловых серверах и рабочих станциях;
- Серверы печати (print server) – компьютеры со специальным ПО, предназначенные для организации процесса печати;

Все перечисленные типы серверов могут функционировать на одном выделенном для этих целей компьютере.

Рабочая станция

Это компьютер рядового пользователя сети, получающий доступ к ресурсам сервера.

Каждая рабочая станция обрабатывает свои локальные файлы и использует свою операционную систему.

Терминал

Устройство не предназначено для работы в автономном режиме (не имеет процессора для обработки команд), но выполняет операции по вводу команд пользователя, их передаче другому компьютеру и выдаче готового результата.

Коммуникационное оборудование

Технические средства компьютерных сетей включают в себя различные функциональные группы оборудования:

- средства линий передачи данных (кабель "витая пара", оптоволоконный и пр.) – реализуют собственно перенос сигнала;
- средства соединения линий передачи с сетевым оборудованием узлов (сетевые платы) – реализуют ввод-вывод данных с оконечного оборудования в сеть;
- средства увеличения дистанции передачи данных – репитеры (или повторители), модемы и пр. – осуществляют усиление сигналов или преобразования в форму, удобную для дальнейшей передачи;
- средства повышения емкости линий передачи (мультиплексирования) – позволяют реализовывать несколько логических каналов в рамках одного физического соединения путем разделения частот передачи, чередования пакетов во времени и т.д.;
- средства управления информационными потоками в сети (коммутации каналов, коммутации пакетов, разветвления линий передачи) – осуществляют адресацию сообщений. Например, концентраторы, коммутаторы, маршрутизаторы, шлюзы.

Устройства

- Сетевой адаптер - устройство, позволяющее компьютеру взаимодействовать с другими устройствами сети.
- Модем - устройство, которое преобразует данные из цифрового формата в формат, подходящий для аналоговой среды передачи, такой как телефон или радио.
- Репитер - устройство, восстанавливающее форму сигнала, искаженного прохождением в длинной линии. Они служат простыми двунаправленными ретрансляторами. Основная их цель – увеличение длины сети.

- Трансивер - устройство, служащее для двунаправленной передачи между адаптером и сетевым кабелем или между двумя сегментами (отрезками) сетевого кабеля.

- Шлюз - устройство, служащее для соединения совершенно разных сетей, например, локальных сетей с глобальными, или локальных сетей с мейнфреймами, использующими совершенно другие правила обмена. В этом случае полностью преобразуется весь поток информации, т. е. коды, форматы, методы управления и т.д.

- Концентратор (англ. Hub) – разветвительное устройство, служащее центральным звеном в локальных сетях, имеющих топологию "звезда". Концентратор имеет несколько портов для подключения отдельных компьютеров и для соединения с другими хабами.

- Коммутатор (англ. Switch) – в переводе с англ. означает переключатель. Это многопортовое устройство, обеспечивающее высокоскоростную коммутацию пакетов между портами. Встроенное в него программное обеспечение способно самостоятельно анализировать содержимое пересылаемых по сети блоков данных и обеспечивать прямую передачу информации между любыми двумя портами, независимо от всех остальных портов устройства.

- Мост (англ. bridge) – устройство, соединяющее одинаковые сети, имеющие некоторые физические различия (на физическом и канальном уровнях).

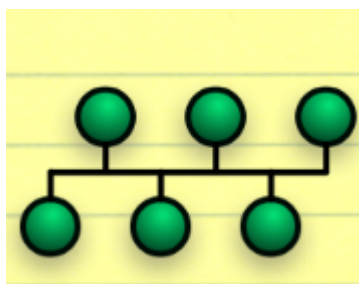
- Маршрутизатор (англ. router) – устройство, соединяющее сети одного или разных типов по одному протоколу обмена данными. Маршрутизатор анализирует адрес назначения и направляет данные по оптимально выбранному маршруту.

17. Основные сетевые топологии

Физическая топология описывает: схему прокладки кабеля, расположение узлов и взаимосвязи между ними. Физическая топология сети определяется возможностями устройств доступа, желаемым уровнем контроля и толерантности к ошибкам, а также стоимостью всех необходимых материалов.

Логическая топология описывает поведение сигнала в сети или путь, которым движутся данные в сети от одного устройства к другому, независимо от их физической взаимосвязи. В многих случаях логическая топология не совпадает с физической, она может динамически изменяться в соответствии с изменениями в конфигурации маршрутизаторов и коммутаторов.

Топология с общей шиной (Bus topology)



На концах кабеля находятся терминаторы, для предотвращения отражения сигнала.

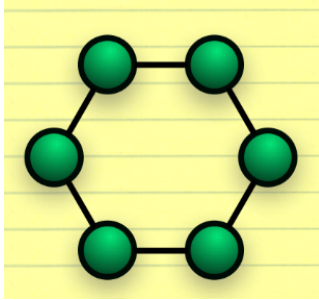
Преимущества:

- Простая.

Недостатки:

- Ненадежная (один разрыв лежит вся сеть);
- Выход из строя рабочей станции не отражается на работе сети; (это, наверное плюс)
- С добавлением новых рабочих станций падает производительность сети.

Кольцевая топология (Ring topology)



Это топология, в которой каждый компьютер соединен линиями связи только с двумя другими: от одного он только получает информацию, а другому только передает. Это позволяет отказаться от применения внешних терминаторов.

Преимущества:

- Простота установки;
- Практически полное отсутствие дополнительного оборудования.

Недостатки:

- Выход из строя одной рабочей станции, и другие неполадки (обрыв кабеля), отражаются на работоспособности всей сети;
- Сложность конфигурирования и настройки;
- Сложность поиска неисправностей.



Топология "Звезда" (Star topology)

Топология компьютерной сети, в которой все компьютеры сети присоединены к центральному узлу (обычно коммутатор), образуя физический сегмент сети.

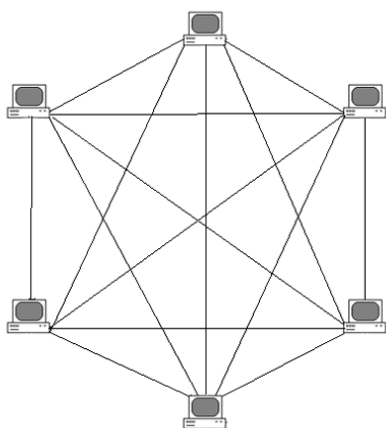
Преимущества:

- Выход из строя одной рабочей станции не отражается на работе всей сети в целом;
- Хорошая масштабируемость сети;
- Лёгкий поиск неисправностей и обрывов в сети;
- Высокая производительность сети (при условии правильного проектирования);
- Гибкие возможности администрирования.

Недостатки:

- Выход из строя центрального концентратора обернётся неработоспособностью сети (или сегмента сети) в целом;
- Для прокладки сети зачастую требуется больше кабеля, чем для большинства других топологий;
- Конечное число рабочих станций в сети (или сегменте сети) ограничено количеством портов в центральном концентраторе.

Полносвязная топология (Full-mesh topology)



Топология компьютерной сети, в которой каждая рабочая станция подключена ко всем остальным.

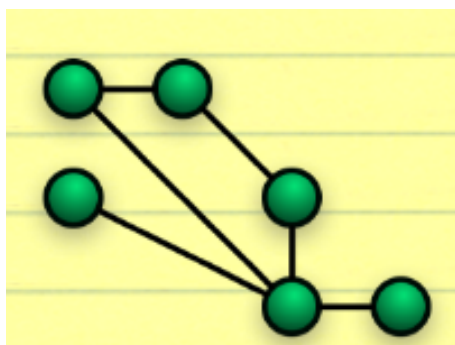
Преимущества:

- Чрезвычайная отказоустойчивость.

Недостатки:

- Дорого;
- Каждому компьютеру нужно много портов.

Неполносвязная топология (Partial-mesh topology)



Получается из полносвязной путем удаления некоторых возможных связей.

Преимущества:

- Отказоустойчивость.

Недостатки:

- Сложность настройки.

Смешанная топология (Mixed topology)

Звезда-Шина

ИТМО

Данная топология также называется "дерево".

Эта сеть состоит из нескольких подсетей с топологией "звезда, соединенных единой шиной.

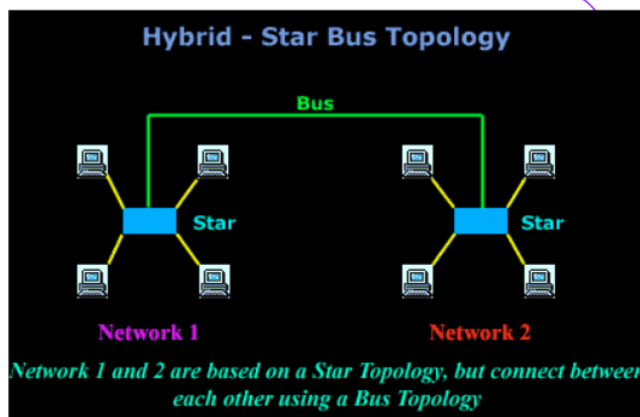
Преимущества:

- простота расширения
- простота поиска обрывов и неисправностей
- устойчивость к неисправностям отдельных компьютеров

Недостатки:

- неустойчивость к неисправности шины и хабов

Логическая топология данной сети - классическая шина.



18. Типы сетевой инфраструктуры и их характеристики

Для классификации компьютерных сетей используются различные признаки:

- Территориальная распространенность;
- Ведомственная принадлежность;
- Скорость передачи информации;
- Тип среды передачи.

Классификация по территориальному признаку:

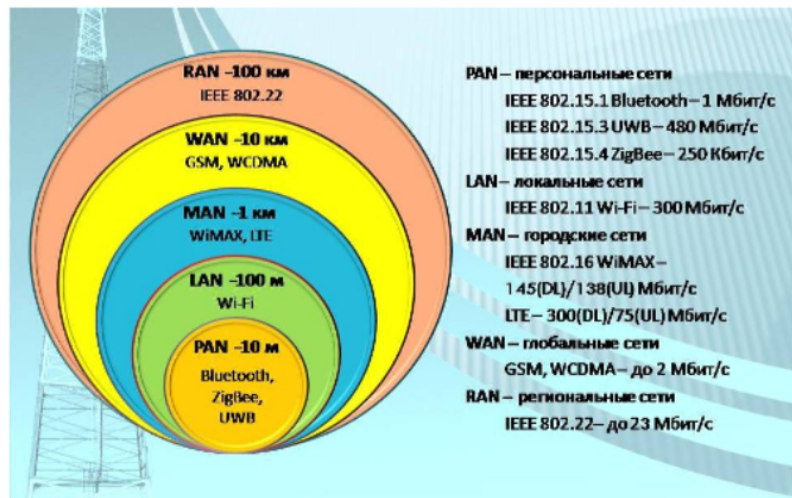
- Глобальные сети – World Area Networks (WAN). Объединяют территориально рассредоточенные компьютеры, которые могут находиться в различных городах и странах. Охватывает большие территории и включает в себя большое число компьютеров.
- Городские сети – Metropolitan Area Networks (MAN). Предназначены для обслуживания территории крупного города – мегаполиса.
- Корпоративные (сети предприятий) – Enterprise Wide Networks(EWN). Объединяют большое количество компьютеров в территориально распределенных филиалах отдельного предприятия.
- Локальные – Local Area Networks (LAN). К локальным сетям относятся сети компьютеров, сосредоточенные на небольшой территории (обычно в радиусе 1-2 км). В общем случае это коммуникационная система, принадлежащая одной организации.
- Персональные – Personal Area Networks (PAN). К персональным сетям относятся сети, предназначенные для взаимодействия устройств, принадлежащих одному владельцу на небольшом расстоянии (обычно до 10 м).

Фоточка на следующей странице про классификацию по территориальному признаку:

Классификация компьютерных сетей по территориальному признаку

ИТМО

Институт инженеров электротехники и электроники — IEEE (*Institute of Electrical and Electronics Engineers*) — некоммерческая инженерная ассоциация из США, разрабатывающая стандарты по радиоэлектронике, электро технике и аппаратному обеспечению вычислительных систем и сетей.



По принадлежности различают:

- Ведомственные сети (принадлежат одной организации и располагаются на ее территории, используются коммерческими бизнес-приложениями);
- Государственные сети (используются в государственных структурах государственными бизнес-приложениями).

По скорости передачи информации компьютерные сети делятся на

- Низкоскоростные (до 10 Мбит/с);
- Среднескоростные (до 100 Мбит/с);
- Высокоскоростные (свыше 100 Мбит/с).

По типу среды передачи разделяются на:

- Сети коаксиальные;
- На витой паре;
- Оптоволоконные;
- С передачей информации по радиоканалам;
- В инфракрасном диапазоне.

Сетевые инфраструктуры могут значительно отличаться по следующим критериям:

- Площадь покрытия;
- Количество подключенных пользователей;
- Количество и типы доступных услуг;
- Область ответственности.

Распространённые типы сетевой инфраструктуры.

- Локальная сеть (LAN) — сетевая инфраструктура, предоставляющая доступ пользователям и оконечным устройствам на небольшой территории; обычно является домашней сетью, сетью малого или крупного предприятия, управляется одним лицом или ИТ-отделом и принадлежит им;
- Глобальная сеть (WAN) — сетевая инфраструктура, предоставляющая доступ к другим сетям на большой территории; обычно принадлежит провайдерам телекоммуникационных услуг и находится под их управлением.

Другие типы сетей:

- Городская сеть (Metropolitan Area Network, MAN) — сетевая инфраструктура, которая охватывает территорию больше, чем локальная сеть, но меньше глобальной сети (например, город). Как правило, управляет городскими сетями одна организация, например крупный сетевой оператор;
- Беспроводные локальные сети (WLAN) — аналогичны локальным сетям, но соединяют пользователей и оконечные устройства на небольшой территории с помощью беспроводной связи;
- Сеть хранения данных (SAN) — сетевая инфраструктура, разработанная для поддержки файловых серверов, хранения данных, их получения из хранилища и репликации.

19. Адресация в сети Интернет. Виды адресов

Виды адресации в компьютерных сетях

Для того, чтобы компьютеры могли идентифицировать друг друга в информационно-вычислительной сети, им присваиваются явные адреса. Основными типами адресов являются следующие:

- MAC-адрес;
- IP-адрес;
- Доменный адрес;
- URL.

Физические адреса

Локальный адрес узла определяется технологией, с помощью которой построена отдельная сеть, в которую входит данный узел.

Для узлов, входящих в локальные сети — это MAC-адрес сетевого адаптера или порта маршрутизатора, например, 11-A0-17-3D-BC-01.

Эти адреса назначаются производителями оборудования и являются уникальными адресами, так как управляются централизованно.

Для всех существующих технологий локальных сетей MAC-адрес имеет формат 6 байтов: старшие 3 байта — идентификатор фирмы производителя, а младшие 3 байта назначаются уникальным образом самим производителем.

IP-адресация

IP-адрес состоит из 4 байт, например, 109.26.17.100. Этот адрес используется на сетевом уровне. Он назначается администратором во время конфигурирования компьютеров и маршрутизаторов. IP-адрес состоит из двух частей: номера сети и номера узла.

Номер сети может быть выбран администратором произвольно, либо назначен по рекомендации специального подразделения Internet NIC (Network Information Center), если сеть должна работать как составная часть Internet. Обычно провайдеры услуг Internet получают диапазоны адресов у подразделений INIC, а затем распределяют их между своими абонентами.

Номер узла в протоколе IP назначается независимо от локального адреса узла. Деление IP-адреса на поле номера сети и номера узла — гибкое, и граница между этими полями может устанавливаться весьма произвольно. Узел может входить в несколько IP-сетей. В этом случае узел должен иметь несколько IP-адресов, по числу сетевых связей.

Таким образом IP-адрес характеризует не отдельный компьютер или маршрутизатор, а одно сетевое соединение.

Для предотвращения истощения доступного адресного пространства IPv4, была разработана новая версия 6 протокола IP, которая более способна к расширению и масштабированию.

Классы IP-адресов

IP-адрес имеет длину 4 байта и записывается в виде четырех чисел, представляющих значения каждого байта в десятичной форме и разделенных точками, например, 128.10.2.30 - традиционная десятичная форма представления адреса, а 10000000 00001010 00000010 00011110 — двоичная форма представления этого же адреса.

Адрес состоит из двух логических частей — номера сети и номера узла в сети. Какая часть адреса относится к номеру сети, а какая — к номеру узла, определяется значениями первых бит адреса. Значения этих бит являются также признаками того, к какому классу относится тот или иной IP-адрес.

Система доменных имен

IP-адресация удобна для машинной обработки таблиц маршрутов, однако сложна для использования человеком. Система доменных имен (Domain Name System — DNS) позволяет присваивать компьютерам легко запоминаемые имена, например, yahoo.com, и отвечает за перевод этих имен в IP-адреса.

Символьный идентификатор-имя, например, SERV1.IBM.COM.

Этот адрес назначается администратором и состоит из нескольких частей, например, имени машины, имени организации, имени домена.

Такой адрес, называемый также DNS-именем, используется на прикладном уровне, например, в протоколах FTP или telnet.

Универсальная идентификация ресурсов (URL) (Uniform Resource Locator — универсальный указатель ресурсов) — система обозначений для однозначной идентификации компьютера, каталога или файла в Internet.

В систему URL заложены следующие принципы:

- Расширяемость — новые адресные схемы должны легко вписываться в существующий синтаксис URL; расширяемость достигается за счет выбора определенного порядка интерпретации адресов, который базируется на понятии "адресная схема". Идентификатор схемы стоит перед остатком адреса, отделен от него двоеточием и определяет порядок интерпретации остатка;
- Полнота — по возможности любая из существовавших схем должна описываться посредством URL;
- Читаемость — адрес должен легко пониматься человеком, что вообще характерно для технологии WWW, — документы вместе с ссылками могут разрабатываться в обычном текстовом редакторе.

Формат URL включает:

- Схему адреса (тип протокола доступа — http, gopher, wais, telnet, ftp и т.п.);
- IP- или доменный адрес машины;
- Номер TCP-порта;
- Адрес ресурса на сервере (каталог или путь к файлу);
- Имя HTML-файла или метку;
- Критерий поиска данных.

Схема HTTP — это основная для Web. Она содержит идентификатор, адрес машины, TCP-порт, путь в директории сервера, поисковый критерий и метку.

<http://astra.net.ru/master/index.html> (пример)

В данном случае путь состоит из доменного адреса машины, на которой установлен сервер HTTP, и пути от корня дерева сервера к файлу index.html.

Кроме подобной полной записи URL существует упрощенная, которая предполагает, что к моменту ее использования многие основные компоненты адреса ресурса уже определены (протокол, адрес машины в сети, некоторые элементы пути). В таком случае достаточно указывать только адрес, относительный определенных базовых ресурсов — относительный адрес.

20. Модель OSI, её уровни

International Standards Organization (Международная Организация по Стандартам), ISO разработала модель, которая четко определяет различные уровни взаимодействия систем, дает им стандартные имена и указывает, какую работу должен делать каждый уровень. Эта модель называется моделью взаимодействия открытых систем (Open System Interconnection, OSI) или моделью ISO/OSI.

Сетевая модель OSI состоит из семи уровней, иерархически расположенных от большего к меньшему. Модель OSI разрабатывалась еще в 1970-х годах, чтобы описать архитектуру и принципы работы сетей передачи данных.

Модель OSI служит инструментом при диагностике сетей. Если в сети что-то не работает, то гораздо проще определить уровень, на котором произошла неполадка, чем пытаться перестроить всю сеть заново.

Зная архитектуру сети, гораздо проще ее строить и диагностировать. Как нельзя построить дом, не зная его архитектуры, так невозможно построить сеть, не зная модели OSI. При проектировании важно учитывать все. Важно учесть взаимодействие каждого уровня с другими, насколько обеспечивается безопасность, шифрование данных внутри сети, какой прирост пользователей выдержит сеть без обрушения, будет ли возможно перенести сеть на другую машину и т.д. Каждый из перечисленных критериев укладывается в функции одного из семи уровней.

Уровни модели OSI [\[править | править код \]](#)

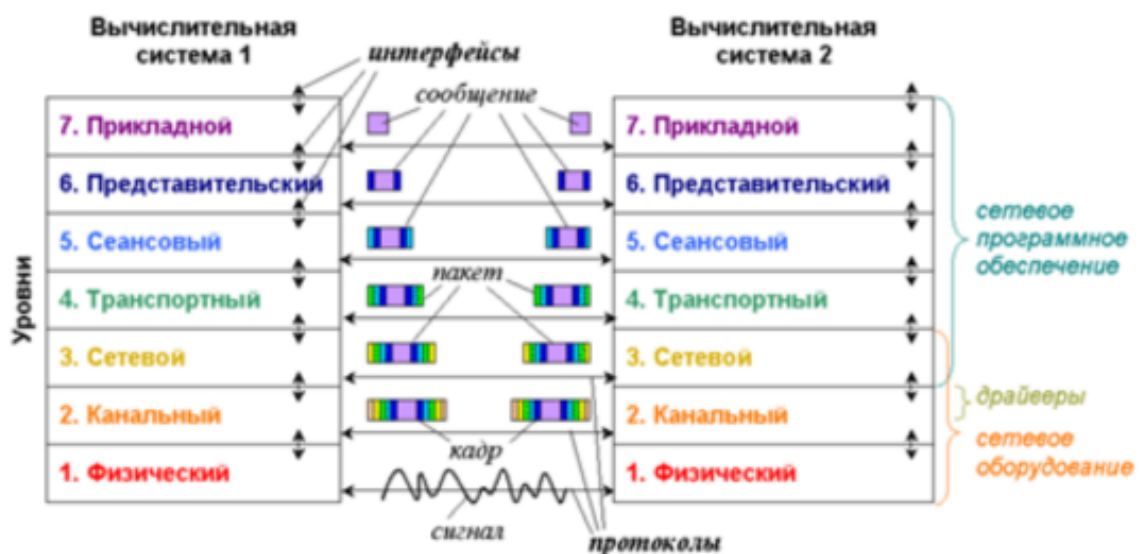
Модель					
Уровень (layer)		Тип данных (PDU ^[15])	Функции	Примеры	Оборудование
Host layers	7. Прикладной (application)	Данные	Доступ к сетевым службам	HTTP, FTP, POP3, SMTP, WebSocket	Хосты (клиенты сети), Межсетевой экран
	6. Представления (presentation)		Представление и шифрование данных	ASCII, EBCDIC, JPEG, MIDI	
	5. Сеансовый (session)		Управление сеансом связи	RPC, PAP, L2TP, gRPC	
	4. Транспортный (transport)	Сегменты (segment) / Датаграммы (datagram)	Прямая связь между конечными пунктами и надёжность	TCP, UDP, SCTP, Порты	
Media ^[16] layers	3. Сетевой (network)	Пакеты (packet)	Определение маршрута и логическая адресация	IPv4, IPv6, IPsec, AppleTalk, ICMP	Маршрутизатор, Сетевой шлюз, Межсетевой экран
	2. Канальный (data link)	Биты (bit)/ Кадры (frame)	Физическая адресация	PPP, IEEE 802.22, Ethernet, DSL, ARP, сетевая карта.	Сетевой мост, Коммутатор, точка доступа
	1. Физический (physical)	Биты (bit)	Работа со средой передачи, сигналами и двоичными данными	USB, RJ («витая пара», коаксиальный, оптоволоконный),	Концентратор, Повторитель (сетевое оборудование)

Все семь уровней модели OSI можно условно разделить на две группы:

- Media layers (уровни среды);
- Host layers (уровни хоста).

Уровни группы Media Layers (L1, L2, L3) занимаются передачей информации (по кабелю или беспроводной сети), используются сетевыми устройствами, такими как коммутаторы, маршрутизаторы и т.п. Уровни группы Host Layers (L4, L5, L6, L7) используются непосредственно на устройствах, будь то стационарные компьютеры или мобильные устройства.

Рисуночки из презы:



Первый, физический уровень (physical layer, L1)

Начнем с самого нижнего уровня. Он отвечает за обмен физическими сигналами между физическими устройствами, «железом». Компьютерное железо не понимает, что такое картинка или что на ней изображено, «железу» картинка понятна только в виде набора нулей и единиц, то есть бит.

Каждый уровень имеет свои PDU (Protocol Data Unit), представляемые в той форме, которая будет понятна на данном уровне и, возможно, на следующем до преобразования. Работа с чистыми данными происходит только на уровнях с пятого по седьмой.

Устройства физического уровня оперируют битами. Они передаются по кабелям (например, через оптоволокно) или без — например, через Bluetooth или IRDA, Wi-Fi, GSM, 4G и так далее.

Второй уровень, канальный (data link layer, L2)

Когда два пользователя находятся в одной сети, состоящей только из двух устройств, — это идеальный случай. Но что если этих устройств больше?

Второй уровень решает проблему адресации при передаче информации. Канальный уровень получает биты и превращает их в кадры (frame, также «фреймы»). Задача здесь — сформировать кадры с адресом отправителя и получателя, после чего отправить их по сети.

У канального уровня есть два подуровня — это MAC и LLC. MAC (Media Access Control, контроль доступа к среде) отвечает за присвоение физических MAC-адресов, а LLC (Logical Link Control, контроль логической связи) занимается проверкой и исправлением данных, управляет их передачей. Для упрощения мы указываем LLC на втором уровне модели, но, если быть точными, LLC нельзя отнести полностью ни к первому, ни ко второму уровню — он между.

На втором уровне OSI работают коммутаторы, их задача — передать сформированные кадры от одного устройства к другому, используя в качестве адресов только физические MAC-адреса.

На канальном уровне активно используется протокол ARP (Address Resolution Protocol — протокол определения адреса). С помощью него 64-битные MAC-адреса сопоставляются с 32-битными IP-адресами и наоборот, тем самым обеспечивается инкапсуляция и декапсуляция данных.

Третий уровень, сетевой (network layer, L3)

На третьем уровне появляется новое понятие — маршрутизация. Для этой задачи были созданы устройства третьего уровня — маршрутизаторы (их еще называют роутерами). Маршрутизаторы получают MAC-адрес от коммутаторов с предыдущего уровня и занимаются построением маршрута от одного устройства к другому с учетом всех потенциальных неполадок в сети.

Четвертый уровень, транспортный (transport layer, L4)

Четвертый уровень — это посредник между Host Layers и Media Layers, относящийся скорее к первым, чем к последним. Его главной задачей является транспортировка пакетов. Естественно, при транспортировке возможны потери, но некоторые типы данных более чувствительны к потерям, чем другие. Например, если в тексте потеряются гласные, то будет сложно понять смысл, а если из видеопотока пропадет пара кадров, то это практически никак не скажется на конечном пользователе. Поэтому при передаче данных, наиболее чувствительных к потерям на транспортном уровне, используется протокол TCP, контролирующий целостность доставленной информации.

Для мультимедийных файлов небольшие потери не так важны, гораздо критичнее будет задержка. Для передачи таких данных, наиболее чувствительных к задержкам, используется протокол UDP, позволяющий организовать связь без установки соединения.

При передаче по протоколу TCP данные делятся на сегменты. Сегмент — это часть пакета. Когда приходит пакет данных, который превышает пропускную способность сети, пакет делится на сегменты допустимого размера. Сегментация пакетов также требуется в ненадежных сетях, когда существует большая вероятность того, что большой пакет будет потерян. При передаче данных по протоколу UDP пакеты данных делятся уже на датаграммы. Датаграмма (datagram) — это тоже часть пакета, но ее нельзя путать с сегментом.

Главное отличие датаграмм — в автономности. Каждая датаграмма содержит все необходимые заголовки, чтобы дойти до конечного адресата, поэтому они не зависят от сети, могут доставляться разными маршрутами и в разном порядке. При потере датаграмм или сегментов получают «битые» куски данных, которые не получится корректно обработать.

Первые четыре уровня — специализация сетевых инженеров. С последними тремя они не так часто сталкиваются, потому что пятым, шестым и седьмым занимаются разработчики.

Пятый уровень, сеансовый (session layer, L5)

Пятый уровень оперирует чистыми данными. Помимо пятого, чистые данные используются также на шестом и седьмом уровне. Сеансовый уровень отвечает за поддержку сеанса или сессии связи. Пятый уровень оказывает услугу следующему: управляет взаимодействием между приложениями, открывает возможности синхронизации задач, завершения сеанса, обмена информации.

Службы сеансового уровня зачастую применяются в средах приложений, требующих удаленного вызова процедур, т.е. чтобы запрашивать выполнение действий на удаленных компьютерах или независимых системах на одном устройстве (при наличии нескольких ОС).

Примером работы пятого уровня может служить видеозвонок по сети. Во время видеосвязи необходимо, чтобы два потока данных (аудио и видео) шли синхронно. Когда к разговору двоих человек прибавится третий — получится уже конференция. Задача пятого уровня — сделать так, чтобы собеседники могли понять, кто сейчас говорит.

Шестой уровень, представления данных (presentation layer, L6)

О задачах уровня представления вновь говорит его название. Шестой уровень отвечает за преобразование протоколов и кодирование/декодирование данных. Шестой уровень также занимается представлением картинок (в JPEG, GIF и т.д.), а также видео-аудио (в MPEG, QuickTime). А помимо этого → шифрованием данных, когда при передаче их необходимо защитить.

Седьмой уровень, прикладной (application layer)

Седьмой уровень иногда еще называют уровень приложений, но чтобы не запутаться можно использовать оригинальное название — application layer. Прикладной уровень — это то, с чем взаимодействуют пользователи, своего рода графический интерфейс всей модели OSI, с другими он взаимодействует по минимуму.

Все услуги, получаемые седьмым уровнем от других, используются для доставки данных до пользователя. Протоколам седьмого уровня не требуется обеспечивать маршрутизацию или гарантировать доставку данных, когда об этом уже позаботились предыдущие шесть. Задача седьмого уровня — использовать свои протоколы, чтобы пользователь увидел данные в понятном ему виде.