# Билет №1.

## **Теория.** Типы стратегий в модели М. Портера.

Стратегия – направленность деятельности и приемы выживания и развития организации, отличающие ее от других организаций. Кроме того, стратегия определяет масштабы деятельности и способы взаимодействия с группами влияния: работниками, потребителями, собственниками, конкурентами.

Этапы разработки стратегии Логика стратегического процесса подразумевает его деление на две генеральные стадии: • Формулирование стратегии; • Осуществление стратегии. Каждая из этих стадий может делиться на разное количество этапов в соответствии с принятыми способами выработки стратегий, а также взглядами и познаниями руководителей высшего звена. Многие «нематериальные» факторы должны приниматься во внимание при попытке следовать внутренней логике компании в осуществлении стратегического процесса.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, диаграмма, Шрифт

Автоматически созданное описание

Пять сил и три о́бщие страте́гии По́ртера, концептуальный подход к обоснованию рыночных стратегий компаний, построенный на разработанных М. Портером в 1979 г. принципах анализа «пяти сил» и дополненный в 1980 г. описанием трёх базовых вариантов стратегий, основанных на анализе этих «пяти сил».

Пять сил Портера — это анализ пяти факторов, которые влияют на прибыль компании. Майкл Портер разработал эту модель ещё в 1979 году, но бизнес активно использует её до сих пор.

5 сил Портера, влияющих на развития бизнеса: Покупатели (клиенты); Поставщики (если их нет, просто не учиваются); Действующие конкуренты; Новые конкуренты; Товары-заменители. Создатель теории утверждает, что каждый из этих факторов оказывает определенное давление на бизнес. В некоторой степени они считаются внешними (поскольку компания никак не может на них повлиять).

Майкл Портер говорит, что существуют всего три базовые конкурентные стратегии: Лидерство по издержкам; Дифференцирование (Лидерство по продукту); Фокусирование.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

1.Стратегия дифференцирования – создание товаров (услуг), отличающихся от аналогов и имеющих повышенную ценность для покупателя.

Источники отличий: навыки маркетинга (рекламы), ключевые компетенции (навыки, технологии), придающие товарам особые черты, за которые потребители готовы платить ценовую премию.

Чаще эта стратегия нацелена на потребителей, не слишком озабоченных ценой.

Пример: производители джипов, немецких автомобилей, французской косметики, швейцарских часов, INTEL.

2. Стратегия доминирования в издержках – создание превосходства компании в снижении издержек, снижении цены на товар.

Стратегия ориентирована на стабильный рынок, чем на риск и поиск возможностей. Компания обладает компетенцией сбивать цены конкурентам, обеспечивая тот же уровень качества и умеренную прибыль. Эта стратегия позволяет создать барьеры для входа на рынок новым «игрокам»

Пример: китайские компании – производители автомобилей, Dell Computer.

3. Стратегия фокусирования (концентрации) – лучшее обслуживание конкретного рыночного сегмента – группе покупателей (либо 1-я, либо 2-я стратегия, сфокусированная на нужды группы потребителей).

Пример: пивоварня, расположенная в регионе.

**Стратегия «Лидерство по издержкам»** Довольно понятная стратегия, когда компания решает сокращать все возможные расходы. Единственное, обратите внимание, что стратегия не подразумевает при этом обязательного лидерства по цене. Т.е. вы можете сокращать издержки и при этом не предлагать самые низкие цены.Также необходимо понимать, что выбрав подобную стратегию и сокращая цены, вам необходимо сконцентрироваться на достаточно большом рынке, где вы сможете заработать необходимые вам доходы.Если же вы не сокращаете цены, но сокращаете издержки, вы можете выбрать смежную стратегию — фокусирования.

**Минусы стратегии.** Данная стратегия очень опасна, так как велика вероятность того, что рано или поздно появятся конкуренты, способные сделать свои издержки еще ниже. Все это возможно, как за счет более качественного маркетинга, так и за счет таких факторов как: сеть дистрибуции, технологический прогресс, ноу-хау в управлении, внешние факторы в стране и мире, приход на рынок более крупных глобальных игроков, потеря мотивации сотрудниками и так далее.

Одним из главных соблазнов для лидера по издержкам является расширение товарного ассортимента. Но прибегать к нему стоит, 10 раз подумав, так как такое расширение может уничтожить все преимущество по издержкам, тем самым погубив компанию. Еще один фактор, который не стоит терять из виду – потребители. Они могут стать тем фактором, который может заставить компанию снизить цены, что поведет за собой уничтожение всего преимущества лидера по издержкам.

Пример стратегии лидерства по издержкам В качестве примера компании, строго придерживающейся этой стратегии, приведем интернет-магазин Плеер.ру, где каждое дополнительное действие оплачивается отдельно (при изначально низких ценах).

**Стратегия «Дифференцирование, или лидерство по продукту»**

Для выбора этой стратегии у вас должно быть некое отличительное свойство, то, что позволяет вашему продукту быть лидером на рынке. То, что делает вас особенным, дифференцированным.

Что это может быть?

* Качество;
* Обслуживание;
* Логистика;
* Рекламные усилия;
* Первооткрывательство (просто вы вышли первыми);
* Диллерская сеть (особая, неповторимая);
* Технологии продукта (или производства продукта);
* Технологии продаж;
* Бренд .

Если у вас есть то, что позволяет вам дифференцироваться, старайтесь выходить на широкие рынки. Согласно канонической теории М. Портера, конкурентное преимущество на рынке возникает на основе предоставления потребителям продукции, дающей большую ценность за ту же стоимость (дифференциация), или предоставления равной ценности, но за меньшую стоимость (низкие Издержки).

**Продуктовая дифференциация** — когда характеристики и/или дизайн предлагаемого продукта лучше, чем у конкурентов. Этот вид дифференциации сложен в применении, если речь идет о каких-либо стандартизованных продуктах (продукты питания первой необходимости, нефтепродукты, металл). Зато при продвижении дифференцированных продуктов (косметика, одежда) следование данной стратегии является обычным явлением.

**Сервисная дифференциация** - предложение дополнительных услуг, которые сопутствуют предлагаемому продукту, в которых покупатель так или иначе нуждается перед покупкой или после неё. Это может быть обучение и консультирование, скорость и надёжность поставок, установка, сервисное обслуживание. Для успешной сервисной дифференциации сопутствующие услуги должны быть либо бесплатными, либо более дешёвыми, либо превосходить по своему уровню услуги конкурентов.

**Дифференциация персонала** — когда ставка делается на персонал, который выполняет свои функции более эффективно, чем персонал конкурентов. Обычно, дифференциация персонала чаще всего используется в сфере предоставления услуг.

Естественно, персонал, который вызывает доверие, создаёт впечатление надежных, ответственных и коммуникабельных людей и является компетентным в сфере своих обязанностей получить непросто. Надо достаточно времени, средств и усилий вложить в его обучение.

**Дифференциация имиджа** заключается в создании определённого образа организации или её продукции, который отличает их в лучшую сторону от конкурентов. Также эта стратегия известна как брендирование, и достигается исключительно за счёт эффективной рекламы.

В зависимости от особенностей конкретных продуктов и возможностей компании можно реализовать одновременно от одного до нескольких направлений дифференциации.

Пример стратегии дифференцирования

Стратегия дифференцирования самый редкий гость на российском рынке, т.к. любимый инструмент отечественного маркетинга — это скидка. В качестве примера приведем интернет-магазин Юлмарт.

Отличительной характеристикой Юлмарта было использование ЦИЗов — огромных помещений размером с Ашан, но при этом без затрат на мерчендайзинг и обслуживание полок, т.к. клиентов держали в предбаннике у десятка терминалов. При этом, в отличие от того же Ашана, ЦИЗ можно было открывать в местах без проходимости, трафик обеспечивала интернет-активность магазина. Более 60% заказов этот интернет-магазин получал непосредственно с терминалов, установленных в ЦИЗах. ЦИЗ – Центр исполнения заказов.

**Стратегия «Фокусирование»**

Отличается от дифференцирования тем, что в фокусировании вы нацеливаетесь на определенный сегмент рынка, а не на отличительное свойство продукта/компании.

Как можно фокусироваться?



**Виды фокусирования**

1.**Стратегия фокусирования на издержках**. Небольшие издержки помогут добиться конкурентного преимущества в данном узком секторе рынка.

2.**Стратегия фокусирования на дифференциации.** Продукт компании должен выглядеть привлекательным, привлекательнее, чем у конкурентов , для выбранной аудитории.

**Принцип фокусирования**

В фокусе могут быть различные вещи: определенный тип потребителя (возраст, социальный статус), географическое положение (страна, регион, город), видовая сфера товара (технические новинки) и т.д. [1].

Примеры: Журнал "Футбол", сеть магазинов для беременных и мам "Кенгуру", московская сеть киосков "Чистка обуви".

Почему не возможны срединные стратегии?

Если у вас есть дифференцирование, оно вам дорого обходится. Например, у вас отличная логистика, вы доставляете быстро, но это удовольствие не из дешевых. Если при этом вы назначаете низкие цены на свой продукт (т.е. взяли еще стратегию лидерства по издержкам), то вам нужен широкий охват, чтобы заработать небольшую прибыль, но с широкой аудитории получить большой валовый доход. Если же вы выбрали узкую аудиторию, при низких ценах и затратах на дифференцирование вам будет крайне сложно выжить.

Просто вставка

«Анализ пяти сил Портера» включает в себя три силы «горизонтальной» конкуренции: угроза появления продуктов-заменителей, угроза появления новых игроков, уровень конкурентной борьбы; и две силы «вертикальной» конкуренции: рыночная власть поставщиков и рыночная власть потребителей.

Анализ 5 сил Портера нужен для разработки долгосрочной стратегии развития бизнеса. Чем меньше давление сил Портера, тем больше у компании шансов получить высокую прибыль. И наоборот, чем больше давление, тем ниже рентабельность бизнеса.

# Билет №2.

Основные функции управления. Связующие процессы: принятие решений, коммуникации.

**Основные функции управления**.

Планирование, организация, руководство и контроль.

Изображение выглядит как текст, снимок экрана, Шрифт, диаграмма

Автоматически созданное описание

Планирование - Анализ ситуации, определение целей организации и способов их достижения.

Суть – определение целей организации и того, что надо сделать, чтобы достичь эти цели.

Поиск ответов на три вопроса: Где мы находимся в настоящее время ?(чего может добиться организация на основе оценки конкурентов, клиентов, законов, политической, демографической ситуации, технологий и др.); Куда мы хотим двигаться? (определение целей); Как мы собираемся сделать это? (определение способов и механизмов, оценка необходимых ресурсов)

Организация - Создание структуры, способов выполнения работы, подбор людей, делегирование полномочий.

Функция организации 1. Организовать – значит создать некую материальную структуру. Материальная структура предполагает определенные способы выполнения работы, порядок распределения ресурсов между частями организации (конвейер – Г. Форд). 2. Организовать – значит создать некую социальную структуру: осуществить подбор людей для конкретной работы; назначить им задания и делегировать полномочия (права) по использованию ресурсов. Делегирование – основной способ осуществления работы с помощью других людей.

Организация как группа людей Организация – это группа людей, деятельность которых сознательно координируется для достижения общей цели (целей). Цели всякой организации включают преобразование ресурсов для достижения результатов Черты организации: ресурсы (человеческие ресурсы, капитал, материалы, технология, информация, предпринимат. способности); зависимость от внешней среды; горизонтальное разделение труда (ГРТ); подразделения; вертикальное разделение труда (ВРТ, координация).

Горизонтальное разделение труда (ГРТ) ГРТ (акцент на функциях и задачах)– это разделение всей работы по производству товара (услуги) на отдельные составляющие и распределения заданий в соответствии с различными обязанностями. Горизонтальное разделение труда (ГРТ) предполагает подчинение функциональному принципу, который помогает сотруднику организации осознать свои функции (должностные обязанности) и понять, каким образом его работа соотносится с функциями, выполняемыми другими работниками, и конечной целью организации. Ярким пример организации с реализованным функциональным принципом и ГРТ - компания МакДоналдс. Менеджмент данной компании апробирует специфический способ действий (стратегическую перспективу), который предполагает разбиение всей работы по приготовлению пищи на узкие операции. В результате существенно повышается производительность и рестораны МакДоналдс обслуживают в течение дня в сотни раз больше людей, чем традиционные маленькие ресторанчики.

Вертикальное разделение труда (ВРТ) формируется под действием скалярного принципа, в соответствии с которым управленческая власть и ответственность распределяются по организации не с учетом содержания выполняемой работы, а на основе степени власти, которой обладают различные менеджеры для достижения цели (целей) организации. Следствием ГРТ является разделение всей работы в организации на составляющие части. Однако после того как вся работа по производству товара (услуги) разделена на отдельные компоненты, кто-то должен координировать работу групп для того, чтобы она была успешной и позволяла достигать цели организации. Деятельность по координированию работы других людей и составляет сущность управления.

Изображение выглядит как текст, линия, диаграмма, Шрифт

Автоматически созданное описание

Координация – упорядоченное согласование руководителем групповых усилий в целях обеспечения единства действий организации, исполнения ее миссии, приспособления к изменениям в окружающей среде. Норма управляемости – численность подчиненных, которых может эффективно контролировать один человек (не более 5-6 чел.)

Контроль - Наблюдение за действиями, измерение результатов, коррекция «разрывов» между фактическими и запланированными результатами

«Внешнее» понимание контроля не затрагивает деятельность непосредственно самого руководителя по выявлению проблем, а также работников – по самостоятельному исправлению допущенных ими отклонений без обращения к менеджерам. Контроль -«наблюдение за тем, чтобы все происходило в соответствии с установленными правилами и точными распоряжениями». Инструмент - штрафные санкции .

«Внутреннее» понимание контроля фокусируется на: 1) установление руководителем «стандартов», т.е. поддающихся измерению целей, которые должны быть достигнуты в обозначенный отрезок времени в соответствии с принятым планом; 2) измерение того, что было в действительности достигнуто за определенный период и сравнение достигнутого с ожидаемыми результатами. На данном этапе менеджер пытается понять, в чем заключаются причины возникших проблем (это и есть основа управленческой деятельности). 3) Получение руководителем обратной связи и практическая реализация идей, направленных на совершенствование поведения организации.

**Обратная связь** – это информация о ходе выполнения задания. Обратная связь отражает то, что человек (педагог, ученик) испытывает по поводу сказанного или сделанного другими. Негативная информация от учителя стимулирует учеников сконцентрировать энергию на исправлении недостатков в учебе. Отсутствие обратной связи может лишь заморозить неудовлетворительные результаты и привести к ухудшению отношений в школе (классе) в соответствии с теорией справедливости.

Изображение выглядит как снимок экрана, линия, текст, Шрифт

Автоматически созданное описание

360-градусная обратная связь В этом случае обратную связь по поводу выполнения, например, работы педагогом дает не только администрация, но и коллеги, сотрудники школы, родители (потребители образовательных услуг), ученики и даже возможно члены семьи педагога. Этот метод оценки персонала приобретает сейчас все большую популярность. Однако одновременно предъявляет дополнительные требования. Например, все стороны, участвующие в обратной связи, должны соблюдать осторожность и обладать навыками межличностного общения. Обратная связь должна содержать не только критику, но также и признание заслуг.

Стратегии контроля в организации Уильям Оучи (University of California):

Существует три стратегии контроля – бюрократическая, рыночная и клановая. Они могут существовать в организации одновременно.

Основные характеристики стратегий контроля:

Бюрократическая: Правила, стандарты, иерархия, легитимность власти.

Рыночная: Цены, конкуренция, взаимосвязи.

Клановая: Традиции, общие ценности, убеждения, доверие.

Бюрократический контроль Главная цель – стандартизация и контроль поведения сотрудников. Нормы и правила поведения формируются методом проб и ошибок в процессе контроля за персоналом и анализа ошибок. Правила содержат информацию о «правильных» формах поведения в организации. Менеджеры должны располагать властью для контроля. Легитимная власть предотвращает другие формы контроля, например, протекционизм и взяточничество.

Рыночный контроль Критерий (форма) контроля – цена (прибыли, получаемые компанией). Прибыли и издержки сводятся в единые отчетные документы и по ним оценивается работа организации в целом и отдельных ее подразделений (сравнивается). В отсутствии конкуренции этот метод неточно отражает внутреннюю эффективность. К рыночному контролю обращаются также и правительственные организации, отдавая подряды на работы на конкурсной основе. В горизонтальных (модульных) структурах оценивается прибыльность автономных (хозрасчетных) подразделений. Они могут конкурировать между собой, предоставляя услуги, необходимые головной организации. Например, исследовательский отдел м.б. преобразован в полуавтономный коммерческий центр (пример с модульной структурой, содержащей опытное производство и исследовательско-предпринимательские ячейки).

Клановый контроль базируется на коллективном мнении о ценностях и правилах поведения, используется чаще в малых организациях (образовательные учреждения) или организациях со сложившейся культурой, требует приверженности людей ценностям, традициям, общему делу и личностной вовлеченности работников. В условиях неопределенности ОС и быстрых перемен традиционный бюрократический контроль, опирающийся на жесткие правила, оказывается не эффективным. Поэтому обучающиеся организации используют даже не клановый контроль (связан с необходимостью «врастания» в группу), а самоконтроль , который базируется на ценностях и целях самого работника. Важно приблизить ценности компании и работников, иметь сильных лидеров, определяющих границы свободы действий работников. Клановый контроль является даже более жестким по сравнению с бюрократической иерархией.

Руководство. - Создание условий для гармонического взаимодействия и мотивации работников.

Функция (процесс) «Руководство» Руководство имеет особое значение для менеджмента, т.к. практически единственным ресурсом, находящимся в распоряжении руководителя организации являются человеческие ресурсы. Руководство- поддержка трудовой активности персонала путем использования влияния менеджера и специфических приемов, повышающих уровень мотивации работников Проблемы руководства: • Организационное поведение; • Организационная культура; • Мотивация; • Группы и Команды; • Управление конфликтами; • Власть и лидерство.

Тема 6. Организационное поведение (ОП). Индивидуальные различия, влияющие на поведение в организации. Организационная культура 1) ОП – междисциплинарная область знания, связанная с изучением человеческих установок, поведения и трудовой деятельности в организации. 2) ОП - наука о том, как люди ведут себя в организации и каким образом их поведение влияет на организацию, например на текучесть кадров, дисциплину, удовлетворенность выполняемой работой, уровень адаптации в социальном окружении. Цели ОП – 1) достижение понимания о причинах поведения людей в организации, 2) предсказание человеческого поведения и 3) управление им.

Организационная культура – это набор ожиданий, ценностей, норм, запретов, симпатий и т.п., которые осознанно или неосознанно разделяются большинством членов организации. Например, в гимназиях работать легче, т.к. дети однородны по интеллекту и социальному составу (ниже требования к пед. квалификации). Но появляются проблемы с внеурочной работой. Ценность – это то, что ценится человеком высоко и от чего он не готов отказаться ни при каких условиях. Ценности позволяют человеку отличать правильное от неправильного, хорошее от плохого. В структуру ценности входит не только содержание, но и место (работа, магазин), и способ ее реализации (уважение надо заслужить, а любовь можно получить в подарок).

Коммуникационный стиль – способ, с помощью которого индивид предпочитает строить взаимодействие с другими.

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Обратная связь. Получаемое вознаграждение определяет восприятие человеком своего поведения как приемлемого и целесообразность его повторения в будущем.

Обобщенный взгляд на мотивацию как на результат потребностей, поведения и вознаграждения:

Изображение выглядит как текст, круг, снимок экрана, диаграмма

Автоматически созданное описание

А. Маслоу: базовые принципы, характеризующие природу человека.

1. Люди являются «животными», обремененными потребностями, которые никогда не могут быть удовлетворены. 2. Состояние частичного или полного неудовлетворения потребностей побуждает человека к действию (неудовлетворенная потребность приводит к действию). 3. Существует иерархия потребностей.

Изображение выглядит как текст, снимок экрана, мультфильм, диаграмма

Автоматически созданное описание

Проявление потребностей на работе Физиологические потребности: люди, которые работают для удовлетворения этих потребностей, мало интересуются содержанием работы и фокусируются на условиях оплаты, удобстве рабочего места, близости работы к дому и т.п. Потребность безопасности – люди стремятся избегать волнительных ситуаций, любят порядок, четкие правила, оценивают свою работу, прежде всего, с точки зрения обеспечения им стабильного существования в будущем. Потребности принадлежности и причастности – человек смотрит на свою работу, как на принадлежность к коллективу и как на возможность установить хорошие отношения с коллегами. Признания и самоутверждения – желание быть сильным, уверенным в себе, люди стремятся к лидерскому положению либо к положению признанного авторитета при решении проблем. Потребность самовыражения – стремление стать тем, кем человек способен стать, люди стремятся к оригинальным и сложным заданиям, хотят большей свободы и самостоятельности в выборе средств для решения задач.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

**Связующие процессы** (выполняются в каждой из функций управления: планирование, организация, руководство и контроль)

Принятие решений – это переход от анализа ситуации к практическому действию, выбор менеджером того, что (кого), зачем, когда и как планировать, организовывать, мотивировать, координировать и контролировать (решение – выбор альтернативы, т. е. варианта действий).

Обмен информацией (коммуникации) – это процесс обмена информацией (смыслами) между субъектами как внутри организации, так и между организацией и окружающей средой. Коммуникации считаются эффективными, если они осуществляются без существенной потери смысла.

Решение – это выбор альтернативы.

1. Решение, основанное на суждении – это выбор, обусловленный знаниями и (или) накопленным опытом. Решение принимается быстро, но срабатывают для простых и повторяющихся ситуаций, т.е. для условий определенности. В новых сложных ситуациях менеджер может упустить эффективную альтернативу

2. Интуитивное решение – это выбор, сделанный только на ощущении того, что он правилен. Руководитель взвешивает «за» и «против» и не нуждается в понимании ситуации. Эти решения принимаются в ситуациях с риском и неопределенностью. Риск – вероятность неблагоприятного исхода, неопределенность – ситуация, сопряженная с отсутствием знания об исходах (последствиях) от принимаемого решения.

3. Рациональное решение – это серия стадий (этапов), через которые должен пройти руководитель, чтобы дойти до выполнения решения. Ограничения реального мира, препятствующие рациональной модели: 1) проблема часто скрыта и ЛПР не знает, что она существует; 2) ограничения во времени, отпущенного для принятия решения; 3) рассматриваются не все альтернативы и при их оценке учитываются не все критерии.

Фундаментальный принцип рационализма (принцип эффективности): из нескольких альтернатив, имеющих одинаковые затраты, следует выбрать ту, которая приводит к наилучшему результату, а из нескольких альтернатив с одинаковым результатом следует выбирать ту, которая обещает наименьшие затраты. *Хотя максимизация результата и минимизация затрат являются целями управленческой деятельности, но всегда ли (для всех ли ситуаций) выполним ли этот принцип?!*

Стадии рационального решения:

1. Признание необходимости решения: - восприятие и признание проблемы; - интерпретация и формулирование проблемы; - определение критериев для выбора альтернатив

2. Выработка решения: – разработка альтернатив; -- оценка альтернатив; -- выбор альтернативы.

3. Выполнение решения: -- организация выполнения решения; -- анализ и контроль; -- обратная связь и корректировка.

Коммуникации – передача информации от одного субъекта к другому. Коммуникации включаю в себя и то, что передается (информацию), и то, как это «что» передается.

1. Коммуникации между организацией и средой (с потребителями, с государством – отчеты, с вышестоящими органами управления и т.д.).

2. Межуровневые коммуникации в организации: а) вертикальные (по нисходящей, по восходящей); б) горизонтальные (между равными по уровню субъектами); диагональные (общение с другими начальниками и другими подчиненными).

3. Неформальные коммуникации – слухи (80-95% слухов верны в отношении эмоционально не окрашенной информации)

Коммуникационная сеть – это соединение определенным образом участвующих в коммуникационном процессе индивидов с помощью коммуникационных потоков. Устоявшиеся образцы коммуникационных сетей (5 субъектов)

**Изображение выглядит как диаграмма, линия, оригами, дизайн

Автоматически созданное описание**

Коммуникационный процесс – обмен информацией между 2-мя или более субъектами. Пять элементов: отправитель, сообщение, канал, получатель, обратная связь

Этап посылки: отправитель – формулирование значения – носители послания (через что решили послать) – форма послания (как организовал значение) – послание (что было послано) – передатчик (кто передал информацию) – канал – -- ШУМ –

Этап получения: - канал – приемник (кто получил) – восприятие послания (что получил) – раскодирование, интерпретация послания (как понял) – оценка послания – принятие значения (принял ли послание) – получатель (кто должен ответить на послание) –

Обратная связь – получатель и отправитель меняются местами

Коммуникационный стиль – способ, с помощью которого индивид предпочитает строить взаимодействие с другими.

**Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание**

Невербальное послание – информация, посланная отправителем без использования слов. Невербальные коммуникации имеют бессознательную основу и свидетельствуют об истинных переживаемых эмоциях.

55% смысла воспринимается через выражение лица, позы, жесты.

38% смысла воспринимается через интонацию и модуляцию голоса.

7% - остается за словом

То, как мы говорим важнее слов, которые мы произносим.

Руководителю (педагогу) важно обладать эмпатией, т. е. уметь разбираться в мыслях и эмоциях, переживаемых подчиненными (учениками

***Функция управления (Функция  менеджмента)*** — это виды деятельности, с помощью которых управляющая подсистема воздействует на объект управления

**Функции управления по А.Файолю:**

1. планирование;
2. организация  - обеспечение предприятия всем необходимым для работы. материалами, капиталом, оборудованием, людьми.
3. распорядительство - извлечение наибольшей пользы и выгоды из подчиненных руководителю работников в интересах предприятия в целом.
4. координирование - достижение соответствия и согласованности между различными частями предприятия
5. контроль - проверка исполнения в соответствии с принятой программой

**Другой подход к классификации функций. Функции:**

* Общие (основные) функции - реализуются в управлении любой организацией.
  + планирование,
  + организация,
  + мотивация,
  + контроль.
* Связующие функции (процессов):
  + принятие управленческих решений;
  + коммуникации.
* Частные (обеспечивающие) функции. Выполняются отдельными структурными подразделениями (специалистами):
  + управление персоналом;
  + управление маркетингом;
  + управление финансами;
  + документационное обеспечение управления;
  + учет (бухгалтерский, управленческий);
  + обеспечение оргтехникой и др.
* Специфические функции. Отражают отраслевую специфику.

**Связующие процессы: принятие решений, коммуникации.**

**Связующие процессы**, такие как принятие решений и коммуникации, помогают объединить различные функции управления и обеспечить их взаимодействие. Они позволяют достичь согласованности действий всех участников организации, создать атмосферу доверия и сотрудничества, что способствует эффективной работе и успешному достижению поставленных целей.

**Принятие решений**. Для успешной работы организации руководитель должен постоянно делать правильный выбор из нескольких альтернативных возможностей. Выбор одной из альтернатив — это и есть решение. Принятие решения — выбор того, как и что планировать, организовывать, мотивировать и контролировать. Основными требованиями для принятия эффективного решения являются наличие адекватной и точной информации.

**Коммуникация** — это процесс обмена информацией, ее смысловым значением между двумя или более людьми. Информация в процессе коммуникации передается для того, чтобы руководители могли принимать необходимые решения и для того, чтобы они могли выполняться. Планы нельзя выполнить, если они неизвестны людям, которые должны их выполнять. До тех пор пока работники не будут понимать обоснованность планов, не будут знать, какое вознаграждение им предложат за хорошо выполненную работу, они не будут хорошо работать.

# Билет №3.

Программные средства для моделирования, анализа и проектирования архитектуры предприятия.

Архитектура – фундаментальная организация системы, воплощенная в ее компонентах, их взаимосвязях друг с другом и со средой, а также руководящие принципы проектирования и развития системы (ISO/IEC 42010:2007).

Архитектура предприятия (АП): (1) Фундаментальная организация предприятия либо как целого, либо вместе с партнерами, поставщиками и (или) покупателями («расширенное предприятие»), либо части (например, бизнес-направление, департамент), а также ((2) руководящие принципы его проектитрования и развития.

бизнес-архитектурой

Этот класс программного обеспечения интегрирует технологические наработки, накопленные в системах автоматизированного проектирования (САПР), или CAD (Computer-Aided Design), реинжиниринга бизнес-процессов и др. Концепция САПР, или CAD-систем, возникла в начале 60-х гг. ХХ в.

В настоящее время CAD-системы стали основным инструментом проектировщиков в архитектуре и строительстве, машиностроении и т.п. В 80-х гг. ХХ в. эти идеи привели к возникновению средств поддержки проектирования информационных систем (Computer-Aided Software/System Engineering, CASE) [Case, 1985; Калянов, 1996]. CASE-средства можно считать родоначальником EAM-инструментов, учитывая, что исторически АП стартовала с ИТ-архитектуры. CASE обеспечивают проектирование функций и структуры приложений, структуры данных и технической инфраструктуры. Популярность реинжиниринга бизнес-процессов в 90-е гг. ХХ в. привела к появлению программных средств для моделирования, анализа и реинжиниринга процессов [Bradley et al., 1995; Spurr et al., 1994; Ойхман, Попов, 1997]. В России на волне популярности книги «7 нот менеджмента» [Бочкарев и др., 1998] и заложенных в нее идей по реструктуризации [Кондратьев, Краснова, 2000] появился уникальный класс программных средств для анализа и проектирования структуры и функций предприятий [Григорьев, 2010]. В последнее время стали возникать инструменты для проектирования стратегии (Strategy design) [Osterwalder, Pigneur, 2013; Fritscher, 2014]. Ключевые возможности и идеи, заложенные в вышеуказанных программных средствах, интегрируются в инструментах управления АП (рис.4), одновременно расширяя область охвата последних

Для моделирования, анализа и проектирования архитектуры предприятия (АП) существуют специальные программные средства — инструменты управления АП (Enterprise Architecture Management tools, EAM-инструменты).

EAM-инструменты предоставляют ряд возможностей:

* Описание в единой модели основных составляющих АП (процессов, структур, целей, информации и др.) и их взаимосвязей. Возможность работать с единой базой данных (репозиторием) и хранить информацию о деятельности предприятия «в одном месте»;
* Наглядное и удобное для восприятия представление знаний о компании разным заинтересованным лицам (руководству, сотрудникам компании, аналитикам, специалистам по ИТ, партнерам, контролирующим и сертифицирующим органам) за счет автоматической генерации отчетов из единой модели в текстовом, табличном или графическом виде;
* сбор данных об объектах архитектуры предприятия и получение обратной связи от заинтересованных сторон;
* управление проектами, реализация которых оказывает влияние на архитектуру предприятия (например, ОТ-проект по разработке и внедрению новой ИС; проекты в рамках цифровой трансформации предприятия);
* анализ моделей (количественный и качественный, статический и динамический);
* поддержание единой системы терминов, понятий и их отношений, которая повышает эффективность внутрифирменных коммуникаций.

В соответствии с аналитическим обзором Gartner лидерами в категории инструментов управления АП являются в том числе MEGA, ARIS (Software AG), Enterprise Studio (BiZZdesign).

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Для задач менеджмента особо актуальны инструменты, позволяющие эффективно работать (моделировать, анализировать и проектировать) с бизнес-архитектурой как компонентом АП. Для этого могут применяться универсальные графические редакторы, например MS Visio

или SmartDraw. И на определенном уровне их вполне хватает. Однако по мере роста масштаба и комплексности задач начинает возникать потребность в специализированных инструментах

Изображение выглядит как текст, снимок экрана, диаграмма, Шрифт

Автоматически созданное описание

Область применения профессиональных EAM-инструментов можно разделить на два критерия:

1) степень комплексности решения — от независимого (в рамках одного функционального аспекта) до взаимосвязанного комплексного решения задач;

2) масштаб решения — от локального (для отдельного объекта — подразделения, процесса) до решения в масштабе всего предприятия.

Решение, попадающее в область 1, может проводиться с использованием простых средств — графических редакторов типа Visio. Роль EAM-инструментов в таких случаях невелика.

Эффект от использования более развитых инструментов бизнес-моделирования увеличивается для решений в областях 2 и 3 и максимально проявляется в случае применения EAM-инструментов для решений из области 4.

Примерами таких решений, которые реализуются в масштабах всего предприятия, могут быть связки задач типа:

— «наведение порядка» + «оптимизация процессов» + подготовка к автоматизации;

— «прояснение ситуации» + «внедрение регулярного менеджмента» + + «оптимизация процессов».

Изображение выглядит как текст, снимок экрана, диаграмма, линия

Автоматически созданное описание

Инструменты управления АП (ЕАМ-инструменты) позволяют получать, хранить, структурировать, анализировать и наглядно представлять информацию, относящуюся к АП. Они призваны помочь успешно реализовать стратегию предприятия максимально эффективно и с наи-

меньшими рисками.

8 основных функциональных областей ЕАМ-инструментов:

1. Комплексное управление архитектурным процессом.

- поддержка настраиваемых форм для обеспечения взаимодействия заинтересованных сторон с артефактами архитектуры предприятия;

- поддержка функции по предложению пользовательских изменений в архитектуру предприятия;

- моделирование сценариев «что, если» при предложении изменений пользователями.

2. Репозиторий/метамодель. Чаще всего репозиторий представляет собой реляционную базу данных и метамодель, которые дают возможность архитекторам хранить и использовать объекты архитектуры. Типовые возможности ЕАМ-инструментов: возможность изменения/дополнения метамодели; возможность использования нескольких методологий без дублирования метамодели; поддержка федеративности архитектуры (возможность выделения отдельных географических регионов или организационных единиц).

3. Разработка, построение, структурирование, анализ и демонстрация моделей. EAM-инструменты должны поддерживать возможность создания диаграмм (а зачастую и иных артефактов) для различных заинтересованных сторон.

Типовые возможности ЕАМ-инструментов:

моделирование текущего и целевого состояния АП; интеграция с другими средствами моделирования; сквозное моделирование от бизнес-стратегии до ИТ-инфраструктуры; автоматическое структурирование моделей; анализ моделей; совместная работа с моделями.

4. Поддержка организационных изменений, цифровой трансформации, принятия решений. Предприятия все время вынуждены выбирать среди различных альтернатив с учетом рисков и других факторов. Типовые возможности ЕАМ-инструментов: анализ разрывов между текущим и целевым состоянием АП; анализ «что, если» для прогнозирования изменений; анализ инвестиционных проектов, проектное и портфельное управление, а также стратегическое планирование; планирование сценариев и применение системного подхода; возможность получения обратной связи от заинтересованных сторон с последующей визуализацией предложений в моделях архитектуры предприятия, в том числе за счет интеграции с инструментами управления проектами.

5. Поддержка инноваций. Современные ЕАМ-инструменты поддерживают возможность сбора идей и формирования банка идей. Типовые возможности ЕАМ-инструментов:

форма сбора идей; банк идей, поддерживающий возможность обсуждения, анализа, классификации, приоритезации идей.

6.Презентация и варианты представления. Максимальная простая презентация и понятна всем. Типовые возможности: визуализации различной информации, интеграция с системами разработки презентация.

7.Управление развитием ИТ-сервисов предприятия и оценка уровня их предоставления. В реальном времени получать количественные данные об ИТ-сервисах предприятия и получать обратную связь. Типовые возможности: поддержка встроенных инструментов проведения опросов; инструменты управления проектами.

8.Администрировани. чтобы быть уверенным в точности и актуальности информации. Типовые возможности: совместная дистанционная работа, управление безопасностью.

9.настриваемость (конфигурируемость) – удобство настройки инструмента для нужд конкретной организации.

10. Стандарты и методологии. Инструменты должны поддерживать различные стандарты и методологии и разработу текущей и целевой архитектруы в рамках методологии.

11.Удобство использования. Интуитивно понятный интерфейс, удобные и гибки возможности поделирования.

Инструменты управления АП, которые сегодня широко применяются на практике или открывают для компании большие возможности по реальному применению архитектурного подхода. Чаще всего можно столкнуться со следующими инструментами:

* Archi;
* Sila Union;
* Knowledge Space;
* Business Studio;
* HOPEX Platform (Mega).

Archi – программное обеспечение с открытым исходным кодом, распространяется бесплатно. Продукт ориентирован на язык Архимэйт. Поддержка русского языка в моделях, кроссплатформенность (виндоус, мак, линукс), наличие подробной документации. Недостатки: крайне органиченный инструментарий для формирования отчетности, невозможность многопользовательской работы.

Sila Union – отечественный продукт, удовлетворяет потребностям крупных компаний и опытных команд, гибкая настройка классификаций, обширные возможности по созданию отчетности, совместная работа. Недостатки: платформенное решение, высокий порог входа.

Работает как через браузер, так и десктоп. Русский язык, настройка прав доступ. Ключевое преимущество – возможность использовать обширные настройки набора языков моделирования и нотаций. Также поддерживается язык архимейт. Также сильная стороны – интерактивно декомпозировать различные объекты, также хранение различных документов, регламентов, материалов для реализации архитектурного процесса.

Knowledge Space – отечественная no-code платформа. Работают бизнес-аналитики без разработчиков – это плюс. Основные вохможности: возможность моделирования и анализа архитектуры (единый репозиторий) используются бпмн, архимейт и др.возможность дополнять графические модели количествнными.возможность создавать интерфейсы и приложения пользователей.

Business Studio – отечественное решение сфокусировано на моделировании бизнес-архитектуры и формировании отчетночти.(позволяет создавать систему сбалансированных показателй, орг диаграммы, осуществлять управление качеством, моделировать деятельность организации на основе нотаций (idef0,basic flowchar,epc,bpnm,sadt). Имитационное моделирование. Особенность – возможность осуществлять контроллинг процессов – получение из внешних информационных систем статичтических данных о выполнении ароцессов для последующего анализа. Поддерживает визуализации параметров процессов. Html-публикации. Недостатки: коммерческий характер, как и sila union.

HOPEX Platform (Mega) – от иностранной компании мега, поддерживает нотацию архимейт, возможности для формирования отчетов и построения дашбордов.

СППР (Система проектирования прикладных решений) – инструмент от фирмы 1С. Создана 10 лет назад. Является важной основой внедрения конфигураций 1С в компании. С помощью инструмента можно управлять изменениями процессов в организации и выстраивать целостную модель системы. СППР позволяет зафиксировать перечень автоматизируемых процессов, процессы при этом могут быть сгруппированы по усмотрению пользователя. При описании процесса фиксируется его описание, отражающее суть процесса, события начала и окончания процесса. Процесс детализируется до отдельных шагов, исполняемых конкретным исполнителем. Логическая модель в СППР строится с использованием методологии IDEF0. В рамках создания логической модели описываются функции системы и производится их декомпозиция. Основой описания функции является ее IDEF- схема. Схема позволяет в наглядной форме отразить взаимосвязь отдельных (дочерних) функций, потоков данных и исполнителей.  
Архимейт – инструмент управления архитектурой предприятия со свободным доступом к пользованию. Разработан в 2004 году. ArchiMate предлагает общий язык для описания построения и функционирования бизнес-процессов, организационных структур, информационных потоков, ИТ систем и технической инфраструктуры. Это помогает различным заинтересованным сторонам разрабатывать, оценивать и сообщать о последствиях решений и изменений внутри и между этими областями бизнеса. ArchiMate дистанцируется от других языков, таких как UML (Unified Modeling Language) и BPMN (Business Process Modeling and Notation) по предназначению для моделирования предприятия.  
Также, применение UML и BPMN подразумевается в узких областях и достаточно тяжело для понимания — они содержат около 150 (UML) и 250 (BPMN) сущностей, в то время как ArchiMate обходится 40 основными элементами, 18 дополнительными и 13 типами связей (в версии спецификации 3.0.1). Чтобы быть лёгким для изучения и применения, ArchiMate был намеренно ограничен.  
Elma – BPM-система, программное обеспечение обеспечивающее моделирование, проектирование, разработку бизнес-процессов компании и контроль выполнения работ. Система ELMA относится к коммерческому типу программного обеспечения. Первый релиз системы был в 2006 году. BPMS позволяет управлять компанией согласно четырем стадиям жизненного цикла бизнес-процессов:  
• Моделирование процессов с помощью условных графических обозначений нотации BPMN 2.0 на русском языке.  
• Исполнение процессов, в ходе которого рабочие задачи, заложенные в модель бизнес-процесса, автоматически поступают ответственным сотрудникам.  
• Контроль и мониторинг исполнения бизнес-процессов: сроки и качество выполнения задач, определенного участка или всего бизнес-процесса.

# Билет №4.

Язык моделирования архитектуры предприятия «Архимейт»: базовый фреймворк, элементы мотивации, бизнес-слой, прикладной и технический уровни.

Для моделирования необходима система понятий и обозначений, а также инструментальные средства составления и анализа моделей.В наст.время не существует единого общепризнанного языка, подходящего для всех случаев.

**1) ArchiMate- открытый, независимый язык поделирования АП, поддерживающий описание, анализ и визуализацию архитектуры внутри и через основные бизнес-домены.**

**ArchiMate** — профессиональный **язык** **моделирования**, который создан специально для описания корпоративной **архитектуры**. Он предоставляет богатый набор структурных и поведенческих элементов для детального описания структуры и функционирования **организации**. Кроме того, он включает в себя набор ясно определëнных графических и грамматических правил, которые позволяют составлять архитектурные модели более эффективно.

Название состоит из двух слов architecture и animate – архитектура, оживить. Разработан в нидерландах а 2004г. Три слоя архитектуры – бизнес-слой, слой приложения, технологический словй.

Бизнес-слой (Business layer) -- все то, что не относится к информационным технологиям. Уровень описывает продукты и сервисы для внешних клиентов, процессы реализации этих продуктов и сервисов, а также организационную модель.

Слой приложений (Application layer) -- описывает поддержку бизнес уровня ОТ-приложениями и основные виды данных.

Технологический слой (Technology layer) - описывает технологическую инфраструктуру, включающую аппаратное обеспечение, общесистемное программное обеспечение, необходимое для ОТ-приложений.

Например, системы хранения данных, каналы связи, ЦОДы, сервера и т. д.

2)**Базовый фреймворк**

Структура фреймворка позволяет моделировать предприятие с разных точек зрения, где положение внутри ячеек подчеркивает интересы заинтересованной стороны. У заинтересованной стороны обычно могут быть проблемы, охватывающие несколько ячеек.

Размеры каркаса следующие:

Уровни: три уровня, на которых можно моделировать предприятие – бизнес, приложения и технологии:

—  Бизнес-уровень предлагает продукты и услуги внешним заказчикам, которые реализуются в организации посредством бизнес-процессов.

—  Прикладной уровень поддерживает бизнес-уровень с помощью прикладных сервисов, которые реализуются (программными) приложениями.

—  Технологический уровень предлагает инфраструктурные услуги (например, услуги обработки, хранения и связи), необходимые для поддержки приложений, реализуемых компьютерным и коммуникационным оборудованием, а также системным программным обеспечением.

Аспекты:

—  Аспект активной структуры представляет структурные концепции (бизнес-субъекты, компоненты приложений и устройства, которые отображают реальное поведение; т.е. “субъекты” деятельности).

—  Поведенческий аспект представляет поведение (процессы, функции, события и сервисы), выполняемое участниками. Поведенческие концепции привязаны к структурным концепциям, чтобы показать, кто или что демонстрирует поведение.

—  Аспект пассивной структуры представляет объекты, над которыми выполняется поведение. Обычно это информационные объекты бизнес-уровня и объекты данных прикладного уровня, но они также могут использоваться для представления физических объектов.

Помимо основных аспектов, (пассивная структура, поведение и активная структура), которые в основном носят операционный характер, работа корпоративного архитектора затрагивает множество других аспектов, явно не охваченных ArchiMate Framework, некоторые из которых могут пересекаться с несколькими (или всеми) концептуальными областями; например:

•           Цели, принципы и требования

•           Риски и безопасность

•           Управление

•           Политики и бизнес-правила

•           Затраты

•           Производительность

•           Хронометраж

•           Планирование и эволюция

Не все эти аспекты могут быть полностью охвачены с помощью стандартных механизмов расширения языка. Чтобы облегчить поставщикам инструментов и экспертам по методологии обеспечение поддержки этих аспектов в рамках общего языка ArchiMate, могут быть добавлены специальные расширения. Это модульное расширение добавляет новые концепции, взаимосвязи или атрибуты, при этом соблюдая ограничение дизайна, заключающееся в том, что ArchiMate явно разработан таким образом, чтобы быть как можно меньше.

Также может оказаться полезным добавить концепции или атрибуты, относящиеся к процессу проектирования, а не к системе или организации, которые должны быть описаны или спроектированы. Примерами таких концепций или атрибутов являются требования и проектные решения.

**3)Элемент** **мотивации** в нотации **Archimate** представляет собой абстрактный архитектурный **элемент**, который используется для описания факторов, мотивов, целей и ограничений, влияющих на архитектуру предприятия. Эти **элементы** позволяют моделировать и анализировать, что мотивирует организацию к изменениям в своей архитектуре, какие стратегические цели она ставит перед собой, какие потребности стейкхолдеров необходимо удовлетворить, и какие ограничения и требования важны для архитектурных решений.

Краткое обозначение элементов мотивационного слоя:

Изображение выглядит как снимок экрана, программное обеспечение, Мультимедийное программное обеспечение, текст

Автоматически созданное описание

**Пример для понимания:**

Давайте рассмотрим такой пример, где есть три стейкхолдера — Генеральный директор, Директор по маркетингу и Финансовый директор. Генеральный директор хочет повысить прибыль компании и увеличить долю рынка, для чего ставит соответствующие цели перед директором по маркетингу и финансовым директором. Директор по маркетингу предлагает разработать новую платформу, которая была бы более привлекательной для клиентов, что предотвратило бы отток клиентов к конкурентам. А финансовый директор предлагает внедрить CRM систему с целью снижения затрат на привлечение новых клиентов.

На первый взгляд и то и другое решение ведет к решению поставленной генеральным директором задачи, но если одновременный запуск двух проектов не возможен, то нам нужно выбрать, какой из них приоритетнее. Давайте посмотрим на приведенную диаграмму.

Изображение выглядит как текст, снимок экрана, диаграмма, Шрифт

Автоматически созданное описание

Итак, мы видим, что оба решения приведут к достижению поставленных целей. Но первый проект, помимо решения стратегической цели — увеличения доли рынка, решает и другую задачу — повышение рентабельности за счет повышения цены (снижения скидки), соответственно он имеет явное преимущество перед вторым, предполагающим сокращение затрат. И хотя финансовый директор изначально предлагал другой проект, его цели и интересы будут выполнены, реализацией проекта разработки новой платформы.

Таким образом, выбор приоритетного проекта осуществляется на основе множества факторов, включая стратегические цели, ожидания стейкхолдеров, влияние проектов друг на друга, а также доступность ресурсов. Мотивационная диаграмма позволяет все эти связи и влияния изобразить в кратком и наглядном виде, что значительно облегчает сбор информации, переговоры и принятие конечного решения.

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

**Прикладной уровень** использует прикладные сервисы, реализованные (программными) приложениями для поддержки бизнес-уровня. **Технический уровень** предоставляет услуги инфраструктуры (такие как услуги обработки, хранения и связи), необходимые для запуска приложений, которые реализуются компьютерным и коммуникационным оборудованием и системным программным обеспечением.

4) На **бизнес-слое** моделируются бизнес-процессы, организационные структуры, продукты и услуги, а также информация, необходимая для бизнеса. Этот слой позволяет лучше понимать, как бизнес-процессы взаимодействуют друг с другом и как они поддерживают выполнение стратегических целей.

# Билет №5.

**Теория.** Нотация моделирования BPMN: область применения, эволюция нотации, графические элементы (элементы потока, данные, соединяющие элементы, зоны ответственности, артефакты), подпроцессы, межпроцессное взаимодействие.

Нотация моделирования BPMN (Business Process Model and Notation) - это стандартный язык для моделирования бизнес-процессов, который помогает компаниям визуализировать, анализировать и оптимизировать свои бизнес-процессы.  
В работе бизнес-аналитиков нотация BPMN служит не только инструментом создания диаграмм, но и средством коммуникации между участниками проекта. Диаграмма эффективна в отображении различных бизнес-сценариев от взаимодействия клиента с сервисом до управления проектами. Рассмотрим некоторые примеры использования нотации BPMN в бизнес-анализе:  
• Визуализация бизнес-процессов. Диаграммы BPMN понятны и легки для использования в качестве инструмента коммуникации между различными участниками проекта. К примеру, можно создать визуальную схему для процесса заказа в интернет-магазине, от выбора товара до оформления покупки.  
• Анализ и моделирование процессов. Позволяет легко обнаруживать улучшения и оптимизационные возможности. Можно смоделировать процесс взаимодействия клиента с онлайн сервисом, включая этапы регистрации, выбора услуг и оплаты.  
• Определение бизнес-требований. Использование BPMN помогает конкретизировать бизнес-требования к процессам, включая входящие и исходящие данные, взаимодействие между участниками и условия выполнения. Например, формулирование бизнес-требований для саппорта клиентов, включая обработку обращений, определение приоритетов и ответы на вопросы.  
• Управление проектами. BPMN позволяет структурировать и визуализировать рабочие процессы, что облегчает управление проектами и взаимодействие между участниками. В частности, использование схемы BPMN для создания календарного графика и рабочих планов проекта разработки нового программного продукта.  
Первая версия BPMN была создана в 2004 году рабочей группой IBM. В 2010 году она была дополнена и выпущена под названием BPMN 2.0. В неё добавили новые типы событий и диаграмм, а также устранили ошибки первой версии.  
Элементы:  
В диаграммах BPMN роли и зоны ответственности играют ключевую роль в определении участников, их ответственности и взаимодействия в рамках бизнес-процесса.  
Роли отображают разных участников или группы в рамках бизнес-процесса. Это может быть отдельное лицо, команда, отдел или внешняя организация. На диаграмме роли отражаются в виде пула (Pool).  
Зона ответственности или «дорожка» (Lane) определяет конкретную область или отдел, отвечающий за выполнение определенных функций в бизнес-процессе. Lines могут быть частью пула и указывать на распределение ответственности между ролями. «Дорожки» отображаются как горизонтальные или вертикальные разделители в пуле.  
Объекты потока управления (Flow Objects) в нотации BPMN отображают последовательность действий, событий, решений и условных переходов в бизнес-процессах. Они определяют, как данные и управление передаются от одного состояния к другому в рамках процесса.  
Происходящие в процессе моменты, такие как старт, завершения или промежуточные, называются событиями (Event). Например, это может быть продажа услуги или разговор с клиентом. На диаграмме события обозначаются кружочком или кружочком с иконкой, указывающей тип события (таймер, ошибка).  
Артефакты в нотации BPMN используются для предоставления дополнительной информации, аннотаций и спецификаций. Помогают расширять понимание процессов и обеспечивать контекстуальную информацию, но не влияют на выполнение бизнес-процесса напрямую.  
Основные типы артефактов:  
• Данные (Data Object): Определяют данные, которые используются или генерируются в бизнес-процессе. Обозначаются прямоугольником с заголовком «Data Object».  
• Группа (Group): Используются для логического группирования элементов диаграммы, чтобы предоставить большую структуру и организацию. Изображены прямоугольником со стрелкой, указывающей на группировку, а также содержит имя группы.

Текстовые аннотации (Text Annotation): Дополнительное описание или объяснение элементов или связей на диаграмме. Представляется линией без стрелки, соединяющей элементы или артефакты. Ассоциации часто используются для обозначения взаимосвязей без указания на порядок или направление.  
Основные типы соединяющих объектов включают поток управления, поток сообщений и ассоциацию.  
• Поток управления (Sequence Flow): Определяет порядок выполнения действий и событий в бизнес-процессе. Отмечается стрелкой, указывающей направление потока.  
• Поток сообщений (Message Flow): Определяет обмен сообщениями между элементами бизнес-процесса, которые могут находиться в разных ролях или пулах. Также изображается стрелкой, но с дополнительными линиями или маркерами, которые конкретизируют, что это за сообщение.  
• Ассоциация (Association): Используется для увязки артефактов, текстовых аннотаций или других элементов с объектами процесса, не влияя на ход выполнения процесса.  
Подпроцессы:  
Встраиваемые подпроцессы: Можно использовать для описания уровня детализации в рамках самого процесса.  
Вызываемые подпроцессы: Могут быть использованы для упрощения моделирования сложных процессов путем разделения на отдельные части.  
Межпроцессное взаимодействие:  
Отправка и получение сообщений: Показывает передачу информации между процессами. События начала и завершения связи: Определяют начало и конец взаимодействия между процессами

**Бизнес-процесс** – целенаправленная последовательность процедур, предназначенная для получения заданного конечного результата.

**Процессный подход** – применение для управления деятельностью и ресурсами организации системы взаимосвязанных процессов.

**Сеть бизнес-процессов организации** – совокупность взаимосвязанных и взаимодействующих бизнес-процессов, включающих все функции, выполняемые в подразделениях организации.

**Автоматизация бизнес-процессов** – реализация системы управления бизнес-процессами с помощью информационных технологий.

**Workflow** – это система обеспечения выполнения задач, поставленных перед исполнителями в рамках процессного управления. Т.е. **Workflow** – система, координирующая выполнение операций, составляющих бизнес-процессы.

Business Process Modeling Notation (BPMN) – стандартизированная модель описания бизнес-процессов, связующее звено между разработкой бизнес-процессов и их реализацией.

Основное назначение стандарта – создание нотации, понятной всем участникам бизнес-сферы, от бизнес-аналитиков, создающих первоначальные эскизы процессов, технических разработчиков, ответственных за внедрение технологии, в которой будут представлены данные процессы, и, наконец, до бизнесменов, которые будут управлять этими процессами, а также осуществлять их мониторинг.

Нотация BPMN была разработана организацией "Business Process Management Initiative (BPMI)", и поддерживается группой компаний "Object Management Group", после слияния организаций в 2005 году.

Область применения

BPMN поддерживает лишь набор концепций, необходимых для моделирования бизнес-процессов. Моделирование иных аспектов, помимо бизнес-процессов, находится вне зоны внимания BPMN. Например, моделирование следующих аспектов не описывается в BPMN:

-организационная структура;

-функциональный анализ.

Несмотря на то, что BPMN позволяет моделировать потоки данных и потоки сообщений, а также ассоциировать данные с действиями, она не является схемой информационных потоков.

Нотация BPMN предназначена для описания:

* Порядка исполнения работ образующих бизнес-процесс,
* Потоков данных между операциями процесса;
* Потоков сообщений между процессами;
* Ассоциации обрабатываемых объектов данных с операциями процесса.

Моделирование осуществляется с помощью визуальных диаграмм, что позволяет участникам быстрее понять логику исполнения.

Нотация BPMN не позволяет моделировать другие аспекты модели бизнес-процесса, например:

• Функциональную (структурную) декомпозицию работ;

• Организационную структуру предприятия;

• Модель данных;

• Бизнес правила,

• Бизнес стратегию компании

Поскольку интегрированная модель бизнес-процесса включает не только поведенческую перспективу, но также другие аспекты, описываемые перечисленными моделями, спецификация BPMN уделяет повышенное внимание вопросам интеграции моделей.

Если резюмировать, то нотацию BPMN бизнес использует для решения двух задач:

* чтобы подробно описать процесс через BPMN: наглядно зафиксировать последовательность шагов, логику действий, всех участников и объекты, участвующие в реализации;
* для цифровизации с помощью специализированных [ИТ-систем](https://www.directum.ru/company/news-analytics/bpm), которые поддерживают описание процессов и их исполнение, то есть workflow.

Эволюция

Аббревиатура BPMN означает дословно Business Process Model and Notation и переводится как «нотация и модель бизнес-процессов». До сих пор актуальной остается версия 2.0, вышедшая в 2012 г. С 2013 года BPMN имеет статус международного стандарта ISO/IEC 19510. Большинство компаний, внедряя автоматизированные бизнес-процессы, используют методологию BPMN для моделирования и проектирования исполняемых процессов.

В спецификации BPMN 2.0 (2011 г.) содержится более подробное, нежели содержащееся в спецификации BPMN 1.2, описание возможностей и областей использования нотации, а именно:

* формализация семантики исполнения всех существующих элементов BPMN;
* определение механизма изменения как для расширений модели бизнес-процесса, так и для расширений графических элементов;
* детализация состава и корреляции **События**;
* расширение определения пользовательских действий;
* введение элемента **Хореография** (**Choreography**) и описание данной модели.

Данная спецификация также разрешает некоторые несоответствия и неточности спецификации BPMN 1.2.

История:  
• BPMN 1.0 - первая версия нотации была выпущена в 2004 году и представляла собой базовый набор элементов для моделирования бизнес-процессов.  
• BPMN 1.1 - версия была выпущена в 2006. дополнительные типы соединений и элементы для моделирования сложных процессов.  
• BPMN 2.0 - в 2011 году. Новые элементов и функций, таких как поддержка событий, правил и подпроцессов это расширил возможности визуализации. Новые элементы и функции позволили пользователям создавать более сложные и структурированные модели, что сделало BPMN 2.0 предпочтительным выбором для многих компаний и организаций.

графические элементы (элементы потока, данные, соединяющие элементы, зоны ответственности, артефакты),

Изображение выглядит как текст, снимок экрана, диаграмма, Параллельный

Автоматически созданное описание

Моделирование в BPMN осуществляется посредством простых диаграмм с небольшим числом графических элементов. Это помогает пользователям быстро понимать логику процесса.

Существует пять основных категории элементов:

* элементы потока – основные графические элементы для определения поведения бизнес-процесса. К ним относятся: события, действия и шлюзы;
* данные: объекты данных, входные и выходные данные, хранилища данных;
* соединяющие объекты: поток операций, поток сообщений и ассоциации;
* роли: пулы и дорожки;
* артефакты: группы и текстовые аннотации. Под артефактами в BPMN понимают объекты, которые не влияют на исполнение бизнес-процесса напрямую. Это могут быть документы, данные, информация.

Элементы этих пяти категорий позволяют строить простейшие диаграммы бизнес-процессов. Для повышения выразительности модели спецификация разрешает создавать новые типы объектов потока управления и артефактов.

Изображение выглядит как текст, снимок экрана, диаграмма, Параллельный

Автоматически созданное описание

подпроцессы,

**Подпроцесс** представляет собой действие, заключающее в себе другие действия, шлюзы, события и потоки операций. Графически подпроцесс изображается в качестве элемента потока операций – процесса.

В практике описания бизнес-процессов элемент нотации BPMN **Подпроцессы** используется в основном в двух случаях:

1. Для декомпозиции и повышения читаемости и наглядности схем (диаграмм).
2. Для описания повторяющихся действий. Единожды описанный Подпроцесс может многократно вызываться (использоваться) внутри различных процессов.

**Подпроцесс** может быть **свернутым**: при этом его детали скрыты. Подпроцесс также может быть развернутым, при этом его детали отображаются внутри процесса, в котором данный подпроцесс содержится. В случае, если подпроцесс является свернутым, то используется маркер, позволяющий отличить подпроцесс от задачи. **Развернутый** **подпроцесс** используется для более компактного отображения группы действий с использованием минимума деталей. Первый случай использования подпроцессов – **декомпозиция процесса**. Довольно часто при описании бизнес-процессов компании для наглядности используют схемы (диаграммы), отражающие верхние уровни организации работы. В этом случае диаграмма отображает "суть" процессов и нацелена на понимание логики процесса без знания деталей. Декомпозиция процесса (разбивка на подпроцессы) позволяет моделировать и вносить изменения в рамках каждого подпроцесса, не изменяя весь основный процесс целиком. **Подпроцессы** являются комплексными задачами в рамках основного процесса. Однако стоит отметить, что **Подпроцессы**, как элемент BPMN, являются не самостоятельными задачами, а лишь отсылкой к другому процессу.

Изображение выглядит как текст, Шрифт, снимок экрана, линия

Автоматически созданное описание

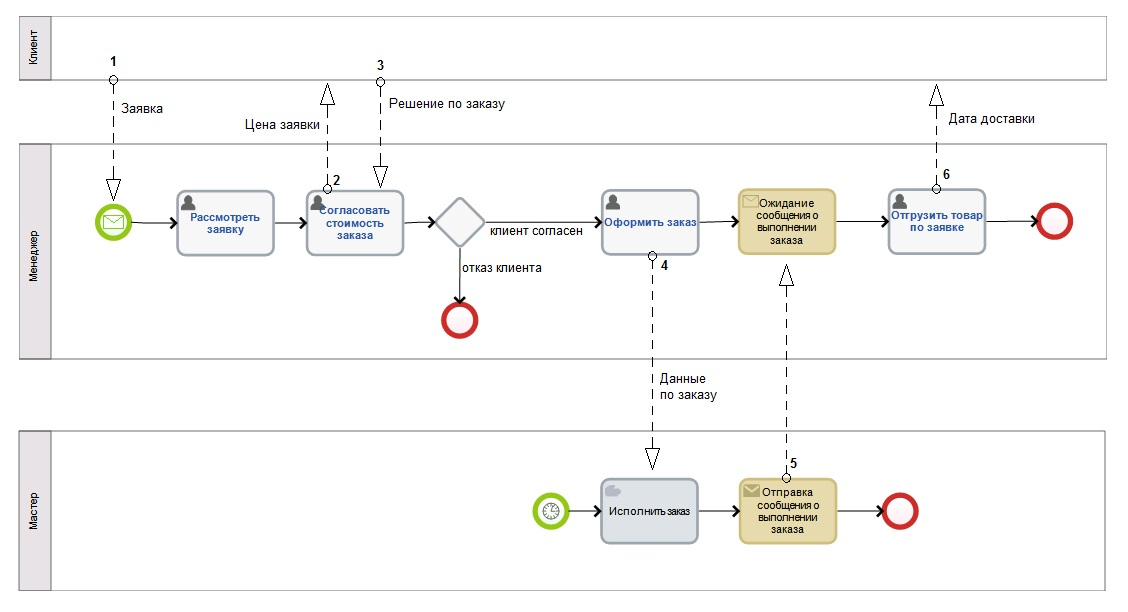
межпроцессное взаимодействие.

**Взаимодействие** представляет собой совокупность участников процесса, представленных в виде **Пулов**, и взаимоотношений, представленных в виде **Потоков Сообщений**.

Поток сообщений может соединять только два разных пула (участников процесса) между собой или элементы, расположенные внутри этих пулов. Однако, поток сообщений не может соединять два элемента, расположенные внутри одного и того же пула. Для этого используется поток операций, который используется только в рамках одного пула.

Поток сообщений не отображает ход выполнения процесса, а показывает передачу сообщений или объектов из одного процесса в другой процесс или внешнюю ссылку. Существует четыре основных варианта использования потока сообщений (Рис. 15):

1. Поток сообщений представляет собой механизм запуска процесса (Вариант 1).
2. Поток сообщений используется для передачи сообщений или объектов от одного действия рассматриваемого процесса во внешний процесс или внешнюю ссылку (Вариант 2).
3. Поток сообщений используется для передачи сообщений или объектов из внешнего процесса (или внешней ссылки) в одно из действий рассматриваемого процесса (Вариант 3).
4. Поток сообщений используется для передачи сообщений или объектов от одного из действий внешнего процесса в одно действие рассматриваемого процесса (Вариант 4).



Варианты 1, 2 и 3 использования Потока сообщений с участием клиента применяется только в рамках аналитических диаграмм. Такие схемы не являются исполняемыми, так как клиент не является участником BPMS-систем.

# **Билет №6.**

**Теория.** Система управления процессами Элма: проектирование (моделирование процессов), исполнение, контроль и мониторинг, улучшение процессов. **Внедрение BPMS-систем в компании,** **практическая польза применения BPMS-систем в организации.**

BPMS, Business Process Management System — это система управления бизнес-процессами

Элма – это системы управления бизнес-процессами, которая позволяет на этапе внедрения системы осуществить моделирование бизнес-процессов организации.

Все функции системы поделены на четыре группы в соответствии со стадиями жизненного цикла (цикл Деминга) процесса PDCA (Plan, Do, Check, Act/планируй, делай, проверяй,управляй):

* Проектирование (моделирование)
* Исполнение
* Контроль и Мониторинг
* Улучшение процессов

\*Метод Деминга широко применяется для непрерывного улучшения качества продукта и производственных процессов. PDCA являет собой простейший алгоритм действий руководителя, позволяющий управлять процессом и достигать его целей. Эффективность методологии достигается за счет постоянных проверок до, во время и после процесса производства. Такой постоянный аудит позволяет обнаруживать слабые места в работе предприятия

В системе ELMA бизнес-процессы **моделируются** и **улучшаются** в Дизайнере ELMA.

Все созданные, настроенные и опубликованные процессы **исполняются** в веб-приложении. При этом каждый созданный в Дизайнере бизнес-процесс может исполняться произвольное количество раз в веб-приложении. Бизнес-процесс, создаваемый в Дизайнере ELMA, называется моделью бизнес-процесса, бизнес-процессом или просто процессом. Запущенный в веб-приложении бизнес-процесс называется экземпляром бизнес-процесса или просто экземпляром процесса.

ПРОЕКТИРОВАНИЕ (МОДЕЛИРОВАНИЕ).

Работа с процессами в системе ELMA BPM начинается с их моделирования. Моделирование осуществляется в программе Дизайнер ELMA, которая входит в комплект системы. Это простой графический редактор для построения бизнес-процессов. Создавать процессы и выполнять их моделирование в системе могут бизнес-аналитики без участия программистов. Для этого используется распространенная нотация BPMN 2.0 (Business Process Management Notation)— набор простых условных обозначений. Нотация переведена на русский язык специалистами компании ELMA. Создаваемые с ее помощью модели процессов привычны для аналитиков и понятны высшему руководству компании. На диаграммах бизнес-процессов обозначаются их участники, перечень и последовательность выполняемых ими действий. Графические модели строятся путем размещения нужных блоков операций на дорожках, обозначающих зоны ответственности разных исполнителей. Затем операции соединяются переходами. Когда модель процесса создана, проводится настройка операций (задается контекст работы). Например, организация определяет способ выполнения пользовательских задач — с какими данными будут работать исполнители, какие действия от них потребуется выполнить и какие результаты предоставить. Моделирование процесса заканчивается его публикацией, после чего бизнес-процесс компилируется и становится доступным для запуска в веб-приложении ELMA.

ИСПОЛНЕНИЕ

В BPM-системе можно одновременно запускать сколько угодно экземпляров одного бизнес-процесса. Все они будут исполняться независимо, используя в своей основе одну и ту же модель. Сотрудники компании работают по процессам в веб-интерфейсе пользователя ELMA — заходят в систему через браузер по логину и паролю. Работа по бизнес-процессу осуществляется в соответствии с его графической моделью — начиная со стартового события, следуя по цепочке переходов, до конечного события. ELMA позволяет поддерживать выполнение правильной очередности операций. Поручения, которые получают сотрудники в системе ELMA, оформлены в виде карточек задач. В них уже есть вся нужная информация для работы и принятия решений. От исполнителей может требоваться ввод некоторых данных — в этом случае карточка задачи будет содержать соответствующие поля, заполнение которых можно сделать обязательным. После завершения задачи, сотруднику не нужно задумываться о том, кто и какие действия должен выполнять дальше — ELMA BPM берет эти вопросы на себя. Каждый бизнес-процесс компании имеет четкий регламент исполнения. Система обеспечивает соблюдение верного порядка работы.

КОНКТРОЛЬ и МОНИТОРИНГ

Для каждого экземпляра запущенного процесса создается карточка в системе ELMA. Она содержит подробную информацию: текущие значения параметров, задачи, статусы их выполнения и комментарии сотрудников. Имея соответствующие права доступа, с карточки можно управлять ходом работы — например, изменить ряд параметров или вовсе прекратить исполнение.

Организация бизнес-процессов в системе ELMA открывает широкие возможности мониторинга их выполнения. В распоряжении пользователей есть несколько инструментов:

* Страница «Мои процессы» содержит список всех экземпляров бизнес-процессов, в которых сотрудник является инициатором, ответственным или исполнителем.
* Страница «Монитор процессов» позволяет анализировать работу по процессам, в которых пользователь выступает в роли владельца, куратора или информируемого. На этой стадии управления сотруднику не обязательно являться исполнителем какой-либо из операций. Со страницы «Монитор процессов» можно перейти на карточку любого из отображаемых процессов, чтобы получить подробную информацию.
* Карта процесса — еще один удобный способ мониторинга. Она выглядит так же, как и графическая модель процесса в Дизайнере ELMA. При этом на карте отображается уже запущенный в системе экземпляр конкретного бизнес-процесса, что позволяет отслеживать дополнительные метки. Здесь обозначаются уже выполненные операции и текущая стадия работы. Открыть карту процесса можно из его карточки в системе.

УЛУЧШЕНИЕ

На основании мониторинга работы, а также в ходе развития компании часто возникает потребность в изменении бизнес-процессов. Рациональные идеи по оптимизации могут предложить исполнители операций. Благодаря использованию BPM-системы любой бизнес-процесс остается гибким к изменениям. Компания может в любое время корректировать порядок работы и создавать новые процессы. Технология BPM поддерживает идею непрерывного развития бизнес-процессов. Не получится выстроить работу один раз и тем самым обеспечить постоянную эффективность предприятия. С развитием компании, увеличением штата сотрудников, новыми реалиями в рынке возникает необходимость изменять процессы. В ELMA оптимизация проводится на лету без остановки системы. Когда вы изменяете bpm процесс в компании, это никаким образом не влияет и не прерывает текущую работу предприятия. Все экземпляры данного процесса, которые уже были запущены в системе на момент внесения изменений, завершатся в соответствии с первоначальной схемой. А каждый новый запущенный процесс будет работать уже согласно улучшенной модели.

Внедрение BPMS-систем в компании.

Первым этапом внедрения любой BPMS-системы является проведение полного анализа деятельности компании: сбора информации, обследования по направлениям, составления реестра бизнес-процессов и описания ситуации "как есть", фиксирования существующих проблем компании. Данный этап очень важен, так как определяет эффективность последующей оптимизации за счёт автоматизации.

Вся информация по существующим бизнес-процессам компании собирается за счёт интервьюирования участников процесса о выполняемых действиях, входящих и выходящих данных, форм документов и т.д. Основываясь на полученных данных, описываются существующие процессы в компании и разрабатываются методики по их оптимизации. Только после полного описания картины "To be" ("Как должно быть") деятельности компании может устанавливаться система BPMS с гарантированным эффектом от автоматизации.

Несколько этапов: подготовка, настройка, обучение персонала.

Практическая польза применения BPMS-систем в организации.

1. Возможность непрерывного совершенствования бизнес-процессов и, соответственно, ускорение реализации бизнес-целей: увеличение выручки, прибыли, стоимости бизнеса и т.п.
2. Сокращение времени выполнения процессов за счет их регламентации и автоматизации отдельных шагов. Каждый исполнитель точно знает, кто, что и в какой последовательности делает. Если какие-то действия возможно переложить на машину, то это будет делать машина без участия человека или с его минимальным участием.
3. Уменьшение доли ошибочных операций за счет того, что бизнес-процесс "направляется" системой. Соответственно, уменьшение доли брака, повышение качества производимых продуктов и оказываемых услуг.
4. Обеспечение соответствия требованиям SLA (уровень предоставления сервиса). Для каждой задачи каждого процесса могут быть определены временные ограничения для соответствия конкретным показателям SLA и BPM-система будет автоматически их контролировать, выдавать предупреждения при срыве сроков и/или предлагать выполнить предопределенные корректирующие воздействия.
5. Существенное повышение прозрачности операционной деятельности. Например, по каждому из десятков, сотен или тысяч одновременно выполняющихся заказов клиентов все заинтересованные лица смогут посмотреть текущую стадию его исполнения, ответственных лиц и прочую сопутствующую информацию.
6. Описания бизнес-процессов становятся "живыми", а организация гораздо более гибкой. Для того, чтобы внести корректировки в бизнес-процессы, описанные на бумаге (в виде схем, регламентов, должностных инструкций), требуются дни, недели или месяцы. Изменения в бизнес-процессы, выполняемые в BPM-системе, могут быть внесены за считанные минуты.
7. Улучшение качества корпоративной информации. В процессе выполнения бизнес-процессов появляется информация, которая должна найти отражение в различных информационных системах, используемых в организации. BPM-решение обеспечивает интеграцию этих систем на уровне бизнес-процессов и гарантирует, что необходимая информация попадет в требуемые системы в требуемом виде и в требуемое время, чего достаточно сложно добиться при ручном внесении информации в системы.

# **Билет №7.**

Основные парадигмы теории Больших данных. Правило Мура. Правила Амдала. Виды распределенных систем в контексте больших данных.

Парадигма – это совокупность теоретических и методологических положений, принятых научным сообществом на известном этапе развития науки и используемых в качестве образца, модели, стандарта для научного исследования, интерпретации, оценки и систематизации научных данных, для осмысления гипотез и решения задач, возникающих в процессе научного познания.

Таким образом, парадигмы теории Больших данных представляют собой основные принципы или подходы, которые используются для работы с огромными объемами данных.

Большие данные ‒ наличие данных больше, чем 100 Гб; данные, которые невозможно обрабатывать в Excel, т. е. традиционным способом, и которые невозможно обработать на одном компьютере.

Big Data ‒ это технологии работы с информацией огромного объема и разнообразного состава, часто обновляемой и находящейся в разных источниках («большие данные») в целях увеличения эффективности работы, создания новых продуктов и повышения конкурентоспособности предприятия.

Термин «большие данные» слышал почти каждый, но часто он употребляется ошибочно. Существует шесть основных критериев или шесть «V», которые помогут определить, что перед вами Big Data:

1. Volume (объем) — информации должно поступать более 150 Гб в сутки.
2. Velocity (скорость) — для работы с массивами информации в режиме реального времени требуются повышенные вычислительные мощности.
3. Variety (разнообразие) — поступающая информация имеет разные форматы или степень структурированности. Например, контент социальных сетей может сильно различаться даже в пределах одной страницы.
4. Veracity (достоверность) — источникам данных можно доверять, а результат их обработки обладает достоверностью, достаточной для принятия решений.
5. Variability (вариативность) — поток данных изменчив, на него может влиять даже время суток или погода. Например, в час пик приходит больше данных от таксистов.
6. Value (ценность) — данные могут иметь разное значение для компании. Например, сделки с крупными покупателями имеют большее значение, чем с мелкими.

Направления работы по управлению Big Data должны основываться на определенных принципах.

* 1. Горизонтальная масштабируемость. Поскольку данных может быть очень много, то любая система, которая подразумевает обработку больших данных, должна быть расширяемой.
  2. Отказоустойчивость. Методы работы с большими данными должны учитывать возможность выхода из строя машин (а их может быть много – до нескольких тысяч) и способность преодолевать эти проблемы без каких-либо значимых последствий.
  3. Локальность данных. В больших распределенных системах данные рассредоточены по большому количеству машин. Принцип локальности данных заключается в том, чтобы по возможности обрабатывать данные на той же машине, где они хранятся.

Для того чтобы следовать этим принципам, необходимы технологии (включают разные методы и способы) средств обработки данных. С учетом принципов и направлений решения вопросов управления большими данными можно так определить Big Data: это горизонтально масштабируемая система, основанная на определенных принципах, использующая набор методик и технологий, позволяющих обрабатывать структурированную и неструктурированную информацию и строить связи, необходимые для получения однозначно интерпретируемых человеком данных, не успевших потерять актуальность, и несущая ценность для достижения поставленных целей

Правило Мура.

Количество транзисторов на интегральной микросхеме удваивается примерно каждые два года. То есть со временем на микросхеме можно разместить больше транзисторов. Таким образом, плотность интеграции транзисторов на микросхемах увеличивается, а их стоимость — снижается. Итог очевиден: технологии становятся эффективнее и дешевле.  
В основе закона — неизбежность технологического прогресса. Уменьшение размеров транзисторов и других компонентов на микросхемах позволяет производить более миниатюрные и эффективные транзисторы, что в свою очередь позволяет увеличивать их количество на чипе.  
Большее количество транзисторов позволяет увеличивать вычислительную мощность компьютера, добавлять новые функции и возможности и повышает энергоэффективность устройств.

Правила Амдала.

Закон Амдала — это формула, демонстрирующая потенциал ускорения вычислительной задачи, которого можно достичь при увеличении количества ресурсов системы. Обычно он используется в параллельных вычислениях, и может предсказать наличие реальных преимуществ от увеличения количества процессоров с учётом ограничений параллелизуемости программы.  
Приведём пример. Если программа состоит из двух частей – части А, которую необходимо исполнять на одном процессоре, и части Б, которую можно распараллелить, видно, что преимущества добавления нескольких процессоров к исполняющей программу системе ограничены. Потенциально это может сильно ускорить часть Б – но скорость части А не изменится.

Как видно, даже если 50% программы можно распараллелить, преимущества от добавления более 10 отдельных процессоров будут незначительными. Если программу можно распараллелить на 95%, улучшения будут заметны даже после добавления тысячи процессоров.  
С замедлением закона Мура и ускорения процессоров параллелизация становится ключом к улучшению эффективности. Прекрасным примером будет программирование графики – современное программирование на основе шейдеров позволяет отрисовывать фрагменты изображения параллельно, поэтому в современных графических картах можно встретить тысячи процессорных ядер (GPU или шейдерных модулей).

Переход от локальной системы (располагающейся на одном компьютере) к распределённой — трудоёмкий и дорогостоящий процесс. В каких случаях это необходимо? Здесь возможны разные варианты: • необходимость в обработке больших объёмов данных; • обслуживание географически распределённых пользователей; • распределение нагрузки; • увеличение производительности; • повышение отказоустойчивости; • предоставление совместного доступа к некоторому общему ресурсу. Кроме того, как показывают исследования, горизонтальное масштабирование вычислительной системы (переход от системы с одним узлом к кластеру) может быть экономически выгоднее вертикальной масштабируемости (увеличение мощности имеющегося сервера).

В контексте больших данных распределенные системы играют важную роль в обеспечении эффективной обработки данных, распределенного хранения и управления данными. Эти системы распределяют данные и задачи обработки по нескольким компьютерам или узлам в сети, обеспечивая параллельную обработку и высокую отказоустойчивость.

Вот несколько основных видов:

Системы хранения данных (Distributed Storage Systems):

Hadoop Distributed File System (HDFS): Распределенная файловая система, используемая в экосистеме Apache Hadoop для хранения больших объемов данных.

Amazon S3: Облачное хранилище Amazon Web Services, предоставляющее высокую доступность и масштабируемость для хранения данных в облаке.

Системы обработки данных (Distributed Processing Systems):

Apache Spark: Фреймворк для обработки данных в памяти, поддерживающий распределенные вычисления и обработку больших объемов данных.

Apache Flink: Распределенная система обработки потоков данных, обеспечивающая низкую задержку и высокую пропускную способность.

Системы управления потоком данных (Stream Processing Systems):

Apache Kafka: Платформа для обработки и управления потоками данных в реальном времени, обеспечивающая отказоустойчивость и масштабируемость.

Apache Storm: Распределенная система обработки потоков данных, предназначенная для высокоскоростной обработки данных в режиме реального времени.

Системы управления кластерами (Cluster Management Systems):

Apache Mesos: Кластерный менеджер, обеспечивающий ресурсное управление и планирование задач для распределенных систем.

Kubernetes: Оркестратор контейнеров, который управляет развертыванием, масштабированием и управлением приложениями в контейнерах на кластере.

Слово *«*парадигма» происходит от греческого paradeigma, *«*образец, пример*»*, от глагола  paradeiknumi, *«*демонстрировать, представлять, выставлять*»,* а также от  para*, «*рядом, за пределами*»* и deiknumi*, «*показывать, указывать*».*

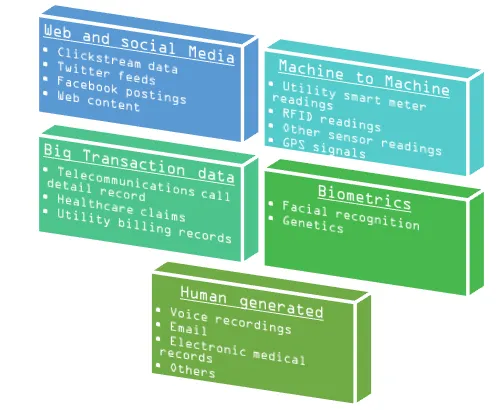
В современном научном и философском понимании **парадигма представляет собой отдельный набор концепций или моделей мышления, включая теории, методы исследования, постулаты и стандарты того, что представляет собой законный вклад в определенную область**.

В контексте вычислений этот термин (Большие данные) появился в 1997 году (Press, 2014). В статье под названием «Пейджинг по запросу, управляемый приложением для внешней визуализации», авторы, оба из Исследовательского центра Эймса MRJ/НАСА, упомянули, что большие наборы данных (использовавшиеся в то время для визуализации в области вычислительной жидкости) динамика) предполагал проблему с точки зрения способности основной памяти компьютера эффективно генерировать соответствующие визуализации. Авторы назвали это проблемой больших данных (Cox and Ellsworth, 1997).

Больши́е да́нные (англ. big data) — обозначение структурированных и неструктурированных данных огромных объёмов и значительного многообразия, эффективно обрабатываемых горизонтально масштабируемыми программными инструментами, появившимися в конце 2000-х годов и альтернативных традиционным системам управления базами данных и решениям класса Business Intelligence.

В 2001 году Лэйни (2001) написал короткую статью, в которой проблемы BD были синтезированы с помощью метода 3 Vs. Эти проблемы были связаны с управлением данными в результате электронной коммерции; это (1) Объем, (3) Скорость, (3) Разнообразие. С тех пор 3V используются для определения или описания контекста BD (Press, 2014).

По данным Шима и др. (2015), основываясь на статье, опубликованной консалтинговой фирмой Gartner, существует пять больших групп данных, которые собирают организации:



Методы и техники анализа, применимые к большим данным, выделенные в отчёте McKinsey:

- методы класса Data Mining: обучение ассоциативным правилам (англ. association rule learning), классификация (методы категоризации новых данных на основе принципов, ранее применённых к уже наличествующим данным), кластерный анализ, регрессионный анализ;

- краудсорсинг — категоризация и обогащение данных силами широкого, неопределённого круга лиц, привлечённых на основании публичной оферты, без вступления в трудовые отношения;

- смешение и интеграция данных (англ. data fusion and integration) — набор техник, позволяющих интегрировать разнородные данные из разнообразных источников для возможности глубинного анализа, в качестве примеров таких техник, составляющих этот класс методов приводятся цифровая обработка сигналов и обработка естественного языка (включая тональный анализ);

- машинное обучение, включая обучение с учителем и без учителя, а также Ensemble learning[en] — использование моделей, построенных на базе статистического анализа или машинного обучения для получения комплексных прогнозов на основе базовых моделей (англ. constituent models, ср. со статистическим ансамблем в статистической механике);

- искусственные нейронные сети, сетевой анализ, оптимизация, в том числе генетические алгоритмы;

- распознавание образов;

- прогнозная аналитика;

- имитационное моделирование;

- пространственный анализ (англ. Spatial analysis) — класс методов, использующих топологическую, геометрическую и географическую информацию в данных;

- статистический анализ, в качестве примеров методов приводятся A/B-тестирование и анализ временных рядов;

- визуализация аналитических данных — представление информации в виде рисунков, диаграмм, с использованием интерактивных возможностей и анимации как для получения результатов, так и для использования в качестве исходных данных для дальнейшего анализа.

**Правило Мура:**

В 1960-е годы, в самом начале информационной революции, Гордон Мур, впоследствии один из основателей корпорации Intel, обратил внимание на интересную закономерность в развитии компьютеров. Он заметил, что объем компьютерной памяти удваивается примерно каждые два года. Эта закономерность стала своего рода эмпирическим правилом в компьютерной промышленности, и вскоре оказалось, что не только память, но и каждый показатель производительности компьютера — размер микросхем, скорость процессора и т. д. — подчиняется этому правилу.

Мы прошли путь от компьютеров на ламповых транзисторах к компьютерам на интегральных схемах и далее — к компьютерам на микропроцессорах, и каждый раз закон Мура находил подтверждение.

Однако рано или поздно законы природы положат конец господству закона Мура. Взять, к примеру, размеры элементов микросхемы. Закон предсказывает, что к 2060 году они должны будут стать размером с одиночный атом — что невозможно с точки зрения квантовой механики!

**Изображение выглядит как текст, письмо, человек, снимок экрана

Автоматически созданное описание**

**Правила Амдала**

Зако́н Амдала (англ. Amdahl's law, иногда также Закон Амдаля — Уэра) — иллюстрирует ограничение роста производительности вычислительной системы с увеличением количества вычислителей. Джин Амдал сформулировал закон в 1967 году, обнаружив простое по существу, но непреодолимое по содержанию ограничение на рост производительности при распараллеливании вычислений: «В случае, когда задача разделяется на несколько частей, суммарное время её выполнения на параллельной системе не может быть меньше времени выполнения самого медленного фрагмента». Согласно этому закону, ускорение выполнения программы за счёт распараллеливания её инструкций на множестве вычислителей ограничено временем, необходимым для выполнения её последовательных инструкций.

Вообще, смысл данного  закона  весьма простой: если мы “улучшаем” (ускоряем) часть некой системы, то общий эффект на быстродействие всей системы будет зависеть от того, насколько этот компонент был важен (то есть какой процент времени работа с ним занимает) и насколько мы его улучшили.

Параллельное программирование изучает методы построения программ, которые будут выполняться на нескольких процессорах. В результате решения одной из первых задач этого раздела информатики появился закон Амдала.

Задача Амдала формулировалась так. Имеется n процессоров и p процентов вычислений не могут выполняться параллельно. Во сколько раз быстрее можно выполнить вычисления по сравнению с одним процессором?

Например, если n = 10, p = 50, а на одном процессоре все вычисления выполняются за время t. Тогда первая половина вычислений (50%) будет выполнена за время t/(2•10) , а вторая - за время t/2. Общее время вычислений в этом случае составит t/2 + t/20 = 11•t/20, а ускорение по сравнению с одним процессором составит 20/11 раза.

Если же n = 10, p = 25, и на одном процессоре все вычисления выполняются за время t. Тогда 75% вычислений будут выполнены за время 3•t/(4•10) , а оставшиеся 25% - за время t/4 . Общее время вычислений в этом случае составит t/4 + 3•t/40 = 13•t/40, а ускорение по сравнению с одним процессором составит 40/13 раза.

Распределенные системы

Распределенные вычисления – это способ объединения нескольких компьютеров для решения общей проблемы. Он превращает компьютерную сеть в мощный единый компьютер, который предоставляет широкомасштабные ресурсы для решения сложных задач.

Например, распределенные вычисления могут шифровать большие объемы данных; решать физические или химические уравнения со многими переменными и выводить высококачественную трехмерную видеоанимацию. Распределенные системы, распределенное программирование и распределенные алгоритмы – это еще несколько терминов, относящихся к распределенным вычислениям.

**Каковы преимущества распределенных вычислений?**

- Возможности масштабирования

- Доступность

- Согласованность

- Прозрачность

- Эффективность

**Какие существуют типы распределенной вычислительной архитектуры?**

Существует четыре основных типа распределенной архитектуры.

**Клиент-серверная архитектура**

Клиент-серверная архитектура – это самый распространенный способ организации распределенной системы. Компьютеры выполняют две роли: клиенты и серверы.

Клиенты

Количество информации и вычислительные возможности клиентов ограничены. При этом, они отправляют запросы серверам, которые управляют большей частью данных и других ресурсов. Вы можете отправлять запросы к клиентам, и те будут связываться с серверами от вашего имени.

Серверы

Серверы синхронизируют ресурсы и управляют доступом к ним. Они отвечают на запросы клиентов, отправляя данные или информацию о состоянии. Обычно один сервер может обрабатывать запросы от нескольких компьютеров.

Преимущества и ограничения

Преимуществами клиент-серверной архитектуры являются безопасность и простота текущего обслуживания. Вам нужно уделять основное внимание только защите серверов. Кроме того, любые изменения в системах баз данных требуют изменений только на сервере.

Ограничением клиент-серверной архитектуры является то, что серверы могут стать узким местом в связи, особенно когда несколько компьютеров отправляют запросы одновременно.

**Трехуровневая архитектура**

В трехуровневых распределенных системах клиентские компьютеры остаются на первом уровне доступа. С другой стороны, серверные компьютеры делятся еще на две категории.

Серверы приложений

Серверы приложений – это средний уровень связи. Они содержат логику приложений или основные функции, для выполнения которых предназначена распределенная система.

Серверы баз данных

Серверы баз данных – это третий уровень, на котором осуществляется хранение данных и управление ими. Они отвечают за получение данных и их целостность.

Благодаря разделению ответственности на сервере трехуровневая распределенная система сокращает количество узких мест в связи и повышает производительность распределенных вычислений.

**N-уровневая архитектура**

N-уровневые модели состоят из нескольких различных клиент-серверных систем, которые взаимодействуют между собой для решения одной и той же проблемы. Большинство современных распределенных систем используют n-уровневую архитектуру, в которой разные корпоративные приложения совместно работают за кулисами как одна система.

**Пиринговая архитектура**

Пиринговые распределенные системы возлагают на все компьютеры в сети одинаковые обязанности. Разделение на клиентские и серверные компьютеры отсутствует, и любой компьютер может выполнять все функции. Пиринговая архитектура стала популярной в сфере совместного использования контента, потоковой передачи файлов и сетях блокчейн.

**Как работают распределенные вычисления?**

В распределенных вычислениях компьютеры передают сообщения друг другу в архитектуре распределенных систем. Протоколы связи или правила создают зависимости между компонентами распределенной системы. Эта взаимосвязь называется взаимозависимостью. Существует два основных типа взаимозависимости.

Слабая взаимозависимость

При слабой взаимозависимости компоненты связаны слабо, и изменения в одном из них не влияют на другой. Например, между клиентом и сервером может быть слабая взаимосвязь по времени. Сообщения от клиента добавляются в очередь на сервере, и клиент может продолжать выполнять другие функции, пока сервер не ответит на его сообщение.

Сильная взаимозависимость

В высокопроизводительных распределенных системах часто используется сильная взаимозависимость. Быстрые локальные сети обычно охватывают несколько компьютеров, составляющих кластер. В кластерных вычислениях все компьютеры выполняют одну и ту же задачу. Центральные системы управления, называемые кластерным промежуточным программным обеспечением, управляют выполнением задач, планируют их и координируют связь между разными компьютерами.

**Что такое параллельные вычисления?**

Параллельные вычисления – это тип вычислений, в котором один или несколько компьютеров в сети проводят многие вычисления или выполняют многие процессы одновременно. Несмотря на то, что термины параллельные вычисления и распределенные вычисления часто используются как взаимозаменяемые, между ними есть некоторые различия.

Параллельные вычисления и распределенные вычисления

Параллельные вычисления – это форма распределенных вычислений с особо сильной взаимозависимостью. При параллельной обработке все процессоры имеют доступ к общей памяти для обмена информацией друг с другом. С другой стороны, при распределенных вычислениях каждый процессор пользуется собственной памятью (распределенной памятью). Процессоры используют для обмена информацией передачу сообщений.

# **Билет №8.**

Экосистема Big Data. Распределенная файловая система Hadoop (HDFS). Инструменты работы с большими данными: Hive и HiveQL.

1. **Экосистема Big Data.**

Экосистема Big Data представляет собой множество технологий, инструментов и платформ, разработанных для обработки, хранения, анализа и визуализации огромных объемов данных.

Big Data — это крупные массивы разнообразной информации и стек специальных технологий, инструментов и платформ, разработанных для её обработки, хранения, анализа и визуализации. Термин применяется к таким объемам данных, с которыми пользовательский компьютер и офисные программы не справятся.

Существует шесть основных критериев (6V), которые помогут определить Big Data:

* Volume (объем) — информации должно поступать более 150 Гб в сутки.
* Velocity (скорость) — для работы с массивами информации в режиме реального времени требуются повышенные вычислительные мощности.
* Variety (разнообразие) — поступающая информация имеет разные форматы или степень структурированности.
* Veracity (достоверность) — источникам данных можно доверять, а результат их обработки обладает достоверностью, достаточной для принятия решений.
* Variability (вариативность) — поток данных изменчив, на него может влиять даже время суток или погода.
* Value (ценность) — данные могут иметь разное значение для компании.

Возникла необходимость придумать новые типы хранилищ данных, поскольку стандартных уже не хватало. Первой платформой, которая взяла на себя работу с такими объемами данных, стала Hadoop. К настоящему времени она обладает мощным стеком инструментов. Вот некоторые из основных компонентов этой экосистемы:

* Apache Hadoop: Фреймворк для распределенного хранения и обработки структурированных и неструктурированных данных через кластеры компьютеров.
* Apache Spark: Система обработки данных в памяти, которая обеспечивает быструю обработку данных в пакетном и потоковом режимах.
* Apache Hive: Инфраструктура для анализа, запросов и обработки данных в распределенных хранилищах, таких как Hadoop.
* Apache HBase: Распределенная система управления базами данных NoSQL, обеспечивающая быстрый доступ к большим объемам данных.
* Apache Kafka: Платформа для обработки данных в реальном времени и передачи данных между приложениями.
* Apache Flink: Фреймворк для обработки данных в потоковом режиме с низкой задержкой, поддерживающий сложные вычисления.

Эти и другие инструменты объединены для создания мощной и гибкой инфраструктуры обработки и анализа больших данных.

1. **Распределенная файловая система Hadoop (HDFS).**

Распределенная файловая система Hadoop (Hadoop Distributed File System – HDFS) – это распределенная файловая система, предназначенная для хранения и обработки больших объемов данных на кластерах вычислительных узлов.

HDFS была разработана как часть проекта Apache Hadoop, который представляет собой платформу с открытым исходным кодом, написанную на Java и позволяющую распределять данные для анализа по кластеру компьютеров. *Кластер* — группа компьютеров, которые представляют собой аппаратный ресурс, выполняющий работу как единое целое. То есть задача одновременно выполняется на всех из них, что ускоряет её решение.

HDFS разделяет данные на множество блоков и хранит их на серверах в кластере. Блок — это кусочек файла стандартного размера. Во второй версии HDFS, которая сейчас считается основной, это 128 МБ. Соответственно, исходный файл разделяется на блоки по 128 МБ, которых может быть десять, а несколько тысяч. Каждый блок с информацией реплицируется на несколько узлов для обеспечения отказоустойчивости.

**Архитектура** HDFS включает в себя несколько ключевых компонентов, каждый из которых выполняет свою уникальную роль в распределенном хранении данных.

1. Клиент (HDFS Client). Клиенты представляют собой приложения, которые взаимодействуют с HDFS для чтения, записи и обработки данных. Они отправляют запросы на операции с файлами HDFS, такие как чтение, запись, создание или удаление файлов.
2. Главный узел (NameNode). Центральный компонент HDFS, отвечающий за управление метаданными файловой системы. Хранит информацию о том, где расположены блоки данных, и предоставляет эту информацию клиентам. NameNode не хранит фактические данные, а только метаданные, что делает его критически важным компонентом для целостности файловой системы. Его отказ может привести к недоступности файловой системы, поэтому он является одним из узких мест в архитектуре HDFS.
3. Вторичный главный узел (Secondary NameNode) Выполняет задачу резервного копирования метаданных от NameNode. Несмотря на название, Secondary NameNode не является резервной NameNode; он предназначен для создания периодических снимков метаданных и их объединения с текущим состоянием, чтобы уменьшить время восстановления в случае сбоя. Помогает предотвратить потерю данных и ускоряет процесс восстановления NameNode в случае сбоя.
4. Узел данных (DataNode). Datanodes – узлы, на которых хранятся фактические данные. Они отвечают за чтение и запись данных в файлы, а также передачу блоков данных NameNode для регистрации метаданных. HDFS использует репликацию для обеспечения отказоустойчивости данных, и DataNode отвечают за создание и управление репликами блоков данных.

Вместе эти компоненты обеспечивают эффективное и отказоустойчивое хранение данных в распределенной среде, что делает HDFS подходящей для обработки больших объемов данных на кластерах серверов.

Изображение выглядит как текст, диаграмма, снимок экрана, План

Автоматически созданное описание

**Преимущества**:

* Распределённое хранение. HDFS разбивает файлы на небольшие блоки и хранит их на разных узлах в кластере серверов. Это равномерно распределяет нагрузку на кластер и позволяет ускорить работу с данными за счёт одновременной обработки сотен и тысяч файловых блоков.
* Репликация данных. Каждый блок данных в HDFS дублируется на несколько узлов. Если один узел выходит из строя, информация может быть восстановлена из других.
* Работа в формате потока данных. Обработка данных может идти в режиме реального времени в процессе их получения, что ускоряет работу. Серверу не нужно ждать, пока поступление данных закончится.
* Простота обслуживания и устойчивость. Благодаря репликации и системе сообщений узлов данных HDFS автоматически обнаруживает сбои и восстанавливает данные из реплицированных узлов.
* Масштабируемость. HDFS легко масштабируется по горизонтали. Если объём данных или нагрузка увеличиваются, то можно просто добавить больше серверов в вычислительный кластер. Система будет автоматически использовать их для хранения и обработки данных.
* Поддержка различных типов данных. HDFS поддерживает хранение разнообразных данных — структурированных (таблицы), полуструктурированных (JSON, XML) и неструктурированных (видео и изображения).
* Интеграция с экосистемой Hadoop. HDFS плотно интегрирована с другими компонентами экосистемы Hadoop — Apache Spark, Apache Hive, Apache Pig и другими. Вместе они обеспечивают полный цикл обработки данных — хранение, распределение, загрузку, анализ, визуализацию и прочие способы представления данных.

**Недостатки**:

* Низкая эффективность работы с файлами меньше размера одного стандартного блока — 128 МБ. Работа с ними приведёт к замедлению работы из-за многократного повышения нагрузки на NameNode, хранящего пространство имён в HDFS.
* Работа системы полностью зависит от главного узла. Если по какой-либо причине он перестанет работать, то вся HDFS выйдет из строя. Восстановить его из вторичного главного узла невозможно.
* Низкая безопасность данных, так как при получении доступа к главному узлу можно получить доступ ко всей хранящейся в файловой системе информации.

1. **Инструменты работы с большими данными: Hive и HiveQL.**

Hive – это система управления базами данных в рамках платформы Hadoop с SQL-подобным языком запросов, позволяет выполнять запросы, агрегировать и анализировать данные; компонент экосистемы Hadoop. Работает напрямую с [HDFS](https://ru.wikipedia.org/wiki/HDFS), поддерживает основные форматы Hadoop. Запросы могут выполняться через [Tez](https://ru.wikipedia.org/wiki/Tez), [Spark](https://ru.wikipedia.org/wiki/Spark_(%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B0_%D1%80%D0%B0%D1%81%D0%BF%D1%80%D0%B5%D0%B4%D0%B5%D0%BB%D1%91%D0%BD%D0%BD%D0%BE%D0%B9_%D0%BE%D0%B1%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8)) или [Hadoop MapReduce](https://ru.wikipedia.org/wiki/Hadoop_MapReduce).

Язык запросов — HiveQL — приближен к SQL, при этом не реализует все возможности стандарта [SQL-92](https://ru.wikipedia.org/wiki/SQL-92). В язык встроены функции для работы с форматами [XML](https://ru.wikipedia.org/wiki/XML) и [JSON](https://ru.wikipedia.org/wiki/JSON), поддержка нескалярных типов данных, таких как массивы, структуры, ассоциативные массивы, реализован достаточно широкий набор [агрегатных функций](https://ru.wikipedia.org/w/index.php?title=%D0%90%D0%B3%D1%80%D0%B5%D0%B3%D0%B0%D1%82%D0%BD%D0%B0%D1%8F_%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F&action=edit&redlink=1), поддерживаются определяемые пользователем функции, блокировки.

*\*MapReduce* – это модель распределённых вычислений от компании Google, используемая в технологиях Big Data для параллельных вычислений над очень большими (до нескольких петабайт) наборами данных в компьютерных кластерах, и фреймворк для вычисления распределенных задач на узлах (node) кластера.

Ключевыми преимуществами Apache Hive являются следующие:

* масштабируемость — динамическое расширение при добавлении машины к кластеру Hadoop;
* расширяемость за счет MapReduce и определяемых пользователем функций (UDF/UDAF/UDTF);
* отказоустойчивость благодаря сохранению всех промежуточных результатов;
* поддержка разных форматов данных — TEXTFILE, Sequence, ORC, RCFILE, а также Parquet (с помощью плагина в версиях позже 0.10).

В Apache Hive запросы к данным, хранящимся в Hadoop, реализуются на SQL-подобном декларативном языке Hive Query Language (HiveQL), который является подмножеством SQL92. Однако, в ряде случаев HiveQL отличается от стандартного SQL, в частности:

* разные способы определения операций join для максимальной производительности;
* в HiveQL нет некоторых функций, операций и операторов SQL (UPDATE и DELETE statements, INSERT для отдельных строк);
* HiveQL позволяет вставлять пользовательский код для ситуаций, которые не вписываются в типовой SQL, предоставляя соответствующие инструменты для обработки входа и выхода – определенные пользователем функции: User Defined Function (UDF), User Defined Aggregate Function (UDAF), User Defined Tabular Function (UDTF);
* HiveQL не поддерживает типы данных даты и времени, т. к. они рассматриваются как строки.

Пример сеанса работы с Hive с применением HiveQL — удаление таблицы, создание таблицы, загрузка в неё данных из текстового файла и запрос для подсчёта, сколько раз каждое слово встречалось в файле:

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

# **Билет №9.**

Инструменты работы с большими данными: Spark и PySpark.

Работа с большими данными требует соответствующих инструментов — обычный компьютер просто не справится с обработкой терабайтов информации. Одним из таких инструментов является платформа Hadoop, которая обеспечивает распределённый анализ данных в кластерах из сотен и тысяч компьютеров.

Важная часть Hadoop — фреймворк Apache Spark. Он позволяет проводить обработку данных в режиме реального времени в отдельных кластерах, распределяя между ними задачи и обрабатывая результаты.

Фреймворк (англ. framework — «каркас, структура») — готовый набор инструментов, который помогает разработчику быстро создать продукт: сайт, приложение, интернет-магазин, CMS-систему.

Apache Spark — это фреймворк для обработки и анализа больших объёмов информации, входящий в инфраструктуру Hadoop. Он позволяет быстро выполнять операции с данными в вычислительных кластерах и поддерживает такие языки программирования, как Scala, Java, Python, R и SQL. Spark ускоряет обработку больших данных, распределяя вычисления между сотнями и тысячами машин, объединённых в кластеры. Это не новая технология — за несколько лет до появления Spark в Hadoop с аналогичной целью использовался MapReduce. Подход Spark выгодно отличается от классической парадигмы MapReduce и позволяет добиться большей скорости обработки данных. Это стало возможным благодаря ряду особенностей фреймворка.

**Обработка в памяти** (in-memory processing). Spark хранит и обрабатывает данные в оперативной памяти. Такой подход обеспечивает гораздо большую скорость, чем загрузка и чтение данных с диска, как в MapReduce.

**Ленивые вычисления.** Spark использует концепцию отложенного выполнения вычислений. Это означает, что операции над данными проводятся только перед непосредственным использованием результатов этих операций. Благодаря этому вычислительные мощности не тратятся на вычисления, которые понадобятся когда-то в будущем.

**Resilient distributed datasets (RDD).**Эта структура хранит датасеты и информацию о выполненных преобразованиях сразу на нескольких узлах кластерной сети. RDD позволяет Spark восстановить данные в случае сбоя и оптимизировать вычисления.

**Параллельная обработка и комбинирование операций.**Spark распределяет данные и вычисления по нескольким узлам в кластере, выполняя разные операции обработки параллельно в режиме реального времени. Это отличает его от MapReduce, при использовании которого каждый следующий этап работы с датасетом требует завершения предыдущего.

Благодаря этим особенностям Spark [имеет](https://www.databricks.com/blog/2014/11/05/spark-officially-sets-a-new-record-in-large-scale-sorting.html) в десятки раз большую скорость работы с данными, чем MapReduce.

Apache Spark состоит из нескольких модулей:

* Spark Core;
* Spark SQL;
* Spark Streaming;
* MLlib;
* GraphX.

Также фреймворк предоставляет API для работы с языками Python, Java и R. Таким образом, Spark поддерживает четыре языка программирования, включая родной Scala, на котором он написан, и SQL-подобный язык запросов.

## **Из чего состоит Spark-приложение**

В основе любого Spark-приложения лежат две сущности:

* управляющий процесс — **драйвер** (driver);
* набор распределённых рабочих процессов — **исполнителей** (executors).

**Драйвер** необходим для запуска заданий на конкретном узле кластера и управления их выполнением. Кроме того, он контролирует работу распределённых узлов и отвечает за их ответы на ввод со стороны пользователя.

**Исполнители** — это распределённые процессы, отвечающие за непосредственное выполнение задач и их хранение в памяти.

Рабочий процесс в Apache Spark устроен просто:

* У любой распределённой системы есть менеджер кластеров (cluster manager), контролирующий распределение ресурсов между исполнителями.
* При запуске Spark-приложения драйвер запрашивает у менеджера кластеров ресурсы для запуска исполнителей.
* Если ресурсы есть, менеджер кластеров запускает исполнители.
* Драйвер отправляет доступным исполнителям свои задачи.
* Исполнители проводят необходимые вычисления и отправляют результат драйверу.
* После завершения всех вычислений исполнители закрываются, их ресурсы возвращаются в общий кластер. Теперь распределённая система может работать над другими задачами.

Spark применяют в работе с большими данными и машинным обучением специалисты разного профиля: аналитики данных, дата-сайентисты и дата-инженеры. Области использования разнообразны. Приведём несколько примеров.

**В онлайн-торговле** Spark используют для обработки информации о заказах и действиях пользователей в режиме реального времени. Например, такие данные можно кластеризовать и фильтровать, создавая персонализированные рекомендации для пользователей в момент их нахождения в интернет-магазине.

**В финансовой сфере** — Spark может использоваться для обеспечения безопасности клиентов: обнаружения подозрительных операций с помощью анализа большого объёма данных о местоположении и IP-адресе устройства, суммах средств и так далее.

**В здравоохранении** — для прогнозирования и составления рекомендаций по обследованию и лечению пациентов. Работа со Spark позволяет учитывать при их составлении большой объём данных — от результатов последних анализов человека до возможностей оказания медицинской помощи в конкретном лечебном учреждении.

RDD- Набор [RDD](https://spark-school.ru/wiki/rdd-wiki/) (Resilient Distributed Dataset) – это неизменяемая коллекция объектов данных. Каждый такой набор делится на определенное количество частей, которые обрабатываются различными узлами в кластере. Распределенные наборы данных ([RDD](https://spark-school.ru/wiki/rdd-wiki/)) можно создавать двумя способами: загружая внешние наборы данных (из существующего файла) или распределяя большие множества данных внутри программы-драйвера. Существует масса методов для работы с распределенными наборами данных ([RDD](https://spark-school.ru/wiki/rdd-wiki/)). Среди них известны такие методы, как фильтрация, удаление дубликатов, случайная выборка элементов, применение функций к каждому элементу в [RDD](https://spark-school.ru/wiki/rdd-wiki/) и т.д.

Прежде всего, определим, по каким параметрам мы будем сравнивать DataFrame, DataSet и RDD. Для этого выделим несколько точек зрения:

* **данные** – представления, форматы, схемы;
* **вычисления** – сериализация, отложенные (ленивые) вычисления, операции агрегирования, использование памяти**;**
* **разработка** – поддерживаемые языки программирования, безопасность типов при компиляции, оптимизация, сборка мусора (Garbage Collection), неизменность (Immutability) и совместимость (Interoperability), область применения.

Перед тем, как приступить к сравнению RDD, DataFrame и DataSet по вышеперечисленным критериям, отметим разный порядок появления этих структур данных в Apache [Spark](https://bigdataschool.ru/wiki/spark) **[1]**:

* **RDD** появился с первого релиза фреймворка (Spark 0);
* **Датафрейм** впервые был представлен в Spark 1.3;
* **Датасет** стал доступен в версии 1.6.

**RDD**– это распределенная коллекция данных, расположенных по нескольким узлам кластера, набор объектов Java или Scala, представляющих данные. RDD работает со структурированными и с неструктурированные данными. Также, как DataFrame и DataSet, RDD не выводит схему загруженных данных и требует от пользователя ее указания.

**DataFrame**– это распределенная коллекция данных в виде именованных столбцов, аналогично таблице в реляционной базе данных. DataFrame работает только со структурированными и полуструктурированными данными, организуя информацию по столбцам, как в реляционных таблицах. Это позволяет Spark управлять схемой данных.

**DataSet** – это расширение API DataFrame, обеспечивающее функциональность объектно-ориентированного RDD-API (строгая типизация, лямбда-функции), производительность оптимизатора запросов Catalyst и механизм хранения вне кучи. DataSet эффективно обрабатывает структурированные и неструктурированные данные, представляя их в виде строки JVM-объектов или коллекции. Для представления табличных форм используется кодировщик (encoder).

Что это: Apache Spark — это фреймворк общего назначения для параллельной обработки данных, разработанный для эффективной работы с большими объемами данных в распределенной среде.

Особенности:

Поддерживает обработку различных типов данных, включая структурированные, полуструктурированные и неструктурированные данные.

Обладает высокой скоростью обработки благодаря возможности ин-памяти обработки данных.

Предоставляет богатый набор инструментов для обработки данных, включая SQL-запросы, машинное обучение, графовые вычисления и потоковую обработку данных.

PySpark:

Что это: PySpark — это API для работы с Apache Spark, предоставляющее возможность использования языка программирования Python для разработки приложений на базе Spark.

Особенности:

Обеспечивает удобный способ интеграции мощностей Spark с языком Python.

Позволяет использовать популярные библиотеки Python, такие как Pandas, NumPy и Matplotlib, в среде распределенной обработки данных Apache Spark.

**Apache Spark:**

**Общий принцип работы**: Apache Spark представляет собой фреймворк обработки данных, который работает на кластерах вычислительных узлов. Он основан на концепции распределенных вычислений в памяти, что позволяет ему достигать высокой производительности при обработке больших объемов данных.

* **Языковая поддержка**: Spark поддерживает несколько языков программирования, включая Scala, Java, Python и R. Это делает его доступным для широкого круга разработчиков и аналитиков данных.
* **Ключевые компоненты**:
  + **Spark Core**: Основной компонент, который предоставляет API для распределенных вычислений и управления кластером.
  + **Spark SQL**: Модуль для обработки структурированных данных с помощью SQL-подобных запросов.
  + **Spark Streaming**: Модуль для обработки потоковых данных в реальном времени.
  + **Spark** **MLlib**: Библиотека машинного обучения для работы с данными, включающая в себя алгоритмы классификации, регрессии, кластеризации и др.
  + **Spark GraphX**: Модуль для анализа графов и выполнения операций с графами.
* **Преимущества**:
  + **Распределенные вычисления в памяти**: Одним из ключевых преимуществ Apache Spark является его способность к выполнению распределенных вычислений в памяти. Это позволяет Spark обрабатывать данные быстрее, чем традиционные фреймворки для обработки данных, такие как Hadoop MapReduce, которые часто используют дисковое хранилище для промежуточных результатов.
  + **Высокоуровневые API**: Spark предоставляет различные высокоуровневые API для работы с данными, что упрощает создание и выполнение сложных вычислительных задач. Например, API для работы с SQL (Spark SQL) позволяет выполнять SQL-подобные запросы к структурированным данным, а API для машинного обучения (MLlib) обеспечивает доступ к алгоритмам машинного обучения для анализа данных.
  + **Масштабируемость**: Spark обладает высокой масштабируемостью, что позволяет ему эффективно работать с данными как на небольших кластерах, так и на крупных кластерах с десятками и сотнями узлов. Это делает Spark идеальным выбором для обработки данных в различных сценариях, включая анализ больших объемов данных и обработку данных в реальном времени.

**PySpark:**

* **PySpark** - это API для работы с Apache Spark, предоставляющее возможность использовать функционал Spark в Python-среде.
* **Интеграция с Python**: PySpark обеспечивает интеграцию Spark с языком программирования Python, что делает его удобным для Python-разработчиков и аналитиков данных.
* **Высокоуровневый интерфейс**: PySpark предоставляет удобный высокоуровневый интерфейс для работы с данными, включая функции для чтения, записи и обработки данных, аналогичные тем, что предоставляются в Spark API для Scala и Java.
* **Мощные возможности Spark**: Пользователи PySpark имеют доступ ко всем функциям и возможностям Apache Spark, включая распределенные вычисления в памяти, обработку потоковых данных, машинное обучение и анализ графов.
* **Популярность: PySpark** стал широко используемым инструментом в области анализа данных благодаря своей простоте использования и интеграции с экосистемой Python.
* **Использование Apache Spark и PySpark** позволяет разработчикам и аналитикам данных эффективно работать с большими объемами информации, обеспечивая высокую производительность, масштабируемость и удобство использования.

Хотя Apache Spark и PySpark тесно связаны, поскольку PySpark является API для работы с Apache Spark из Python, есть несколько ключевых различий между ними:

* **Языковая поддержка**:
  + **Apache Spark**: Оригинальный Apache Spark предоставляет API для работы на нескольких языках программирования, включая Scala, Java, Python и R.
  + **PySpark:** PySpark, с другой стороны, является API, специально созданным для работы с Apache Spark из языка программирования Python. Это означает, что PySpark предоставляет интерфейс для использования Spark с использованием Python.
* **Интеграция и удобство использования**:
  + **Apache Spark**: Apache Spark может использоваться в различных средах разработки и интегрироваться с инструментами, предназначенными для Scala, Java и других языков программирования. Однако использование Spark API на Scala или Java может требовать от разработчиков дополнительного времени и усилий для написания кода.
  + **PySpark**: PySpark предоставляет более естественный и удобный интерфейс для использования Spark в среде Python. Это позволяет Python-разработчикам использовать знания и навыки Python для работы с данными, что может быть более привлекательным и эффективным для них.
* **Экосистема и инструменты**:
  + **Apache Spark**: Apache Spark имеет развитую экосистему инструментов и библиотек для анализа данных, машинного обучения, обработки потоков данных и других задач. Она доступна для всех языков API, поддерживаемых Spark.
  + **PySpark**: PySpark взаимодействует с экосистемой инструментов Python, включая такие популярные библиотеки, как Pandas, NumPy, SciPy и Matplotlib. Это обеспечивает большую гибкость и удобство для аналитиков данных и разработчиков, работающих в среде Python.
* **Производительность**: Оба варианта предоставляют доступ к тем же функциям Apache Spark и используют те же механизмы выполнения задач. Поэтому производительность Spark кода в PySpark идентична производительности кода в Scala или Java.

# Билет №10.

Основы реляционных баз данных (БД): тип данных, домен, атрибут, кортеж, отношение. Фундаментальные свойства отношений. Целостность данных. Основные понятия реляционной алгебры.

База данных — набор информации, который организуют и хранят по определенным правилам. Ее основная особенность — наличие структуры. То есть это не хаотичный список, а организованный.

Базы данных разделяют на два основных типа: реляционные и нереляционные. Последние делятся ещё на два: сетевые и иерархические. Получается, существует три главных типа баз данных — реляционные, сетевые и иерархические.

Реляционная база данных (РБД) — это тип базы данных, основанный на реляционной модели данных. Реляционная модель данных — способ организации данных с использованием таблиц. Таблицы формируют отношения между данными, и в контексте РБД термины «таблицы» и «отношения» используются как синонимы. В реляционной модели с помощью таблиц данные представляются в удобной и структурированной форме для хранения, доступа и управления. Каждая таблица — отдельный объект, каждая строка в таблице (или кортеж) соответствует конкретной записи или экземпляру этого типа, а столбцы (или атрибуты) определяют характеристики этой сущности.

NoSQL (not only SQL, «не только SQL») — это термин, обозначающий технологии управления данными, отличных от SQL. К NoSQL относятся столбцовые, графовые, документоориентированные системы, а также системы по модели «ключ — значение».

Реляционная модель данных (РМД) была разработана сотрудником IBM Э.Ф. Коддом (E.F. Codd) в 1969-70 г.г. на основе математической теории отношений. Реляционные базы данных (БД) используют структурированную модель данных, основанную на таблицах (или "отношениях"), которые состоят из строк и столбцов. Вот основные компоненты и концепции реляционных баз данных:

Изображение выглядит как текст, диаграмма, План, схематичный

Автоматически созданное описание

**Основные Компоненты**

1. Тип данных — определяет характер и формат данных, которые могут храниться в столбце. Это необходимо для того, чтобы система могла правильно обрабатывать данные при выполнении запросов. Примеры включают `INT` (целые числа), `VARCHAR` (строковые значения переменной длины), `DATE` (дата), `FLOAT` для чисел с плавающей точкой.

2. Домен — набор допустимых значений для поля (атрибута). Он служит дополнительным ограничением на данные в столбце, помимо типа данных. Например, если атрибут — это возраст человека, домен может быть задан как диапазон от 0 до 120. Это ограничение помогает поддерживать логическую корректность данных.

Простой домен: ГОД = {1985, 2003, 2000}; ДЕНЬГИ = {500, 1000, 850}

Составной домен: ИСТОРИЯ ЗАРПЛАТЫ = {{, }, {}, {, , }}

3. Атрибут — это столбцы в таблице БД. Каждый атрибут имеет уникальное имя в рамках таблицы и заданный тип данных. Атрибуты определяют структуру данных, которая будет храниться в таблице. Например, таблица `Пользователи` может иметь атрибуты `Имя`, `Email`, `Возраст`. Атрибут, по которому можно однозначно определить каждый объект и другие данные о нем, называют ключевым полем или индификатором объекта

4. Кортеж — это одна строка в таблице, содержащая уникальный набор данных. Каждый кортеж состоит из значений, соответствующих атрибутам таблицы. Все кортежи в таблице должны иметь одинаковое количество и тип атрибутов, но значения данных в каждом кортеже могут быть разными.

5. Отношение — это совокупность кортежей с одинаковыми атрибутами. В контексте реляционных баз данных отношение часто синонимично термину "таблица". Отношения используются для организации данных в формате, удобном для манипулирования и запроса. Отношения могут быть связаны друг с другом через внешние ключи, что позволяет образовывать сложные структуры данных.

**Фундаментальные свойства отношений**

*Отношения в реляционных БД характеризуются следующими свойствами:*

- Структурная целостность: все кортежи уникальны; нет дублирующихся строк.

- Целостность порядка: порядок строк и столбцов не имеет значения.

- Целостность типа данных: каждый столбец должен содержать данные одного и того же типа, соответствующего его декларации.

*Свойства реляционной модели данных*

1. Каждый атрибут отношения имеет уникальное в данном отношении имя

2. Каждый атрибут определен на каком-то одном домене.

3. На одном и том же домене может быть определено несколько атрибутов.

4. Имя атрибута может совпадать с именем домена.

5. Порядок следования атрибутов не устанавливается.

6. В отношении нет совпадающих кортежей (каждый кортеж уникален).

7. Порядок следования кортежей не устанавливается.

8. Отношение имеет имя, которое в схеме базы данных отличается от имен всех других отношений.

Отношение – это множество кортежей, соответствующих одной схеме отношений.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

**Целостность данных**

Целостность данных – устойчивость хранимых данных к разрушению (уничтожению), связанному с неисправностями технических средств, системными ошибками и ошибочными действиями пользователей

Целостность данных в реляционных БД обеспечивается через различные ограничения:

- Ограничения первичного ключа: каждая таблица должна иметь первичный ключ, уникально идентифицирующий каждый кортеж.

- Ограничения внешнего ключа: обеспечивают соблюдение ссылочной целостности между таблицами.

- Проверочные ограничения (CHECK constraints): обеспечивают, что данные в столбце удовлетворяют определенным условиям.

**Основные понятия реляционной алгебры**

Реляционная алгебра представляет собой формальную систему, которая лежит в основе многих запросов к реляционным базам данных и является теоретической основой для SQL. Основные операции реляционной алгебры позволяют манипулировать наборами данных (отношениями), выполняя различные типы запросов для извлечения или изменения данных. Вот подробный обзор ключевых операций реляционной алгебры:

Изображение выглядит как текст, снимок экрана, Шрифт, документ

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт, документ

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт, документ

Автоматически созданное описание

**Операции реляционной алгебры**

Основные восемь операций реляционной алгебры были предложены [Э.Коддом](http://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4%D0%B4,_%D0%AD%D0%B4%D0%B3%D0%B0%D1%80).

* Объединение
* Пересечение
* Вычитание
* Декартово произведение
* Выборка
* Проекция
* Соединение
* Деление

Первая половина операций аналогична таким же операциям над множествами. Часть операций можно выразить через другие операции. Рассмотрим большую часть операций с примерами.  
  
Для понимания важно запомнить, что результатом любой операции алгебры над отношениями является еще одно отношение, которое можно потом так же использовать в других операциях.  
Создадим еще одну таблицу, которая нам пригодится в примерах.  
  
таблица SELLERS

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Условимся, что в этой таблице ID это внешний ключ, связанный с первичным ключом таблицы PRODUCTS.  
  
Для начала рассмотрим самую простую операцию — имя отношения. Её результатом будет такое же отношение, то есть выполнив операцию PRODUCTS, мы получим копию отношения PRODUCTS.

**Проекция**

Проекция является операцией, при которой из отношения выделяются атрибуты только из указанных доменов, то есть из таблицы выбираются только нужные столбцы, при этом, если получится несколько одинаковых кортежей, то в результирующем отношении остается только по одному экземпляру подобного кортежа.  
Для примера сделаем проекцию на таблице PRODUCTS выбрав из нее ID и PRICE.  
  
Синтаксис операции:  
π(ID, PRICE)PRODUCTS  
  
В результате этой операции получим отношение:

Изображение выглядит как текст, снимок экрана, линия, чек

Автоматически созданное описание

**Выборка**

Выборка — это операция, которая выделяет множество строк в таблице, удовлетворяющих заданным условиям. Условием может быть любое логическое выражение.  
Для примера сделаем выборку из таблицы с ценой больше 90.  
  
Синтаксис операции:  
σ(PRICE>90)PRODUCTS

В условии выборки мы можем использовать любое логическое выражение. Сделаем еще одну выборку с ценой больше 90 и ID товара меньше 300:  
  
σ(PRICE>90 ^ ID<300)PRODUCTS

Совместим операторы проекции и выборки. Мы можем это сделать, потому что любой из операторов в результате возвращает отношение и в качестве аргументов использует также отношение.  
Из таблицы с продуктами выберем все компании, продающие продуты дешевле 110.  
  
πCOMPANYσ(PRICE<100 )PRODUCTS

**Умножение**

Умножение или декартово произведение является операцией, производимой над двумя отношениями, в результате которой мы получаем отношение со всеми доменами из двух начальных отношений. Кортежи в этих доменах будут представлять из себя все возможные сочетания кортежей из начальных отношений. На примере будет понятнее.  
  
Получим декартово произведения таблиц PRODUCTS и SELLERS.  
Синтаксис операции:  
  
PRODUCTS × SELLERS  
Можно заметить, что у двух этих таблиц есть одинаковый домен ID. В подобной ситуации домены с одинаковыми названиями получают префикс в виде названия соответствующего отношения, как показано ниже.  
Для краткости перемножим не полные отношения, а выборки с условием ID<235

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Для примера использования этой операции представим себе необходимость выбрать продавцов с ценами меньше 90. Без произведения необходимо было бы сначала получить ID продуктов из первой таблицы, потом по этим ID из второй таблицы получить нужные имена SELLER, а с использованием произведения будет такой запрос:  
  
π(SELLER)σ(RODUCTS.ID=SELLERS.ID ^ PRICE<90)PRODUCTS × SELLERS  
  
**Соединение и естественное соединение**

Операция соединения обратна операции проекции и создает новое отношение из двух уже существующих. Новое отношение получается конкатенацией кортежей первого и второго отношений, при этом конкатенации подвергаются отношения, в которых совпадают значения заданных атрибутов. В частности, если соединить отношения PRODUCTS и SELLERS, этими атрибутами будут атрибуты доменов ID.  
  
Также для понятности можно представить соеднинение как результат двух операций. Сначала берется произведение исходных таблиц, а потом из полученного отношения мы делаем выборку с условием равенства атрибутов из одинаковых доменов. В данном случае условием явлется равенство PRODUCTS.ID и SELLERS.ID.  
  
Попробуем соединить отношения PRODUCTS и SELLERS и получим отношение.

**Пересечение и вычитание.**

Результатом операции пересечения будет отношение, состоящее из кортежей, полностью входящих в состав обоих отношений.  
Результатом вычитания будет отношение, состоящее из кортежей, которые являются кортежами первого отношения и не являются кортежами второго отношения.  
Данные операции аналогичны таким же операциям над множествам, так что, я думаю, нет необходимости подробно их расписывать.

Практическое применение

Реляционная алгебра служит теоретическим фундаментом для SQL, языка запросов, который используется в большинстве современных реляционных систем управления базами данных. Понимание реляционной алгебры помогает разрабатывать более эффективные и оптимальные SQL-запросы, а также лучше понимать, как системы БД обрабатывают данные.

Такие понятия, как нормализация, целостность данных, транзакции и оптимизация запросов, тесно связаны с операциями реляционной алгебры и играют ключевую роль в проектировании и использовании реляционных баз данных.

# Билет №11.

Этапы разработки БД. ER-диаграммы. Нормализация отношений

База данных — набор информации, который организуют и хранят по определенным правилам. Ее основная особенность — наличие структуры. То есть это не хаотичный список, а организованный.

База данных — это форма организации информации, при которой информация хранится в упорядоченном, структурированном виде в специальных электронных таблицах, связанных между собой.

Разработка базы данных – это критически важный аспект создания программного продукта, который требует глубокого технического понимания и стратегического планирования.

База данных – это, фактически, модель предметной области (ПрО). Значит, для создания БД надо сначала проанализировать ПрО и создать её модель (это называется инфологическим проектированием).

Процесс проектирования включает в себя следующие шаги:

1. Определение задач, стоящих перед базой данных.

2. Сбор и анализ документов, относящихся к исследуемой предметной области.

3. Описание особенностей ПрО, которые позволяют установить зависимости и связи между объектами (субъектами) предметной области.

4. Создание модели предметной области.

5. Определение групп пользователей и перечня задач, стоящих перед каждой группой.

6. Выбор аппаратной и программной платформы для реализации БД.

7. Выбор СУБД (системы управления базой данных).

8. Создание логической схемы БД.

9. Создание схем отношений, определение типов данных атрибутов и ограничений целостности.

10.Нормализация отношений (до третьей или четвёртой нормальной формы).

11.Определение прав доступа пользователей к объектам БД.

12.Написание текста создания основных объектов базы данных на языке SQL в синтаксисе выбранной СУБД (пользователи, таблицы и др.).

13.Написание текста создания вспомогательных объектов базы данных (представления, индексы, триггеры, роли и т.д.).

Эти шаги можно объединить с 5 этапов:

1. Инфологическое проектирование (1-5).

2. Определение требований к операционной обстановке, в которой будет функционировать информационная система (6).

3. Выбор системы управления базой данных (СУБД) и других инструментальных программных средств (7).

4. Логическое проектирование БД (8-11).

5. Физическое проектирование БД (12-13).

1. Инфологическое проектирование

Основными задачами этапа инфологического проектирования являются определение предметной области системы и формирование взгляда на неё с позиций сообщества будущих пользователей БД, т.е. информационно-логической модели ПрО.

Основными подходами к созданию инфологической модели предметной области являются [1]: 1. Функциональный подход к проектированию БД ("от задач"). 2. Предметный подход к проектированию БД ("от предметной области"). 3. Метод "сущность-связь" (entity–relation, ER–method).

2. Определение требований к операционной обстановке. На этом этапе производится оценка требований к вычислительным ресурсам, необходимым для функционирования системы, определение типа и конфигурации конкретной ЭВМ, выбор типа и версии операционной системы. Объём вычислительных ресурсов зависит от предполагаемого объёма проектируемой базы данных и от интенсивности их использования. Если БД будет работать в многопользовательском режиме, то требуется подключение её к сети и наличие соответствующей многозадачной операционной системы

3.выбор субд: Выбор СУБД осуществляется на основании таких критериев, как тип модели данных и её адекватность потребностям рассматриваемой ПрО; характеристики производительности; набор функциональных возможностей; удобство – 8 – и надежность СУБД в эксплуатации; стоимость СУБД и дополнительного программного обеспечения

4.логическое проектирование: На этапе логического проектирования разрабатывается логическая (концептуальная) структура БД. Для реляционной модели существуют формальные правила, которые позволяют преобразовать инфологическую модель ПрО в виде ER-диаграммы в логическую схему базы данных. Кроме получения схемы БД в целом на этом этапе выполняют создание схем отношений и их нормализацию. Этот этап более подробно рассмотрен в п.2 "Пример проектирования реляционной базы данных"

5.физическое: Этап физического проектирования заключается в определении схемы хранения, т.е. физической структуры БД. Схема хранения зависит от той физической структуры, которую поддерживает выбранная СУБД. Физическая структура БД, с одной стороны, должна адекватно отражать логическую структуру БД, а с другой стороны, должна обеспечивать эффективное размещение данных и быстрый доступ к ним. Результаты этого этапа документируются в форме схемы хранения на языке определения данных (DDL, Data Definition Language) выбранной СУБД. Принятые на этом этапе решения оказывают огромное влияние на производительность системы. Одной из важнейших составляющих проекта базы данных является разработка средств защиты БД. Защита данных имеет два аспекта: защита от сбоев и защита от несанкционированного доступа. Для защиты от сбоев на этапе физического проектирования разрабатывается стратегия резервного копирования. Для защиты от несанкционированного доступа каждому пользователю доступ к данным предоставляется только в соответствии с его правами доступа, набор которых также является составной частью проекта БД.

ER-диаграмма.

В реальном проектировании структуры базы данных применяются другой метод - так называемое, ***семантическое моделирование***. Семантическое моделирование представляет собой моделирование структуры данных, опираясь на смысл этих данных. В качестве инструмента семантического моделирования используются различные варианты ***диаграмм сущность-связь*** (***ER - Entity-Relationship***).

Первый вариант модели сущность-связь был предложен в 1976 г. Питером Пин-Шэн Ченом [37]. В дальнейшем многими авторами были разработаны свои варианты подобных моделей (нотация Мартина, нотация IDEF1X, нотация Баркера и др.). Кроме того, различные программные средства, реализующие одну и ту же нотацию, могут отличаться своими возможностями. По сути, все варианты диаграмм сущность-связь исходят из одной идеи - рисунок всегда нагляднее текстового описания. Все такие диаграммы используют графическое изображение сущностей предметной области, их свойств (атрибутов), и взаимосвязей между сущностями.

Мы опишем работу с ER-диаграммами близко к нотации Баркера, как довольно легкой в понимании основных идей. Данная глава является скорее иллюстрацией методов семантического моделирования, чем полноценным введением в эту область.

#### Основные понятия ER-диаграмм

*Определение 1*. ***Сущность*** - это класс однотипных объектов, информация о которых должна быть учтена в модели.

Каждая сущность должна иметь наименование, выраженное существительным в единственном числе.

Примерами сущностей могут быть такие классы объектов как "Поставщик", "Сотрудник", "Накладная".

Каждая сущность в модели изображается в виде прямоугольника с наименованием:

Изображение выглядит как текст, Шрифт, снимок экрана, линия

Автоматически созданное описание

**Рис. 1**

*Определение 2*. ***Экземпляр сущности*** - это конкретный представитель данной сущности.

Например, представителем сущности "Сотрудник" может быть "Сотрудник Иванов".

Экземпляры сущностей должны быть *различимы*, т.е. сущности должны иметь некоторые свойства, уникальные для каждого экземпляра этой сущности.

*Определение 3*. ***Атрибут сущности*** - это именованная характеристика, являющаяся некоторым свойством сущности.

Наименование атрибута должно быть выражено существительным в единственном числе (возможно, с характеризующими прилагательными).

Примерами атрибутов сущности "Сотрудник" могут быть такие атрибуты как "Табельный номер", "Фамилия", "Имя", "Отчество", "Должность", "Зарплата" и т.п.

Атрибуты изображаются в пределах прямоугольника, определяющего сущность:

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

**Рис. 2**

*Определение 4*. ***Ключ сущности*** - это *неизбыточный* набор атрибутов, значения которых в совокупности являются *уникальными* для каждого экземпляра сущности. Неизбыточность заключается в том, что удаление любого атрибута из ключа нарушается его уникальность.

Сущность может иметь несколько различных ключей.

Ключевые атрибуты изображаются на диаграмме подчеркиванием:

Изображение выглядит как текст, Шрифт, снимок экрана, линия

Автоматически созданное описание

**Рис. 3**

*Определение 5*. ***Связь*** - это некоторая ассоциация между *двумя* сущностями. Одна сущность может быть связана с другой сущностью или сама с собою.

Связи позволяют по одной сущности находить другие сущности, связанные с нею.

Например, связи между сущностями могут выражаться следующими фразами - "СОТРУДНИК может иметь несколько ДЕТЕЙ", "каждый СОТРУДНИК обязан числиться ровно в одном ОТДЕЛЕ".

Графически связь изображается линией, соединяющей две сущности:

Изображение выглядит как текст, Шрифт, снимок экрана, линия

Автоматически созданное описание

**Рис. 4**

Каждая связь имеет два конца и одно или два наименования. Наименование обычно выражается в неопределенной глагольной форме: "иметь", "принадлежать" и т.п. Каждое из наименований относится к своему концу связи. Иногда наименования не пишутся ввиду их очевидности.

Каждая связь может иметь один из следующих ***типов связи***:

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

**Рис. 5**

Связь типа ***один-к-одному*** означает, что один экземпляр первой сущности (левой) связан с одним экземпляром второй сущности (правой). Связь один-к-одному чаще всего свидетельствует о том, что на самом деле мы имеем всего одну сущность, неправильно разделенную на две.

Связь типа ***один-ко-многим*** означает, что один экземпляр первой сущности (левой) связан с несколькими экземплярами второй сущности (правой). Это наиболее часто используемый тип связи. Левая сущность (со стороны "один") называется ***родительской***, правая (со стороны "много") - ***дочерней***. Характерный пример такой связи приведен на Рис. 4.

Связь типа ***много-ко-многим*** означает, что каждый экземпляр первой сущности может быть связан с несколькими экземплярами второй сущности, и каждый экземпляр второй сущности может быть связан с несколькими экземплярами первой сущности. 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.

Каждая связь может иметь одну из двух ***модальностей связи***:

![Изображение выглядит как текст, Шрифт, снимок экрана, линия

Автоматически созданное описание](data:image/gif;base64,R0lGODlhawBMAJEAAP///wAAAAAA/wAAACH5BAAAAAAALAAAAABrAEwAAAL/hI+py+0Po5y02ouz3rz7D4biSJbmiabqyrbuC8fyTNf2jef6zvf+vxMIhwIGcWg8FhfKpaKZPEaB1Kr1is0mAguuVhTwHsJfsNhALoO4YvY4DUd7w3P6GetGA+Zj/d7f1qfGx/fn95dnqHiXRQgoKHdmFzioV0iJuNilpvnoeQnJiBf6VthJhyDKucra6voKGys7S1tre4ubq7vL28vTBBwsPExcbBzsm6y8DDupqizJrBAtnWodmYoat1eH+bxD7f3paYhJFU6qmUhpDoR+aMotaReZdr4FLxi4Pg3pjt+p3LhT/nw4QyVPTh9xCfPd61dN1LdqFCtavIgxo8aNBxw7evy4oQAAOw==)

**Рис. 6**

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.

Модальность "***должен***" означает, что экземпляр одной сущности *обязан быть связан не менее чем с одним* экземпляром другой сущности.

Связь может иметь *разную модальность* с разных концов (как на Рис. 4).

Описанный графический синтаксис позволяет *однозначно* читать диаграммы, пользуясь следующей схемой построения фраз:

<Каждый экземпляр СУЩНОСТИ 1> <МОДАЛЬНОСТЬ СВЯЗИ> <НАИМЕНОВАНИЕ СВЯЗИ> <ТИП СВЯЗИ> <экземпляр СУЩНОСТИ 2>.

Каждая связь может быть прочитана как слева направо, так и справа налево. Связь на Рис. 4 читается так:

Слева направо: "каждый сотрудник может иметь несколько детей".

Справа налево: "Каждый ребенок обязан принадлежать ровно одному сотруднику".

Разработанный выше пример ER-диаграммы является примером концептуальной диаграммы. Это означает, что диаграмма не учитывает особенности конкретной СУБД. По данной концептуальной диаграмме можно построить физическую диаграмму, которая уже будут учитываться такие особенности СУБД, как допустимые типы и наименования полей и таблиц, ограничения целостности и т.п. Физический вариант диаграммы, приведенной на Рис. 9 может выглядеть, например, следующим образом:

Нормализация:

Нормализация – процесс позволяющий распределить атрибуты по таблицам, исключая их избыточность. Подход с нормальными формами был разработан Эдгаром Коддом в 1970-х гг. Существуют следующие стандартные нормальные формы: θ 1-я нормальная форма θ 2-я нормальная форма θ 3-я нормальная форма θ нормальная форма Бойса-Кодда θ 4-я нормальная форма θ 5-я нормальная форма Нормализация желательна, однако нужно понимать, чем выше степень нормализации, тем больше получается таблиц, что усложняет схему и снижает производительность БД

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Этапы разработки базы данных

При разработке базы данных обычно выделяется несколько уровней моделирования, при помощи которых происходит переход от предметной области к конкретной реализации базы данных средствами конкретной СУБД. Можно выделить следующие уровни:

* Предметная область;
* Модель предметной области;
* Логическая модель данных;
* Физическая модель данных;
* База данных и приложения.

**Предметная область** – это часть реального мира, данные о которой мы хотим отразить в базе данных. Например, в качестве предметной области можно выбрать бухгалтерию какого-либо предприятия, отдел кадров, банк, магазин и т. д. Предметная область бесконечна и содержит как существенно важные понятия и данные, так и малозначащие или вообще не значащие данные.

**Модель предметной области** – это наши знания о предметной области. Знания могут быть как в виде неформальных знаний в мозгу эксперта, так и выражены формально при помощи каких-либо средств. Гораздо более информативными и полезными при разработке баз данных являются описания предметной области, выполненные при помощи специализированных графических нотаций. Имеется большое количество методик описания предметной области. Из наиболее известных можно назвать методику структурного анализа SADT и основанную на нем IDEF0, диаграммы потоков данных Гейна-Сарсона, методику объектно-ориентированного анализа UML, и др. Модель предметной области описывает скорее процессы, происходящие в предметной области и данные, используемые этими процессами.

**Логическая модель** описывает понятия предметной области, их взаимосвязь, а также ограничения на данные, налагаемые предметной областью. Примеры понятий – «сотрудник», «отдел», «проект», «зарплата». Логическая модель данных является начальным прототипом будущей базы данных. Логическая модель строится в терминах информационных единиц, но без привязки к конкретной СУБД. Основным средством разработки логической модели данных в настоящий момент являются различные варианты ER-диаграмм (Entity-Relationship, диаграммы сущность-связь).

**Физическая модель данных.** На еще более низком уровне находится физическая модель данных. Физическая модель данных описывает данные средствами конкретной СУБД. Отношения, разработанные на стадии формирования логической модели данных, преобразуются в таблицы, атрибуты становятся столбцами таблиц, для ключевых атрибутов создаются уникальные индексы, домены преображаются в типы данных, принятые в конкретной СУБД. Ограничения, имеющиеся в логической модели данных, реализуются различными средствами СУБД, например, при помощи индексов, декларативных ограничений целостности, триггеров, хранимых процедур.

Собственно база данных и приложения как результат предыдущих этапов. Поэтому особую роль играет принятие правильных решений на ранних этапах моделирования.

ER-диаграммы

Схема «сущность-связь» (также ERD или ER-диаграмма) — это разновидность блок-схемы, где показано, как разные «сущности» (люди, объекты, концепции и так далее) связаны между собой внутри системы.

ER-диаграммы чаще всего применяются для проектирования и отладки реляционных баз данных, исследования и разработки программного обеспечения и информационных систем для бизнеса.

ER-диаграммы полагаются на стандартный набор символов, включая прямоугольники, ромбы, овалы и соединительные линии, для отображения сущностей, их атрибутов и связей. Эти диаграммы устроены по тому же принципу, что и грамматические структуры: сущности выполняют роль существительных, а связи — глаголов.

Изображение выглядит как текст, снимок экрана, диаграмма, Параллельный

Автоматически созданное описание

В ER-моделях и моделях данных обычно выделяют до трех уровней детализации:

* **Концептуальная модель данных** — схема наивысшего уровня с минимальным количеством подробностей. Достоинство этого подхода заключается в возможности отобразить общую структуру модели и всю архитектуру системы. Менее масштабные системы могут обойтись и без этой модели. В этом случае можно сразу переходить к логической модели.
* **Логическая модель данных** содержит более подробную информацию, нежели концептуальная модель. На этом уровне определяются более подробные операционные и транзакционные сущности. Логическая модель не зависит от технологии, в которой она будет применяться.
* **Физическая модель данных**: на основе каждой логической модели данных можно составить одну или две физических модели. В последних должно присутствовать достаточно технических подробностей для составления и внедрения самой базы данных.

Следует четко понимать, что цель ER-диаграмм — показать связи и отношения между элементами, поэтому они отображают только реляционную структуру.

Данные должны быть четко разбиты на поля, столбцы и строки, иначе пользы от ER-диаграммы будет мало. Это касается и частично структурированных данных, так, как только некоторые из них будут пригодны для работы.

**Символы и способы нотации ERD. Символы ERD-сущностей**

Под понятием «сущности» подразумеваются объекты или понятия, несущие важную информацию. С точки зрения грамматики, они, как правило, обозначаются существительными, например, «товар», «клиент», «заведение» или «промоакция». Ниже представлены три наиболее распространенных типа сущностей, используемых в ER-диаграммах.

|  |  |  |
| --- | --- | --- |
| Символ | Название | Описание |
| Символ независимой сущности | Независимая (сильная) сущность | Не зависит от других сущностей и часто называется «родительской», так как у нее в подчинении обычно находятся слабые сущности. Независимые сущности сопровождаются первичным ключом, который позволяет идентифицировать каждый экземпляр сущности. |
| Символ зависимой сущности | Зависимая (слабая) сущность | Сущность, которая зависит от сущности другого типа. Не сопровождается первичным ключом и не имеет значения в схеме без своей родительской сущности. |
| Символ ассоциативной сущности | Ассоциативная сущность | Соединяет экземпляры сущностей разных типов. Также содержит атрибуты, характерные для связей между этими сущностями. |

**Символы ERD-связей**

Связи используются в схемах «сущность-связь» для обозначения взаимодействия между двумя сущностями. Грамматически связи, как правило, выражаются глаголами, например, «назначить», «закрепить», «отследить», и несут полезную информацию, которую невозможно получить, опираясь только на типы сущностей.

|  |  |  |
| --- | --- | --- |
| Символ | Название | Описание |
| Символ | Связь | Отношение между сущностями. |
| Символ слабой связи | Слабая связь | Связь между зависимой сущностью и ее «хозяином». |

**Символы ERD-атрибутов**

ERD-атрибуты характеризуют сущности, позволяя пользователям лучше разобраться в устройстве базы данных. Атрибуты содержат информацию о сущностях, выделенных в концептуальной ER-диаграмме.

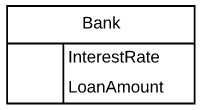
|  |  |  |
| --- | --- | --- |
| Символ | Название | Описание |
| Символ | Атрибут | Характеризует сущность, а также отношения между двумя или более элементами. |
| Символ многозначного атрибута | Многозначный атрибут | Атрибут, которому может быть присвоено несколько значений. |
| Символ производного атрибута | Производный атрибут | Атрибут, чье значение можно вычислить, опираясь на значения связанных с ним атрибутов. |
| Символ | Связь | Отношение между сущностями. |

**Символы физических ER-диаграмм**

Физическая модель данных — самый детальный уровень ER-схем, где представлен процесс добавления информации в базу данных. Физические модели «сущность-связь» отображают всю структуру таблицы, включая названия столбцов, типы данных, ограничения столбцов, первичные и внешние ключи, а также отношения между таблицами.

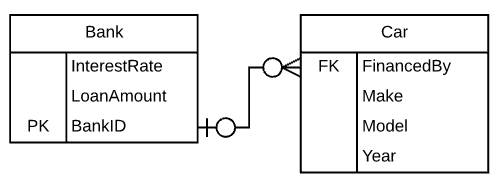
Ключевые составляющие таблиц «сущность-связь»:

* Поля — это участки таблицы, где задаются атрибуты сущностей. Под атрибутами обычно подразумеваются столбцы базы данных, которая моделируется по принципу «сущность-связь». В примере ниже InterestRate («процентная ставка») и LoanAmount («сумма займа») оба являются атрибутами сущности и оба находятся внутри полей.

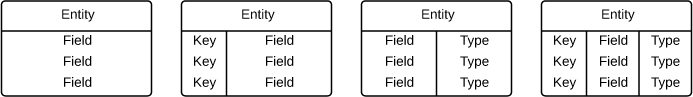


* Ключи — один из способов категоризации атрибутов. Напоминаем, что ER-диаграммы помогают пользователям моделировать базы данных посредством таблиц, которые обеспечивают им упорядоченность, эффективность и высокую скорость работы. Ну а ключи применяются с целью максимально эффективно связать между собой разные таблицы в базе данных.
  + Первичный ключ — это атрибут или сочетание атрибутов, идентифицирующих один конкретный экземпляр сущности.
  + Внешний ключ создается каждый раз, когда атрибут привязывается к сущности посредством единичной или множественной связи.

К примеру, займ на каждый отдельный автомобиль может быть выдан только одним банком, поэтому в качестве внешнего ключа FinancedBy («кем выдан займ») в таблице Car (автомобиль») использован основной ключ BankId («идентификатор банка»). При этом идентификатор BankId может служить внешним ключом сразу для нескольких автомобилей.



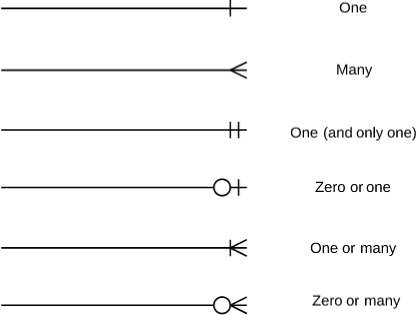
* Тип данных в соответствующем поле таблицы. Однако это также может быть и тип сущности, то есть описание ее составляющих. Например, у сущности «книга» будут следующие типы: «автор», «название» и «дата публикации».



**Кардинальность и ординальность**

Под кардинальностью подразумевается максимальное число связей, которое может быть установлено между экземплярами разных сущностей. Ординальность, в свою очередь, указывает минимальное количество связей между экземплярами двух сущностей.

Кардинальность и ординальность отображаются на соединительных линиях согласно выбранному формату нотации.



Нормализация отношений

Нормализация — это процесс организации данных в базе данных, Она включает в себя создание таблиц и установление связей между ними в соответствии с правилами, разработанными как для защиты данных, так и для повышения гибкости базы данных, устраняя избыточность и несогласованную зависимость.

Избыточность данных приводит к непродуктивному расходованию свободного места на диске и затрудняет обслуживание баз данных. Например, если данные, хранящиеся в нескольких местах, потребуется изменить, в них придется внести одни и те же изменения во всех этих местах. Изменение адреса клиента проще реализовать, если эти данные хранятся только в таблице Customers и нигде в базе данных.

Существует несколько правил нормализации баз данных. Каждое правило называется «обычной формой». Если соблюдается первое правило, база данных, как говорят, находится в «первой нормальной форме». При соблюдении первых трех правил база данных считается в «третьей нормальной форме».

Как и во многих формальных правилах и спецификациях, реальные сценарии не всегда позволяют обеспечить идеальное соответствие требованиям. Как правило, для выполнения нормализации приходится создавать дополнительные таблицы, и некоторые клиенты считают это нежелательным. Собираясь нарушить одно из первых трех правил нормализации, убедитесь в том, что в приложении учтены все связанные с этим проблемы, такие как избыточность данных и несогласованные зависимости.

**Первая нормальная форма**

* Устраните повторяющиеся группы в отдельных таблицах.
* Создайте отдельную таблицу для каждого набора связанных данных.
* Идентифицируйте каждый набор связанных данных с помощью первичного ключа.
* Не используйте несколько полей в одной таблице для хранения похожих данных. Например, для слежения за товаром, который закупается у двух разных поставщиков, можно создать запись с полями, определяющими код первого поставщика и код второго поставщика.

**Вторая нормальная форма**

* Создайте отдельные таблицы для наборов значений, относящихся к нескольким записям.
* Свяжите эти таблицы с помощью внешнего ключа.Записи не должны зависеть от чего-либо, кроме первичного ключа таблицы (составного ключа, если это необходимо).

**Третья нормальная форма**

* Исключите поля, которые не зависят от ключа.

Значения в записи, которые не являются частью ключа этой записи, не принадлежат в таблице. Если содержимое группы полей может относиться более чем к одной записи в таблице, попробуйте поместить эти поля в отдельную таблицу.

Выполнять нормализацию до третьей нормальной формы может быть целесообразно только для часто изменяемых данных. Если при этом сохранятся зависимые поля, спроектируйте приложение так, чтобы при изменении одного из этих полей пользователь должен был проверить все связанные поля.

**Другие формы нормализации**

Четвертая нормальная форма, также называемая Boyce-Codd нормальной форме (BCNF), и пятая нормальная форма существуют, но редко рассматриваются в практическом проектировании. Игнорирование этих правил может привести к не совсем идеальному дизайну базы данных, но не должно влиять на функциональные возможности.

# Билет №12.

Управление, основанное на целях. Требования к целям организации. Концепции Х и Y. Принцип интеграции.

**Управление, основанное на целях**

Управление — это воздействие (субъекта управления) на управляемую систему (объект управления) с целью обеспечения требуемого её поведения или изменения её характеристик

Цель - стремление (дело), ради выполнения которого объединяют и концентрируют свои усилия члены организации.

Цель – это мечта со сроком исполнения.

Общепризнанные цели организации в идеале должны удовлетворять и индивидуальные потребности отдельных людей, и организации в целом (теория Y).

Цели (objectives) – конечные результаты запланированной деятельности. Показывают ЧТО и К КАКОМУ СРОКУ должно быть достигнуто. Повозможности должны представляться в числовом виде. Достижение целей должно приводить к выполнению миссии компании. Пример: «с 2023 по 2027 г.г. в два раза увеличить объем продаж, повышать каждый год чистый доход более, чем на 20%» Задача (goal) – заявление открытого типа о том, чего организация хочет добиться, но без количественного уточнения результатов и без установления временных сроков выполнения. Направления, по которым часто устанавливаются цели и задачи: рентабельность (чистая прибыль); производительность (издержки); рост бизнеса (повышение активов, продаж); благосостояние акционеров, репутация; забота о сотрудниках; вклад в общество (налоги, благотворительность, товары, услуги); лидерство на рынке; технологическое лидерство (инновации); предотвращение банкротства; личные потребности руководителей и др.

Управление, основанное на цели (целях), опирается на следующие базовые положения: 1) постановка конкретных и одновременно достижимых целей; 2) содействие процессу участия в выработке целей сотрудников, которые будут работать над их достижением; 3) обеспечение обратной связи (оценки эффективности полученных результатов).

**Требования к целям организации**

Требования целям 1. Цели должны быть конкретными и измеримыми, чтобы создать основу для последующих решений и оценки хода работы («Не можешь измерить, не сможешь управлять») 2. Цели должны охватывать весь горизонт прогнозирования. Данное условие предполагает, что цели могут быть долгосрочными (от 3-х до 5-ти лет), которые разрабатываются в первую очередь, среднесрочными (до 2-х лет) и краткосрочными (от месяца до года). 3. Принятая к исполнению цель должна быть достижимой. 4. Цели должны быть взаимно поддерживающими. В противном случае может возникнуть конфликт между различными группами влияния и подразделениями организации, которые функционально отвечают за достижение разных целей организации. 5. Цели должны быть направлены на достижение интересов основных групп влияния, т.е. детализировать миссию - контракт (компромисс) между заинтересованными группами, например, соответствовать а) ожиданиям собственников по достижению финансовых показателей, б) ожиданиям персонала – по достижению удовлетворенности от работы и пребывания в организации, в) ожиданиям клиентов по потребительским свойствам и цене товара (услуги).

Изображение выглядит как текст, линия, диаграмма, Шрифт

Автоматически созданное описание

**Концепции Х и Y.**

Дихотомия менеджмента: концепции Х и Y (Дуглас Макгрегор, 1906-1964)

Теория Х (административная школа, А. Файоль): Организация – механизм, без активного вмешательства менеджеров люди пассивны, мало работают, им не хватает честолюбия, ответственности, сопротивляются переменам, легковерны, предпочитают, чтобы ими управляли. Теория Y: Организация – организм, люди – творцы, любят свою работу, сами управляют своей деятельностью, принимают на себя ответственность. Экономическими стимулами невозможно удовлетворить неэкономические потребности людей: социальные (в общении), эгоистические (относятся к уверенности, независимости, в поощрении, уважении), а также потребность в самовыражении (в раскрытии человеком своего потенциала, в саморазвитии).

Ситуационный характер концепций Х и Y Какая теория правильная? Имеют значение стиль управления и этапы развития организации (учебной группы): • На этапе создания в целях консолидации, формирования культуры необходима организация как механизм, ориентированный на власть, статус, контроль (теория Х); • На этапе стабилизации организация – организм (возникают взаимодействия, группы, интересы, культура, ориентация на личную ответственность и компетентность, теория Y); • На этапе изменений необходимо сочетание и контроля (Х) для быстрого выхода из кризиса, и обучения (Y) для создания инноваций.

Лидерство (руководство) – способность человека влиять на других людей во имя достижения групповых (организационных) целей.

Руководство (лидерство) – это «работа с людьми» в отличие от административной бумажной работы или самостоятельной деятельности человека по осознанию и решению проблем.

Идеальный руководитель должен стремиться к балансу качеств лидера и качеств менеджера, управляющего подчиненными

Качества лидера: метательность, неравнодушие, творчество, гибкость, умениче вдохновлять, новаторство, воображение. Качества менеджера: рациональнось, скловнность давать советы, настойчивость, аналитический склад ума, властность.

Власть – способность человека влиять на поведение других людей. Источник власти формального руководителя (директора) – должность. Власть в этом случае должна быть направлена на достижение целей организации, поддержание стабильности и порядка. Власть лидера проистекает из его личности (интересов, целей, ценностей), которые могут использоваться на благо организации, но остаются «собственностью» человека. Лидер может проявиться в экстремальной ситуации, не обладая при этом никакими формальными полномочиями. 5 видов власти: законная, основанная на вознаграждении, основанная на принуждении, экспертная и референтная. 2 источника власти: 1) занимаемая должность (учитель имеет власть по отношению к детям); 2) личные качества лидера.

**Принцип интеграции.**

Принцип интеграции Важнейшая задача менеджмента: Создание таких условий, при которых члены организации наилучшим образом достигали бы своих личных целей (удовлетворяли личные потребности), одновременно внося вклад в достижении целей всей организации. Вера побуждает человека действовать. Подчиненные (ученики) рано или поздно начнут себя вести в соответствии с ожиданиями менеджера (учителя) – его ценностями. Люди (ученики) - не пассивны. Они становятся пассивными после некоторого опыта работы в организации или учебы в школе (классе), которыми руководит человек в соответствии с теорией «кнута и пряника» (теорией Х)

**Управление, основанное на целях.**

**Управление по целям** (*Management by Objectives*, MBO) — это процесс согласования целей внутри организации таким образом, что руководство компании и сотрудники разделяют цели и понимают, что они означают для организации.

Термин «Управление по целям» впервые был введён и популяризирован [Питером Друкером](https://ru.wikipedia.org/wiki/%D0%9F%D0%B8%D1%82%D0%B5%D1%80_%D0%94%D1%80%D1%83%D0%BA%D0%B5%D1%80) в 1954 году в книге «The Practice of Management».

Суть управления по целям заключается в кооперативном процессе определения целей, выбора направления действий и принятии решений. Важной частью управления по целям является измерение и сравнение текущей эффективности деятельности сотрудников между собой и с набором установленных стандартов. В идеале, когда сотрудники сами вовлекаются в процесс постановки целей и определения направления действий, необходимых для их достижения, в этом случае сотрудники более мотивированы на выполнение их обязанностей.

В начале определенного периода времени подразделениям и сотрудникам компании устанавливаются цели, от достижения которых, в частности, может зависеть переменная часть зарплаты. Эти цели должны отвечать условиям [SMART](https://ru.wikipedia.org/wiki/SMART), то есть быть конкретными (*specific*) для компании или её подразделения; измеримыми (*measurable*); достижимыми (*achievable*); обоснованными и последовательными (*relevant*) и с четкими сроками выполнения (*time-based*).

Сверху вниз устанавливаются цели для всех уровней компании, причем цели даже на самых нижних уровнях должны соответствовать целям и стратегии организации в целом.

Для оценки достижения целей определяются [ключевые показатели эффективности](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D1%8E%D1%87%D0%B5%D0%B2%D1%8B%D0%B5_%D0%BF%D0%BE%D0%BA%D0%B0%D0%B7%D0%B0%D1%82%D0%B5%D0%BB%D0%B8_%D1%8D%D1%84%D1%84%D0%B5%D0%BA%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D1%81%D1%82%D0%B8) (KPI).

**Концепции Х и Y. Принцип интеграции.**

Теория X и Теория Y - это две противоположные теории стилей управления, которые были предложены Дугласом МакГрегором в 1960-х годах. Согласно Теории X, люди ленивы, немотивированы и безответственны, и для того, чтобы они работали эффективно, необходимо применять авторитарные методы управления. Согласно Теории Y, люди обладают высокой мотивацией и способны к самоконтролю, но нуждаются в поддержке и помощи для достижения своих целей.

Теория X является одной из наиболее известных теорий мотивации и управления. Согласно данной теории, люди склонны избегать ответственности и лениться, поэтому для того, чтобы заставить их работать, необходимо применять жесткие методы контроля и наказания.

Перечислим, основные принципы теории X:

- Люди по своей природе являются ленивыми и избегают ответственности.

- Для того, чтобы заставить людей работать эффективно, необходимо использовать жесткие методы контроля и наказания.

- Руководитель должен быть авторитарным и контролировать все аспекты работы.

- Сотрудники должны быть мотивированы только материальными стимулами, такими как деньги и премии.

Теория X имеет свои недостатки, которые связаны с тем, что она игнорирует человеческий потенциал и мотивацию. Люди могут работать эффективно только при наличии определенных условий, таких как возможность самореализации, признание и уважение со стороны руководства.

В настоящее время теория X считается устаревшей и не соответствует современным требованиям к управлению персоналом. Вместо жестких методов контроля и наказания, современные руководители используют более гибкие методы мотивации, основанные на вовлеченности сотрудников в работу, развитии их потенциала и создании благоприятной рабочей среды.

Теория Y - теория, согласно которой люди по своей природе стремятся к развитию, самосовершенствованию и ответственности. Согласно теории Y, для того чтобы мотивировать людей на работу, необходимо создавать условия, которые позволят им развиваться и реализовывать свой потенциал.

Основными принципами теории Y являются:

- Люди по своей природе склонны к развитию и самосовершенствованию.

- Для того, чтобы люди работали эффективно, необходимо создавать благоприятные условия для их развития.

- Руководитель должен поддерживать своих сотрудников и помогать им в развитии.

- Сотрудники должны быть мотивированы не только материальными, но и нематериальными стимулами.

Теория Y имеет свои преимущества, которые заключаются в том, что она стимулирует развитие

персонала и способствует повышению эффективности работы. Однако, она также имеет свои недостатки, связанные с тем, что не всегда возможно создать благоприятные условия для развития всех сотрудников.

Сегодня теория Y продолжает развиваться и применяться в практике управления персоналом. Руководители используют различные методы мотивации и стимулирования, которые позволяют сотрудникам развиваться и достигать своих целей.

Данные теории формируют три стиля управления или руководства: авторитарный, демократический и либеральный. Каждый из них имеет свои достоинства и недостатки. Руководитель организации должен тщательно проанализировать все стили и выбрать наиболее подходящий для своего управления. Важно понимать, какой стиль позволит развивать компанию на длительное будущее, а не способствовать прибыли только в текущий момент [2].

Авторитарный стиль руководства - это стиль, при котором руководитель принимает все решения самостоятельно и не учитывает мнения подчиненных. Такой стиль может быть эффективным в ситуациях, когда требуется быстрое принятие решений и высокая степень контроля. Однако, он может привести к низкой мотивации и недовольству среди сотрудников. Это один из самых распространенных стилей руководства, при котором руководитель берет на себя все полномочия и принимает все решения самостоятельно. Такой стиль управления часто используется в крупных организациях, где требуется быстрая реакция на изменения внешней среды и высокая степень контроля.

Преимущества авторитарного стиля руководства:

- Высокая скорость принятия решений;

- Четкое распределение ролей и обязанностей;

- Контроль за выполнением задач;

- Быстрое реагирование на изменения внешней среды.

Недостатки авторитарного стиля:

- Отсутствие обратной связи и диалога с подчиненными;

- Невозможность учета индивидуальных особенностей каждого сотрудника;

- Ограничение инициативы и самостоятельности подчиненных;

- Высокий уровень стресса у подчиненных.

Чтобы использовать авторитарный стиль руководства эффективно, необходимо иметь определенные навыки и знания. Например, умение быстро принимать решения, контролировать выполнение задач и управлять конфликтами.

Однако, следует помнить, что авторитарный стиль может привести к снижению мотивации и эффективности работы сотрудников, а также к ухудшению отношений в коллективе. Поэтому, перед использованием авторитарного стиля управления, необходимо тщательно оценить все возможные риски и последствия.

Демократический стиль руководства предполагает участие всех сотрудников в принятии решений. Руководитель выслушивает мнение каждого сотрудника и учитывает его при принятии решений. Такой стиль способствует развитию командного духа и повышению мотивации. Однако, при этом может возникать проблема с быстрым принятием решений.

Демократический стиль управления - это стиль управления, при котором руководитель опирается на мнение своих подчиненных и учитывает их интересы и потребности. Он основан на принципах сотрудничества, открытости и доверия.

Демократический стиль управления предполагает, что руководитель должен быть доступен для своих подчиненных, общаться с ними на равных и выслушивать их мнения. Он должен создавать атмосферу доверия и уважения, чтобы сотрудники чувствовали себя комфортно и свободно выражали свои идеи и предложения.

При демократическом стиле управления руководитель не принимает решения единолично, а обсуждает их с подчиненными. Он прислушивается к их мнению и учитывает его при принятии решений. Такой подход помогает создать команду, которая работает вместе и достигает общих целей.

Однако, демократический стиль управления не всегда эффективен в сложных ситуациях, когда требуется быстрое принятие решений или когда необходимо сконцентрироваться на выполнении конкретной задачи. В таких случаях руководитель может взять на себя ответственность и принять решение самостоятельно.

В целом, демократический стиль управления является эффективным, если руководитель умеет балансировать между доверием и контролем, а его подчиненные готовы к сотрудничеству и готовы делиться своими идеями.

Либеральный стиль руководства предполагает максимальную свободу действий для сотрудников. Руководитель не вмешивается в работу сотрудников и не контролирует их действия. Такой стиль подходит для творческих и инновационных проектов, где требуется максимальная свобода действий. Однако, такой стиль может привести к потере контроля над процессами и снижению эффективности работы.

Либеральный стиль руководства - это один из самых мягких стилей управления, который характеризуется тем, что лидер предоставляет своим подчиненным большую свободу действий и не вмешивается в их работу. Такой стиль управления часто используется в творческих профессиях, где важно дать свободу самовыражения и возможность экспериментировать.

Основной принцип либерального стиля руководства - это доверие к своим подчиненным. Лидеры, использующие этот стиль, уверены в своих сотрудниках и доверяют им принимать решения самостоятельно. Они не ограничивают их свободу действий и позволяют им проявлять свою инициативу.

Один из главных преимуществ либерального стиля - это возможность развивать творческий потенциал сотрудников. Когда лидеры предоставляют своим подчиненным свободу действий, они могут проявить свои лучшие качества и внести свой вклад в развитие компании. Кроме того, такой стиль руководства помогает создавать благоприятную атмосферу в коллективе, где каждый чувствует себя свободным и уважаемым.

Тем не менее, либеральный стиль руководства может быть неэффективным в ситуациях, когда необходимо принимать быстрые решения или когда требуется высокая степень контроля. Также он может привести к тому, что сотрудники будут избегать ответственности и перекладывать ее на своих коллег.

Таким образом, либеральный стиль руководства подходит для компаний, где важны творческие способности сотрудников и их самостоятельность. В других случаях лучше использовать другие стили управления, такие как авторитарный или демократический.

# Билет №13.

Основы языка SQL: История возникновения. Основные возможности. Типы данных. Операторы. Базовые функции. Операции определения данных (DDL).

SQL или Structured Query Language (язык структурированных запросов) — язык программирования, предназначенный для управления реляционными базами данных. «Реляционные» означает, что все данные хранятся в виде взаимосвязанных таблиц. А SQL как раз используют для того, чтобы как-то влиять на элементы внутри этих таблиц: добавлять, удалять, изменять и так далее.

История возникновения.

История возникновения SQL берет начало в 1970-х годах в IBM. В это время сотрудники IBM, в частности, Дональд Чемберлин, Раймонд Бойс и Эдгар Кодд, работали над проектом System R, целью которого было создание системы управления базами данных с реляционной моделью данных.

Одним из ключевых элементов этой системы стал язык запросов, который затем стал известен как SEQUEL (Structured English Query Language). SEQUEL был создан для того, чтобы пользователи могли удобно обращаться к данным в реляционной базе. Он предоставлял удобный и понятный способ создания, изменения, и извлечения данных из базы.

Однако, когда IBM зарегистрировала торговую марку SEQUEL, возникла проблема с конфликтом имен с уже существующей компанией. В результате было принято решение переименовать язык в SQL (Structured Query Language). С течением времени SQL стал широко распространенным и стандартизированным языком запросов для работы с реляционными базами данных. ANSI (American National Standards Institute) и ISO (International Organization for Standardization) утвердили стандарт SQL, что сделало его общепризнанным и широко применяемым языком.

Основные возможности.

**Управление структурой базы данных** (DDL - Data Definition Language): SQL позволяет создавать, изменять и удалять структуру базы данных, включая создание, изменение и удаление таблиц, индексов, представлений и других объектов.

**Управление транзакциями (Transaction Control):** SQL поддерживает транзакции для обеспечения целостности данных, включая начало транзакции (BEGIN TRANSACTION), фиксацию изменений (COMMIT) и откат изменений (ROLLBACK).

**Функции и агрегация данных:** SQL предоставляет различные встроенные функции для работы с данными, такие как арифметические функции (SUM, AVG, MAX, MIN), строковые функции (CONCAT, SUBSTRING), функции работы с датами и временем, и другие.

**Управление доступом к данным** (DCL - Data Control Language): Управление правами доступа к данным в базе данных, такими как GRANT и REVOKE.

**Манипулирование данными** (DML - Data Manipulation Language): Выборка, добавление, обновление и удаление данных из таблиц.

**Фильтрация и сортировка данных** (WHERE и ORDER BY): SQL позволяет выбирать только нужные данные с помощью условий фильтрации (WHERE) и сортировать их по определенным критериям (ORDER BY).

**Объединение данных (JOIN):** SQL позволяет объединять данные из различных таблиц на основе определенных условий связи.

Типы данных.

Типы данных SQL необходимы для организации работы всей базы. В таблице каждый столбец обязан иметь имя и указание на информацию, которая в нем содержится. Это определяет как язык, на котором написана БД, будет взаимодействовать с наполнением таблицы.

Строковые типы данных

CHAR – фиксированная длина строки. В процессе создания таблицы определяется точное значение – от 1 до 225 символов;

NCHAR – одна из разновидностей CHAR, которая поддерживает Unicode или многобайтовые символы

TEXT (она же LONG, VARCHAR или MEMO) – строки переменной длинны

NVARCHAR – подвид TEXT, которые поддерживает Unicode или многобайтовые символы

Числовые типы данных

NUMBER или FLOAT – числа с плавающими точками;

NUMERIC или DECIMAL – числа с фиксированными или плавающими точками;

BIT – одноразрядное значение, которое используют для битовых флагов: 0 или 1;

REAL – 4-байтовые числа с плавающими точками;

INTEGER или INT – целые 4-байтовые числа, у которых диапазон значений варьируется от -2147483648 до 2147483647;

TINYINT – целые 1-байтовые числа в диапазоне от 0 до 255;

SMALLINT – целые 2-байтовые числа в диапазоне от -32768 до 32767.

Типы данных, включающие обозначение даты и времени

DATE – дата;

TIME – время;

TIMESTAMP или DATETIME – дата и время;

SMALLDATETIME – дата и время с точностью до минуты.

Бинарные типы данных

BINARY – данные в двоичном виде в диапазоне от 255 до 8000 байт;

RAW – данные фиксированной длинны в двоичном виде в диапазоне до 255 байт;

LONG RAW – данные переменной длины в двоичном виде в диапазоне до 2Гбайт;

VARBINARY – данные переменной длины в двоичном виде в диапазоне до 8000 байт или до 255 байт в зависимости от реализации.

Операторы.

Операторы SQL – это символы и слова, помогающие выполнить определенные операции, например, выбрать конкретные данные из большой базы. SQL предоставляет широкий спектр операторов для выполнения различных операций над данными.

**Группы операторов SQL**

Операторы базы SQL подразделяются на несколько основных групп по признаку типа задач, которые можно решить с их помощью.

**DDL (Data Definition Language):** представляют собой группу операторов для определения данных. Они работают с целыми таблицами. Такие операторы SQL используются в тех случаях, когда нужно внести в базу новую таблицу или, напротив, удалить старую. Они включают в себя следующие командные слова:

**CREATE** — создание нового объекта в существующей базе.

**ALTER** — изменение существующего объекта.

**DROP** — удаление объекта из базы.

**DML (Data Manipulation Language):** эти операторы языка SQL предназначены для манипуляции данными. С их помощью меняется наполнение таблиц. Они позволяют изменять значение строк, столбцов и прочих атрибутов. Такие операторы SQL, например, позволяют удалить информацию о сотруднике, который больше не работает в компании, или исправить данные действующих специалистов. Эти операторы SQL представлены следующими командными словами:

**SELECT** — позволяет выбрать данные в соответствии с необходимым условием.

**INSERT** — осуществляют добавление новых данных.

**UPDATE** — производит замену существующих данных.

**DELETE** — удаление информации.

**WHERE** — Фильтрация данных по определенным критериям.

**JOIN** — Объединение данных из нескольких таблиц.

**GROUP BY**— Группировка данных для агрегации.

**ORDER BY** — Сортировка результатов запроса.

**DCL (Data Control Language):** это операторы SQL, предназначенные для определения доступа к данным. С их помощью можно закрыть или открыть для пользователей работу с базой. Такие операторы необходимы, чтобы ограничить кого-либо из сотрудников в доступе к информации или, наоборот, позволить работать с базой новому специалисту.

**GRANT**— предоставляет доступ к объекту.

**REVOKE**— аннулирует выданное ранее разрешение на доступ.

**DENY**— запрет, который прекращает действие разрешения.

**TCL (Transaction Control Language):** предназначен для управления транзакциями, то есть таким сочетанием команд, которые выполняются в определённом алгоритме. Транзакция проведена успешно, если все необходимые команды выполнены пошагово. Если же в какой-либо из них произошёл сбой, то вся операция, включая предыдущие команды, отменяется. Простым и понятным примером таких операторов SQL является проведение банковских платежей.

При этом вы сначала вводите сумму, а затем подтверждаете отправку платежа кодом, который вам присылает банк. Если операция не будет подтверждена, то транзакция отменится автоматически.

**BEGIN TRANSACTION**— начало транзакции.

**COMMIT TRANSACTION**— изменение команд транзакции.

**ROLLBACK TRANSACTION**— отказ в транзакции.

**SAVE TRANSACTION**— формирование промежуточной точки сохранения внутри операции.

**Операторы сравнения:** операторы сравнения SQL в результате своей работы позволяют выяснить, верна или нет запрашиваемая информация. По-английски эти значения определяются как true или false.

**=** Этот SQL оператор не арифметическое «равно», известное со школы, а сравнение на равенство. Если равенство верное, то получится результат TRUE, если нет – FALSE.

**!=** SQL оператор не равно или сравнение на неравенство, TRUE система выдаст, если значения будут не равны.

<> Аналогичный предыдущему SQL оператор. TRUE мы получим в том случае, если значения будут не равны.

> SQL оператор больше. Если левая часть (ее еще называют левый операнд) больше правой, то результат TRUE.

< SQL оператор меньше. Принимает значение TRUE, когда правый операнд больше левого.

>= SQL оператор больше либо равно. Выдает значение TRUE, если правая часть больше либо равна левой.

<= SQL оператор меньше либо равно. Значение TRUE появляется тогда, когда правая часть больше либо равна левой.

!< SQL оператор не меньше. Приобретает значение TRUE, когда правая часть не меньше левой.

!> SQL оператор не больше. Принимает значение FALSE, когда правая часть не больше левой.

**Логические операторы**

ALL - осуществляет вывод всех значений из таблицы.

AND - представляет собой получение результата при соблюдении двух поставленных условий.

ANY - Осуществляет сравнение текущего задания с дополнительным запросом.

BETWEEN - Для этого оператора SQL условия можно установить в определённом диапазоне. Для корректной работы нужно задать минимальное и максимальное значение.

EXISTS - Применяется тогда, когда нужно обозначить, интересует ли пользователя результат подзапроса.

IN - указывает, с какими значениями нужно вывести строки.

LIKE - Данный оператор позволяет осуществлять поиск подстроки в тексте и, если подстрока найдена, то она выводится.

NOT - Аннулирует любые условия.

OR - Дает результат в том случае, когда значение TRUE есть хотя бы в одном из операндов.

IS NULL - позволяет проверить значение поля на NULL.

UNIQUE - позволяет проверить уникальность каждой строки.

**Арифметические операторы:** Арифметические операторы SQL являются самыми простыми и наиболее популярными. Учиться пользоваться ими все начинают ещё в школе.

+ Бинарный оператор сложения.

— Бинарный оператор вычитания.

\* Бинарный оператор умножения.

/ Бинарный оператор деления.

% Оператор деления по модулю. Результатом будет целочисленный остаток.

**Операторы агрегации**

COUNT: Подсчет значений

SUM: Сумма значений

AVG: Среднее значение

MIN: Минимальное значение

MAX: Максимальное значение

Базовые функции.

SQL предлагает различные встроенные функции для выполнения операций над данными:

Агрегатные функции (например, SUM, AVG, MAX, MIN)

Строковые функции (например, CONCAT, SUBSTRING, UPPER, LOWER)

Функции работы с датами и временем (например, DATE\_FORMAT, DATE\_ADD, DATE\_DIFF)

Логические функции (например, AND, OR, NOT)/

**DDL (Data Definition Language)** — это подмножество языка SQL, используемое для определения структуры базы данных и её объектов, таких как таблицы, представления, индексы и процедуры.

**Некоторые распространённые DDL-операторы включают:**

1. **CREATE** — создаёт новый объект базы данных, такой как таблица, представление или индекс.
2. **ALTER** — используется для изменения существующего объекта базы данных.
3. **DROP** — используется для удаления существующего объекта базы данных.
4. **TRUNCATE** — используется для удаления всех строк в таблице, но сохраняет структуру таблицы и индексы.
5. **RENAME** — используется для переименования существующего объекта базы данных.

DDL-операторы выполняются немедленно и являются постоянными, то есть после создания, изменения или удаления объекта изменения невозможно отменить.

# Билет №14.

Сущность структурного подхода. Фазы проекта: анализ поведения системы и анализ данных. Средства функционального моделирования. Средства моделирования данных..

Исторически первым был структурный подход, который возник в конце 60-х вместе с появлением методологии SADT (Structured Analysis and Design Technique), которая позже вошла в набор стандартов IDEF (Icam DEFinition). Набор этих стандартов был разработан в рамках программы ICAM (Integrated Computer Automated Manufacturing), проводимой по инициативе ВВС США.

Суть структурного подхода заключается в функциональной декомпозиции, т.е. задача представляется как одна большая функция, которая преобразует входные данные в выходные. Данная функция разбивается на подфункции, которые, в свою очередь, тоже разбиваются на подфункции, и так до тех пор, пока в очередном разбиении не отпадает необходимость вследствие тривиальности соответствующей функции. В качестве языка используется графическая нотация диаграмм SADT. Кроме диаграмм SADT в структурном подходе используются также диаграммы потоков данных (DFD – Data Flow Diagram), на которых изображаются потоки данных между функциональными узлами системы, а также диаграммы «сущность–связь» (ERD – Entity-Relationship Diagram), которые описывают структуры данных.

Принципы структурного подхода: 1. Разделяй и властвуй. Для решения сложной задачи используется ее разбиение на несколько более простых подзадач. 2. Иерархическое упорядочивание. Решение задачи представляется в виде иерархии описаний (функций, потоков данных), где описания вышестоящих уровней детализируются при помощи описаний нижестоящих уровней. 3. Абстрагирование. При решении задачи рассматриваются только важные ее аспекты; все незначительное отбрасывается. 4. Формализация. Все принятые решения описываются формально. 5. Непротиворечивость. Никакая часть системы не может противоречить никакой другой. Все должно быть согласованно и находиться в соответствии.

Достоинства структурного подхода. Пожалуй, самым большим достоинством структурного подхода является возможность эффективной реализации. Действительно, единственное, на что приходится тратить процессорное время кроме решения собственно задачи, – это на вызовы функций и возвраты из них. Таким образом, структурный подход оказывается полезным в 41 тех случаях, когда важна максимальная производительность, например, для задач реального времени. Недостатки структурного подхода. Основным недостатком структурного подхода является его чувствительность к изменениям – малейшее изменение в постановке задачи может привести к необходимости серьезной переделки всей иерархии, т.к. все функции жестко связаны в иерархию и определяются друг через друга. В случае с информационными системами, когда постановка часто меняется (например, вследствие естественной эволюции предметной области), такое свойство подхода является крайне нежелательным.

**Фазы проекта**.Весь проект разделяется на 4 фазы: анализ, глобальное проектирование(проектирование архитектуры системы), детальное проектирование и реализация (программирование).

На фазе **анализа** строится модель среды (Environmental Model). Построение модели среды включает: **анализ поведения системы** (определение назначения ИС, построение начальной контекстной диаграммы потоков данных (DFD) и формирование матрицы списка событий (ELM), построение контекстных диаграмм); **анализ данных** (определение состава потоков данных и построение диаграмм структур данных (DSD), конструирование глобальной модели данных в виде ER-диаграммы).

|  |
| --- |
| В качестве предметной области используется описание работы видеобиблиотеки, которая получает запросы на фильмы от клиентов и ленты, возвращаемые клиентами. Запросы рассматриваются администрацией видеобиблиотеки с использованием информации о клиентах, фильмах и лентах. При этом проверяется и обновляется список арендованных лент, а также проверяются записи о членстве в библиотеке. Администрация контролирует также возвраты лент, используя информацию о фильмах, лентах и список арендованных лент, который обновляется. Обработка запросов на фильмы и возвратов лент включает следующие действия: если клиент не является членом библиотеки, он не имеет права на аренду. Если требуемый фильм имеется в наличии, администрация информирует клиента об арендной плате. Однако, если клиент просрочил срок возврата имеющихся у него лент, ему не разрешается брать новые фильмы. Когда лента возвращается, администрация рассчитывает арендную плату плюс пени за несвоевременный возврат.  Видеобиблиотека получает новые ленты от своих поставщиков. Когда новые ленты поступают в библиотеку, необходимая информация о них фиксируется. Информация о членстве в библиотеке содержится отдельно от записей об аренде лент.  Администрация библиотеки регулярно готовит отчеты за определенный период времени о членах библиотеки, поставщиках лент, выдаче определенных лент и лентах, приобретенных библиотекой. |
| Весь проект разделяется на 4 фазы: анализ, глобальное проектирование (проектирование архитектуры системы), детальное проектирование и реализация (программирование).  На фазе анализа строится модель среды (Environmental Model). Построение модели среды включает:   * анализ поведения системы (определение назначения ИС, построение начальной контекстной диаграммы потоков данных (DFD) и формирование матрицы списка событий (ELM), построение контекстных диаграмм); * анализ данных (определение состава потоков данных и построение диаграмм структур данных (DSD), конструирование глобальной модели данных в виде ER-диаграммы).   Назначение ИС определяет соглашение между проектировщиками и заказчиками относительно назначения будущей ИС, общее описание ИС для самих проектировщиков и границы ИС. Назначение фиксируется как текстовый комментарий в "нулевом" процессе контекстной диаграммы.  Например, в данном случае назначение ИС формулируется следующим образом: ведение базы данных о членах библиотеки, фильмах, аренде и поставщиках. При этом руководство библиотеки должно иметь возможность получать различные виды отчетов для выполнения своих задач.  Перед построением контекстной DFD необходимо проанализировать внешние события (внешние объекты), оказывающие влияние на функционирование библиотеки. Эти объекты взаимодействуют с ИС путем информационного обмена с ней.  Из описания предметной области следует, что в процессе работы библиотеки участвуют следующие группы людей: клиенты, поставщики и руководство. Эти группы являются внешними объектами. Они не только взаимодействуют с системой, но также определяют ее границы и изображаются на начальной контекстной DFD как терминаторы (внешние сущности).  Начальная контекстная диаграмма изображена на рисунке 2.42. В отличие от нотации Gane/Sarson внешние сущности обозначаются обычными прямоугольниками, а процессы - окружностями.  *Рис. 2.42. Начальная контекстная диаграмма*  Список событий строится в виде матрицы (ELM) и описывает различные действия внешних сущностей и реакцию ИС на них. Эти действия представляют собой внешние события, воздействующие на библиотеку. Различают следующие типы событий:   |  |  | | --- | --- | | **Аббревиатура** | **Тип** | | NC | Нормальное управление | | ND | Нормальные данные | | NCD | Нормальное управление/данные | | TC | Временное управление | | TD | Временные данные | | TCD | Временное управление/данные |   Все действия помечаются как нормальные данные. Эти данные являются событиями, которые ИС воспринимает непосредственно, например, изменение адреса клиента, которое должно быть сразу зарегистрировано. Они появляются в DFD в качестве содержимого потоков данных.  Матрица списка событий имеет следующий вид:   |  |  |  |  | | --- | --- | --- | --- | | **№** | **Описание** | **Тип** | **Реакция** | | **1** | Клиент желает стать членом библиотеки | ND | Регистрация клиента в качестве члена библиотеки | | **2** | Клиент сообщает об изменении адреса | ND | Регистрация измененного адреса клиента | | **3** | Клиент запрашивает аренду фильма | ND | Рассмотрение запроса | | **4** | Клиент возвращает фильм | ND | Регистрация возврата | | **5** | Руководство предоставляет полномочия новому поставщику | ND | Регистрация поставщика | | **6** | Поставщик сообщает об изменении адреса | ND | Регистрация измененного адреса поставщика | | **7** | Поставщик направляет фильм в библиотеку | ND | Получение нового фильма | | **8** | Руководство запрашивает новый отчет | ND | Формирование требуемого отчета для руководства |   Для завершения анализа функционального аспекта поведения системы строится полная контекстная диаграмма, включающая диаграмму нулевого уровня. При этом процесс "библиотека" декомпозируется на 4 процесса, отражающие основные виды административной деятельности библиотеки. Существующие "абстрактные" потоки данных между терминаторами и процессами трансформируются в потоки, представляющие обмен данными на более конкретном уровне. Список событий показывает, какие потоки существуют на этом уровне: каждое событие из списка должно формировать некоторый поток (событие формирует входной поток, реакция - выходной поток). Один "абстрактный" поток может быть разделен на более чем один "конкретный" поток.   |  |  | | --- | --- | | **Потоки на диаграмме верхнего уровня** | **Потоки на диаграмме нулевого уровня** | | Информация от клиента | Данные о клиенте, Запрос об аренде | | Информация для клиента | Членская карточка, Ответ на запрос об аренде | | Информация от руководства | Запрос отчета о новых членах, Новый поставщик, Запрос отчета о поставщиках, Запрос отчета об аренде, Запрос отчета о фильмах | | Информация для руководства | Отчет о новых членах, Отчет о поставщиках, Отчет об аренде, Отчет о фильмах | | Информация от поставщика | Данные о поставщике, Новые фильмы |   На приведенной DFD (рисунок 2.43) накопитель данных "библиотека" является глобальным или абстрактным представлением хранилища данных.  Анализ функционального аспекта поведения системы дает представление об обмене и преобразовании данных в системе. Взаимосвязь между "абстрактными" потоками данных и "конкретными" потоками данных на диаграмме нулевого уровня выражается в диаграммах структур данных (рисунок 2.44).  На фазе анализа строится глобальная модель данных, представляемая в виде диаграммы "сущность-связь" (рисунок 2.45).  Между различными типами диаграмм существуют следующие взаимосвязи:   * ELM-DFD: события - входные потоки, реакции - выходные потоки * DFD-DSD: потоки данных - структуры данных верхнего уровня * DFD-ERD: накопители данных - ER-диаграммы * DSD-ERD: структуры данных нижнего уровня - атрибуты сущностей   На фазе проектирования архитектуры строится предметная модель. Процесс построения предметной модели включает в себя:   * детальное описание функционирования системы; * дальнейший анализ используемых данных и построение логической модели данных для последующего проектирования базы данных; * определение структуры пользовательского интерфейса, спецификации форм и порядка их появления; * уточнение диаграмм потоков данных и списка событий, выделение среди процессов нижнего уровня интерактивных и неинтерактивных, определение для них миниспецификаций.   *Рис. 2.43. Контекстная диаграмма*  *Рис. 2.44. Диаграмма структур данных*  Результатами проектирования архитектуры являются:   * модель процессов (диаграммы архитектуры системы (SAD) и миниспецификации на структурированном языке); * модель данных (ERD и подсхемы ERD); * модель пользовательского интерфейса (классификация процессов на интерактивные и неинтерактивные функции, диаграмма последовательности форм (FSD - Form Sequence Diagram), показывающая, какие формы появляются в приложении и в каком порядке. На FSD фиксируется набор и структура вызовов экранных форм. Диаграммы FSD образуют иерархию, на вершине которой находится главная форма приложения, реализующего подсистему. На втором уровне находятся формы, реализующие процессы нижнего уровня функциональной структуры, зафиксированной на диаграммах SAD.   *Рис. 2.45. Диаграмма "сущность-связь"*  На фазе детального проектирования строится модульная модель. Под модульной моделью понимается реальная модель проектируемой прикладной системы. Процесс ее построения включает в себя:   * уточнение модели базы данных для последующей генерации SQL-предложений; * уточнение структуры пользовательского интерфейса; * построение структурных схем, отражающих логику работы пользовательского интерфейса и модель бизнес-логики (Structure Charts Diagram - SCD) и привязка их к формам.   Результатами детального проектирования являются:   * модель процессов (структурные схемы интерактивных и неинтерактивных функций); * модель данных (определение в ERD всех необходимых параметров для приложений); * модель пользовательского интерфейса (диаграмма последовательности форм (FSD), показывающая, какие формы появляются в приложении и в каком порядке, взаимосвязь между каждой формой и определенной структурной схемой, взаимосвязь между каждой формой и одной или более сущностями в ERD).   На фазе реализации строится реализационная модель. Процесс ее построения включает в себя:   * генерацию SQL-предложений, определяющих структуру целевой БД (таблицы, индексы, ограничения целостности); * уточнение структурных схем (SCD) и диаграмм последовательности форм (FSD) с последующей генерацией кода приложений.   На основе анализа потоков данных и взаимодействия процессов с хранилищами данных осуществляется окончательное выделение подсистем (предварительное должно было быть сделано и зафиксировано на этапе формулировки требований в техническом задании). При выделении подсистем необходимо руководствоваться принципом функциональной связанности и принципом минимизации информационной зависимости. Необходимо учитывать, что на основании таких элементов подсистемы как процессы и данные на этапе разработки должно быть создано приложение, способное функционировать самостоятельно. С другой стороны при группировке процессов и данных в подсистемы необходимо учитывать требования к конфигурированию продукта, если они были сформулированы на этапе анализа. |

**3. Средства функционального моделирования**

Функциональное моделирование – это **вид моделирования, который подразумевает описание процессов в виде взаимосвязанных, четко структурированных функций**. Т.е. главный элемент – это функция (операция), а бизнес-процесс представляется в виде последовательности функций, преобразующих входы процесса в выходы с использованием определенных ресурсов.

Выбор CASE-средства для использования в проекте разработки конкретной информационной системы зависит от множества факторов и является весьма ответственной и сложной задачей.

В настоящее время на российском рынке программного обеспечения (ПО) представлены десятки CASE-средств для построения функциональных моделей в нотации IDEF0 для систем различного уровня сложности как отечественных, так и зарубежных производителей.

А) Business Studio 4.2 позволяет как построить комплексную иерархическую модель деятельности компании, так и описать ряд отдельных процессов [2]. Для этого в распоряжение бизнес-аналитика предоставляются наиболее популярные и удобные нотации моделирования: IDEF0, Процедура (Cross-Functional Flowchart), BPMN 2.0, Процесс (Basic Flowchart), EPC (Event-Driven Process Chain). Среди преимуществ программы можно выделить: возможности отката/повтора изменений, перенос на нижние уровни стрелок с диаграмм верхнего уровня (связность диаграмм модели по стрелкам), отображение туннелирования стрелок, создание отчетов, наличие русского языка модели и интерфейса. Однако для работы с программой требуется установка Microsoft Visio. На сайте доступна демонстрационная версия программы для ОС Windows.

Б) Dia 0.97 — свободно распространяемый кроссплатформенный редактор диаграмм, часть GNOME Office, который может быть установлен автономно. Он может быть использован для создания различных видов диаграмм: блок-схем алгоритмов программ, древовидных схем, статических диаграмм UML, структур баз данных, диаграмм «сущность – связь», радиоэлектронных элементов, потоковых диаграмм, сетевых диаграмм и других. К преимуществам программы относится наличие русского языка интерфейса и модели, возможность отката изменений. Среди недостатков можно отметить не очень удобную работу с элементами модели и нестабильную работу программы. На сайте доступна версия программы для операционной системы (ОС) Windows, MacOS и Linux.

В) RFFlow 5.06 rev2(trial) позволяет создавать широкий спектр диаграмм: блоксхемы и гистограммы, организационные схемы, графики аудита, причинноследственные диаграммы (диаграммы Исикавы), диаграммы Ганта и графики проектов, медицинские блок-схемы), поддерживает нотации IDEF0, IDEF1X, UML, DFD, ERD и др. [1]. К преимуществам программного средства можно отнести наличие функций отката и повтора изменений. Однако работа с объектами модели неудобна, после группировки объектов нет возможности разгруппировать их. На сайте доступна 30-дневная пробная версия программы для ОС Windows.

**3)** [616.pdf (ptsj.ru)](https://ptsj.ru/articles/616/616.pdf) тут много, поэтому пусть каждый определит сам, сколько ему нужно для понимания

4)**Инструменты для моделирования данных**

Моделирование данных - это процесс, используемый для определения и анализа требований к данным, необходимым для поддержки бизнес-процессов в рамках соответствующих информационных систем в организациях. Таким образом, в процессе моделирования данных участвуют профессиональные разработчики моделей данных, тесно работающие с заинтересованными сторонами бизнеса, а также с потенциальными пользователями информационной системы.

Моделирование и проектирование данных и приложений основывается на сформулированных требованиях и является весьма важной частью процесса создания готового продукта. Инструменты для поддержки данного этапа жизненного цикла приложений можно условно разделить на средства моделирования бизнес-процессов, средства проектирования данных и средства объектно-ориентированного моделирования. Отметим, однако, что сегодня многие компании производят все три категории инструментов, интегрирующихся между собой (например, позволяющих сгенерировать модель данных и модели бизнес-процессов или синхронизировать их между собой), либо реализуют функциональность нескольких разнотипных средств моделирования в одном продукте, поэтому применительно к указанной категории инструментов имеет смысл говорить о линейках продуктов различных производителей.

[**CA ERwin Modeling Suite 7.3**](https://www.interface.ru/home.asp?artId=7561)**-** мощная линейка интегрированных CASE-средств, которые позволяют моделировать различные аспекты деятельности предприятия и проектировать информационные системы.

**В  линейку CA ERwin Modeling Suite 7.3 входят:**

* [**CA ERwin Process Modeler (BPwin) 7.3**](https://www.interface.ru/home.asp?artId=7524)**-** средство функционального моделирования бизнес-процессов.
* [**CA ERwin Data Modeler (ERwin)**](https://www.interface.ru/home.asp?artId=7556)**7.3-** проектирование, документирование и сопровождение баз данных и хранилищ данных.
* [**CA ERwin Data Profiler 7.3**](https://www.interface.ru/home.asp?artId=22190)**-** решение для анализа и профилирования данных, обеспечивает эффективную обработку исходной **информации**
* [**CA ERwin Data Model Validator (ERwin Examiner) 7.3**](https://www.interface.ru/home.asp?artId=7574)**-** проверка структуры баз данных и качества моделей AllFusion ERwin Data Modeler
* [**CA ERwin Model Manager (ModelMart) 7.3**](https://www.interface.ru/home.asp?artId=7525)**-** среда для совместного моделирования в CA ERwin Data Modeler и/или CA ERwin Process Modeler

**Дополнительно:**

* [**CA ERwin Saphir Option**](https://www.interface.ru/home.asp?artId=7563)
* [**CA ERwin Model Navigator**](https://www.interface.ru/home.asp?artId=7564)

В России эта линейка продуктов весьма популярна, а AllFusion ERwin Data Modeler является одним из самых распространенных средств проектирования данных, главным образом благодаря поддержке широкого спектра СУБД.

[**Oracle Designer**](https://www.interface.ru/home.asp?artId=193#o2)представляет собой инструмент, позволяющий проектировать данные, моделировать бизнес-процессы, создавать диаграммы потоков данных и функциональные модели, а также реализовывать их в виде серверных объектов. Этот продукт главным образом предназначен для применения совместно с СУБД Oracle и поддерживает все особенности данной СУБД, хотя с его помощью можно осуществлять и обратное проектирование для СУБД других производителей.

[**Sybase PowerDesigner**](http://www.itshop.ru/Sybase/PowerDesigner/l3t1i1777)- это инструмент, в состав которого входят средства создания моделей и объектно-ориентированного моделирования. Помимо серверных СУБД производства Sybase, PowerDesigner способен работать с любыми ODBC-источниками, генерировать код клиентских приложений для PowerBuilder, а также классы Java и компоненты JavaBeans. Возможно и обратное проектирование диаграмм классов из исходных текстов Java. Набор функциональных возможностей продукта варьируется в зависимости от редакции. Отметим, что PowerDesigner весьма популярен на российском рынке, и отнюдь не только среди пользователей СУБД и средств разработки Sybase.

**System Architect** является универсальным средством, позволяющим осуществить не только проектирование данных, но и структурное моделирование. В его состав входит средство проектирования данных и создания ER-диаграмм. Этот продукт поддерживает СУБД практически всех ведущих производителей. Компоненты System Architect позволяют документировать процесс работы над проектом, включая техническое задание, план тестирования и др. С помощью System Architect возможно генерировать код клиентских приложений для Visual Basic, Delphi и PowerBuilder, классы C++.

[**Microsoft Office Visio 2007**](https://www.interface.ru/home.asp?artId=22609) представляет собой универсальное средство моделирования данных и приложений, поддерживающее и создание моделей данных, и объектно-ориентированное моделирование приложений. Как и подавляющее большинство средств проектирования данных, Visio позволяет производить прямое и обратное проектирование данных, поддерживает все ODBC- и OLE DB-источники данных и особенности серверных СУБД всех ведущих производителей. Помимо средств проектирования данных, Visio включает средства объектно-ориентированного моделирования и генерации кода приложений (главным образом для платформы Microsoft .NET).

[**IBM Rational Rose**](https://www.interface.ru/home.asp?artId=314) - одно из самых популярных средств объектно-ориентированного UML-моделирования приложений. Данный продукт позволяет решать практически любые задачи в проектировании информационных систем - от анализа бизнес-процессов и моделирования данных до генерации кода на различных языках программирования, а также обладает средствами интеграции с другими инструментами Rational, в частности с Requisite Pro.

**Rational XDE Professional** (IBM) - инструмент UML-моделирования, встраиваемый в среды разработки Microsoft Visual Studio .NET и IBM WebSphere Studio Application Developer. Этот продукт дает возможность осуществлять визуальное проектирование на основе диаграмм UML и по окончании процесса проектирования генерировать код на выбранном языке программирования, а также проводить двустороннюю синхронизацию кода и модели.

[**Borland Together 2008**](https://www.interface.ru/home.asp?artId=18125)является платформой для анализа и проектирования приложений, интегрирующейся с различными средствами разработки как самой компании Borland, так и других производителей (в частности, Microsoft). Данный продукт позволяет осуществлять моделирование и проектирование приложений и данных, причем степень его интеграции со средствами разработки на данный момент такова, что изменения модели данных приводят к автоматическому изменению кода приложения, равно как и изменения в коде приводят к изменению в моделях (указанная технология интеграции инструментов моделирования и средств разработки получила название LiveSource).

Из вышесказанного следует, что основной тенденцией развития средств моделирования в настоящее время является активное предложение средств интеграции их между собой и с инструментами поддержки других этапов жизненного цикла приложений, как-то: средств разработки, средств управления требованиями и средств управления изменениями. Происходившее в последние годы слияние компаний, специализирующихся на производстве подобных инструментов, позволило создать линейки продуктов, в своей совокупности реализующих все или почти все задачи, которые могут возникнуть на этапе проектирования приложений.

Еще одной тенденцией можно назвать появление таких средств моделирования, которые тесно интегрированы со средствами разработки не только на уровне синхронизации кода и модели, но и на уровне полного определения поведения приложения непосредственно в самой модели (как это, например, реализовано в архитектуре Model Driven Architecture).

Для графического представления модели данных в виде схемы или диаграммы системные аналитики используют разные способы. У каждого из них свой набор символов и правил их применения. Такие способы называются [нотациями](https://practicum.yandex.ru/blog/notacii-modelirovaniya-biznes-processov/).   
Выбор способа зависит от особенностей задачи. Например, для графического представления связей между сущностями в модели данных можно построить ER-диаграмму. При этом для концептуального уровня можно использовать нотацию Чена, а для логического — нотацию Мартина. Первая состоит из простых символов, поэтому её легче презентовать заказчику. Вторая — позволяет подробно описать атрибуты.  
  
Есть множество инструментов, которые упрощают создание моделей данных и последующую работу с ними. Обычно это программные решения со встроенным графическим интерфейсом, который поддерживает разные нотации моделирования данных. Вот несколько из них:  
  
● **Erwin Data Modeler** — программа для проектирования баз данных и создания ER-диаграмм. Инструмент поддерживает работу со многими популярными системами управления базами данных, например, MySQL и [PostgreSQL](https://practicum.yandex.ru/blog/chto-takoe-subd-postgresql/). После создания модели данных можно подключиться к базе и автоматически создать нужные таблицы. У программы есть бесплатная версия с ограниченной функциональностью — ERwin Data Modeler Community Edition.  
  
● **ER/Studio Data Architect** — приложение для проектирования и визуального моделирования баз и хранилищ данных. С помощью приложения можно создавать ER-диаграммы в разных нотациях для графического представления баз данных разных типов, в том числе реляционных. Приложение поддерживает интеграцию метаданных из хранилищ данных и [BI-платформ](https://practicum.yandex.ru/blog/bi-sistemy-business-intelligence/).  
  
● **Enterprise Architect** — платформа для моделирования бизнес-процессов, организационной структуры компании и информационных систем. На платформе есть набор UML-инструментов. UML (от англ. Unified Modeling Language) — это стандартизированный язык для визуализации сложных структур и процессов. Например, с помощью UML можно графически представить последовательность этапов обработки данных системой. UML знают многие IT-специалисты, поэтому чаще всего разработчики без труда поймут модель.

# **Билет №15.**

Анализ и визуализация данных в Yandex DataLens.

Yandex DataLens представляет собой мощный инструмент для анализа и визуализации данных, предоставляемый компанией Яндекс. Он позволяет пользователям проводить сложный анализ данных напрямую в интернет-браузере без необходимости установки специализированного программного обеспечения.

Основные возможности Yandex DataLens включают:

1. Импорт данных: Пользователи могут импортировать данные из различных источников, таких как файлы Excel, CSV, Google Sheets, а также базы данных и API.
2. Анализ данных: DataLens предоставляет широкий набор инструментов для анализа данных, включая фильтрацию, сортировку, группировку, расчет агрегатных показателей (суммы, средние значения и т. д.) и многие другие функции.
3. Визуализация: Одним из основных преимуществ DataLens является возможность создания различных типов визуализаций данных, таких как графики, диаграммы, сводные таблицы и дашборды. Это позволяет пользователю наглядно представить результаты анализа.
4. Работа с большими объемами данных: DataLens хорошо справляется с большими объемами данных благодаря использованию облачных вычислений, что позволяет обрабатывать и анализировать даже очень большие наборы данных.
5. Доступность и безопасность: Поскольку DataLens работает в облаке, данные доступны из любой точки мира и поддерживается высокий уровень безопасности.

**Анализ данных:**

* Фильтрация и сортировка: DataLens позволяет быстро фильтровать данные по различным критериям и сортировать их для выделения интересующих показателей или трендов.
* Группировка и агрегация: С помощью DataLens можно группировать данные по определенным полям (например, по дате, региону или категории) и вычислять агрегатные функции, такие как сумма, среднее, минимум, максимум и т. д., для каждой группы.
* Построение сводных таблиц: DataLens поддерживает создание сводных таблиц, которые позволяют анализировать данные с различными аспектами и уровнями детализации.
* Вычисление метрик и ключевых показателей: Инструменты DataLens позволяют проводить расчеты для определения ключевых метрик и показателей производительности, что помогает в выявлении тенденций и принятии информированных решений.

**Визуализация данных:**

* Графики и диаграммы: DataLens предоставляет множество типов графиков и диаграмм для визуализации данных, включая линейные графики, столбчатые диаграммы, круговые диаграммы, гистограммы, диаграммы рассеяния и т. д. Это позволяет быстро и наглядно представлять данные.
* Дашборды: С помощью DataLens можно создавать интерактивные дашборды, объединяющие несколько визуализаций и таблиц в одном месте. Дашборды удобны для мониторинга ключевых показателей и отслеживания производственных метрик в реальном времени.
* Интерактивность: Визуализации DataLens поддерживают интерактивность, позволяя пользователям проводить более детальный анализ данных, изменять параметры отображения и проводить динамические исследования.
* Экспорт и представление: DataLens позволяет экспортировать визуализации в различные форматы (например, изображения, документы), что удобно для представления результатов анализа перед другими пользователями или в документации.

**Преимущества визуализации и анализа данных в Yandex DataLens:**

1. Простота использования: Интерфейс DataLens интуитивно понятен и позволяет быстро освоить инструмент для проведения анализа данных даже без специальных навыков программирования.
2. Интерактивность и гибкость: DataLens предлагает широкий выбор инструментов для настройки визуализаций и анализа данных, что позволяет пользователям получать именно те результаты, которые им необходимы.

Быстродействие: Благодаря использованию облачных технологий DataLens обеспечивает высокую производительность даже при работе с большими объемами данных.

# **Билет №16.**

Анализ и визуализация данных в Microsoft Power BI

**BI** (от англ. Business Intelligence, «бизнес-интеллект») — **это** набор средств, методологий и концепций для сбора, обработки и анализа данных, поступающих как из самой компании, так и из внешних источников.

**Microsoft Power BI** — это система Business Intelligence, иначе говоря — платформа для бизнес-аналитики. Она позволяет аккумулировать и структурировать данные из разных источников, преобразовывая их в единый формат. Затем из данных можно создавать визуализации различного типа и объединять их в интерактивные отчёты для наглядного анализа. Отчёты можно персонализировать по желаемым показателям.

Платформа считается одним из самых простых в использовании решений для бизнес-аналитики и подходит не только профессиональным аналитикам данных, но и менеджерам, руководителям компании и сотрудникам различных специализаций.

Power Bi состоит из десктопного приложения, онлайн-сервиса и мобильного приложения. Создавать дашборды можно с помощью Power Bi Desktop, публиковать и просматривать отчёты — с помощью браузерной и мобильной версий.

**Программа Power BI** — сервисы, совместно формирующие интерактивные отчёты из не связанных между собой источников. Источниками могут быть базы данных Access и SQL, облачные сервисы, txt-файлы, csv-файлы, данные из API-хранилища, Excel- и Google-таблицы. Power BI без проблем интегрируется с Microsoft 365, Dynamics 365, Azure и другими приложениями.

Платформа позволяет анализировать все данные компании, визуализировать их и предоставлять сотрудникам доступ к ним. То есть с одной и той же информационной панелью могут работать одновременно несколько человек в разных точках планеты. Достаточно лицензии и доступа к беспроводной сети. При этом настройки доступа к программе Power BI можно персонализировать для каждого сотрудника, данные — шифровать с помощью криптографии, а доступ — мониторить в режиме реального времени.

Сервис Microsoft Power BI работает с использованием технологий ИИ. Это даёт пользователям возможность создавать модели [машинного обучения](https://practicum.yandex.ru/blog/chto-takoe-mashinnoe-obuchenie/) и анализировать структурированные и неструктурированные данные, включая даже изображения и текст.

А для интеграции аналитики данных с управлением контентом, аудиторией и кампаниями платформа Power Bi использует [Adobe Creative Cloud](https://customers.microsoft.com/en-us/story/adobe-azure-dynamics-power-bi). Это упрощает [работу с панелями мониторинга Microsoft Power BI в Adobe Analytics](https://powerbi.microsoft.com/en-us/integrations/adobe-analytics/). Благодаря этому аналитики, работающие с Power BI, могут использовать облачные каналы данных из Adobe Analytics, Dynamics 365 и других источников, чтобы создавать интерактивные панели мониторинга.

В программе также есть функция автоматического обновления данных: можно назначить расписание для обновления отчётов из подключённых источников. Для промышленного использования существует ещё серверная версия приложения. Но эта версия подходит для огромных корпораций, и новичку она не нужна.

Отчёты из Power Bi можно встраивать прямо в приложения и веб-страницы. Вот как пошагово выглядит обработка данных в системе Power Bi:

1. Загружаем данные из онлайн- или локального источника.
2. Группируем и обрабатываем загруженные в систему данные.
3. Созданные модели данных визуализируем.
4. Создаём интерактивный отчёт из нескольких моделей — по сути, это интеллектуальный интерактивный дашборд с многопользовательским доступом.
5. Используем полученную информационную панель для анализа и принятия бизнес-решений.

Анализ данных при помощи Microsoft Power BI не требует умения писать SQL-запросы и знать базы данных на профессиональном уровне. Но необходимо владеть инструментами для бизнес-аналитики в Excel: Power Query и Power Pivot, в том числе формулами DAX, — а ещё желательно уметь писать скрипты на языках программирования Python и R. Это может пригодиться при подключении источников данных.

Дело в том, что Power Bi Desktop состоит из трёх интегрированных сервисов, у каждого из которых собственный интерфейс:

● Power Query — редактор запросов, который загружает и очищает данные;

● Power Pivot — для работы с табличными данными в оперативной памяти;

● Power View — для визуализации и построения отчётов.

Обеспечить доступ к данным можно с помощью локальных и облачных сервисов Dynamics 365, Salesforce, Azure SQL DB, Excel и SharePoint.

Подготовить данные к анализу позволит Power Query.

Анализировать данные вы сможете с помощью языка DAX — язык запросов, который используют в анализе данных. С помощью него можно обрабатывать большие объёмы информации, проводить сложные расчёты, создавать дашборды.

Среди доступных видов визуализации в Power BI есть графики, линейчатые, комбинированные, водопадные, воронкообразные, круговые, точечные и другие диаграммы, датчики, спидометры, карточки, таблицы и даже географические карты.

Система состоит из трех больших блоков: панель мониторинга, приложения и отчеты.

1. Отчеты. Представляют собой страницы с визуализациями данных в виде диаграмм, графиков, карт и т.д. Их можно создавать самостоятельно внутри Power BI или импортировать с помощью панелей мониторинга. Отчеты могут быть созданы автоматически, если подключить Power BI к внешним хранилищам данных — книгам Excel, API различных систем, Google Документам, приложению SaaS и различным базам данных. В зависимости от наличия прав доступа можно просматривать отчеты без возможности редактирования и в режиме правки. Отчет может содержать одну или несколько страниц с визуализациями.
2. Панель мониторинга. Отдельная страница, на которой можно увидеть визуализированные данные (плитки и мини-приложения). Обычно ее называют полотном. Панели мониторинга нужны для того, чтобы иметь доступ ко всем необходимым сведениям, быть уверенным, что команда проекта пользуется одними и теми же данными, следить за состоянием дел в рамках задачи благодаря отображению ключевых метрик.
3. Приложения. Элемент системы, который объединяет все панели мониторинга и отчеты. Весь контент, связанный между собой, будет находиться в одном месте.

# Билет №17.

Процессы и методы управления архитектурой предприятия.

Процессы и методы управления архитектурой предприятия

В данном разделе рассматриваются общие процессы и методы, применимые для разных сценариев использования АП (при этом, однако, разные сценарии будут вносить свою специфику).

Классический цикл управления АП перекликается с идеями совершенствования и реинжиниринга бизнес-процессов и в упрощенном виде состоит из следующих процессов (этапов): • идентификация существующей АП (модель «как есть»); • анализ существующей АП, стратегических требований и проблем/возможностей компании; • проектирование перспективной АП (модель «как надо»); • планирование перехода в перспективное состояние из существующего (портфель проектов и программа развития); • мониторинг и контроль проводимых изменений, позволяющие замкнуть цикл управления АП.

Некоторые методологии АП уточняют и конкретизируют этот упрощенный цикл, указывая: • какие составляющие АП, насколько детально и в какой последовательности следует идентифицировать и проектировать; • как работать с уровнем охвата (компания в целом, бизнес-направление или функциональная область, отдельная способность); • как планировать и осуществлять переход между состояниями «как есть» и «как надо» — проектирование моделей промежуточных состояний, работа со сценариями; • какие техники и инструменты использовать для решения частных задач (например, разработка архитектуры бизнес-процессов). Другие методологии АП, зачастую современные, ставят под сомнение некоторые или все составляющие «упрощенного цикла» (подробнее см. об этом завершающий статью раздел «Тренды и перспективные направления развития архитектуры предприятия»), задаваясь следующими вопросами: • нужно ли строить модель «как есть»; • стоит ли проектировать АП или лучше работать с ограничениями на частные решения по трансформации предприятия; • как сочетать проектирование и планирование с гибкими итеративными (agile) методами? Рассматривая методы управления АП, следует отметить значительную роль моделирования в АП, которая прослеживается в работах зарубежных [Lankhorst, 2013] и отечественных авторов [Калянов, 2006]. Методы и связанные с ними технологии управления АП позволяют: представить знания об устройстве предприятия в структурированном виде (выделить объекты и связи): перевести информацию из неструктурированной в структурированную (визуальные и невизуальные модели); интегрировать структурированную информацию в соответствии с согласованной структурой; обеспечить непротиворечивость условий применения различных подходов за счет формирования общего языка или правил перевода между языками; • сформировать необходимые ракурсы (viewpoints) и представления (views) для разных заинтересованных сторон для «борьбы со сложностью» [ISO/IEC/IEEE 42010:2007]. За счет этого осуществляется «нарезка» общей модели организации на ситуативные, контекстно-ориентированные представления — аналитические отчеты и диаграммы, регламентирующие документы, спецификации для разработки и/или настройки информационных систем; • реализовать повторное использование знаний и обеспечить упрощение задач перепроектирования или проектирования новых организационно-технических конструкций (например, открытие нового филиала) за счет использования типовых повторно используемых элементов (например, бизнес-процессов, конфигураций систем и др.); • проектировать целевое состояние организации, в том числе рассматривать альтернативные варианты; проводить анализ вариантов целевого состояния, в том числе с помощью «что если» анализа и/или изучения влияния изменений АП на связанные компоненты (например, что будет, если поменять бизнес-процесс Х тем или иным образом, — как это повлияет на другие процессы, на организационную структуру, на использование информационных систем). Добавим, что многие перечисленные методы архитектура предприятия наследует из общих установок системной инженерии [Levenchuk, 2015; Левенчук, 2015]. Применение архитектуры предприятия в менеджменте и ее роль в интеграции различных управленческих дисциплин рассмотрены в [Арзуманян, Кудрявцев, 2015]. На основе данной работы центром компетенций по архитектуре предприятия EA Lab (http://ealab.org/) были описаны сценарии применения АП (табл.1). Исторически исходным сценарием использования АП является «Выравнивание бизнеса и ИТ» (Business and IT alignment) [Henderson, Venkatraman, 1993]. Постепенно стали развиваться другие сценарии, и сейчас все больше внимания начинает уделяться сценарию «Трансформация предприятия» (Enterprise transformation) [Harmsen et al, 2009; Labusch, Winter, 2013]. При этом в цифровой экономике данный сценарий автоматически подразумевает «Выравнивание бизнеса и ИТ». В Приложении статьи даны описания конкретных примеров использования АП в компаниях.

**Модульность**

Принцип модульности предполагает разделение архитектуры предприятия на отдельные модули или компоненты. Каждый модуль выполняет определенные функции и имеет четкие границы, что облегчает разработку, тестирование и поддержку системы. Модули могут быть независимыми и взаимозаменяемыми, что позволяет гибко изменять и расширять архитектуру предприятия.

**Иерархия**

Принцип иерархии предполагает организацию архитектуры предприятия в виде иерархической структуры. Высокоуровневые компоненты определяют общую стратегию и цели предприятия, а низкоуровневые компоненты реализуют конкретные функции и задачи. Иерархическая структура обеспечивает четкое распределение ответственности и управление ресурсами предприятия.

**Модульная связность**

Принцип модульной связности предполагает, что каждый модуль должен быть связан только с необходимыми ему модулями. Это позволяет уменьшить зависимости между компонентами и обеспечить их независимость. Модули должны взаимодействовать через четко определенные интерфейсы, что облегчает их совместную работу и обновление.

**Расширяемость**

Принцип расширяемости предполагает, что архитектура предприятия должна быть гибкой и способной к изменениям. Она должна позволять добавлять новые функции и компоненты, а также изменять существующие без необходимости полной переработки системы. Расширяемость обеспечивает адаптивность предприятия к изменяющимся требованиям и условиям рынка.

**Повторное использование**

Принцип повторного использования предполагает использование уже существующих компонентов и решений при разработке архитектуры предприятия. Это позволяет сократить время и затраты на разработку, а также повысить качество и надежность системы. Повторное использование также способствует стандартизации и унификации компонентов, что упрощает их интеграцию и совместную работу.

**Управление рисками**

Принцип управления рисками предполагает анализ и учет возможных рисков при разработке архитектуры предприятия. Риски могут быть связаны с техническими, организационными или экономическими аспектами. Управление рисками включает в себя их идентификацию, оценку, планирование и контроль. Это позволяет минимизировать негативные последствия рисков и обеспечить стабильность и надежность архитектуры предприятия.

Классический цикл управления архитектурой предприятия (АП) перекликается с идеями совершенствования и реинжиниринга бизнес-процессов и в упрощенном виде состоит из следующих процессов (этапов):

Изображение выглядит как текст, снимок экрана, диаграмма, Шрифт

Автоматически созданное описание

Некоторые методологии АП уточняют и конкретизируют этот упрощенный цикл, указывая:

* какие составляющие АП, насколько детально и в какой последовательности следует идентифицировать и проектировать
* как работать с уровнем охвата (компания в целом, бизнес-направление или функциональная область, отдельная способность)
* как планировать и осуществлять переход между состояниями «как есть» и «как надо» — проектирование моделей промежуточных состояний, работа со сценариями
* какие техники и инструменты использовать для решения частных задач (например, разработка архитектуры бизнес-процессов)

Другие методологии АП, зачастую современные, ставят под сомнение некоторые или все составляющие «упрощенного цикла», задаваясь следующими вопросами:

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рассматривая методы управления АП следует отметить значительную роль моделирования в АП, которая прослеживается в работах зарубежных и отечественных авторов. Методы и связанные с ними технологии управления АП позволяют:

* представить знания об устройстве предприятия в структурированном виде: визуальные и не визуальные модели, формирование общего языка или правил перевода между языками и пр.;
* сформировать необходимые ракурсы (viewpoints) и представления (views) для разных заинтересованных сторон для «борьбы со сложностью». За счет этого осуществляется «нарезка» общей модели организации на ситуативные, контекстно-ориентированные представления — аналитические отчеты и диаграммы, регламентирующие документы, спецификации для разработки и/или настройки информационных систем;
* реализовать повторное использование знаний и обеспечить упрощение задач перепроектирования или проектирования новых организационно-технических конструкций (например, открытие нового филиала) за счет использования типовых повторно используемых элементов (например, бизнес-процессов, конфигураций систем и др.);
* проектировать целевое состояние организации, в том числе рассматривать альтернативные варианты; проводить анализ вариантов целевого состояния, в том числе с помощью «что если» анализа и/или изучения влияния изменений АП на связанные компоненты.

Многие методы архитектура предприятия наследует из общих установок системной инженерии. Центром компетенций по архитектуре предприятия EA Lab были описаны сценарии применения АП (см. ниже). Исторически исходным сценарием использования АП является «Выравнивание бизнеса и ИТ» (Business and IT alignment). Постепенно стали развиваться другие сценарии, и сейчас все больше внимания начинает уделяться сценарию «Трансформация предприятия» (Enterprise transformation). При этом в цифровой экономике данный сценарий автоматически подразумевает «Выравнивание бизнеса и ИТ».

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание

Изображение выглядит как снимок экрана, текст, Мультимедийное программное обеспечение

Автоматически созданное описание

Изображение выглядит как снимок экрана, текст, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание

# Билет №18.

**Теория.** Моделирование бизнес-процессов: методы, нотации, инструменты, сравнительный анализ инструментов моделирования.

\*Модель — это графическое или текстовое представление бизнес-процессов и логической взаимосвязи между ними.

Моделирование бизнес-процессов (Business Process Modeling) — это методология, которая позволяет оптимизировать и улучшать работу организации с помощью создания упрощенных и абстрактных моделей разных бизнес-процессов.

Метод моделирования включает в себя последовательность действий, которые необходимо выполнить для построения модели, т. е. процедуру моделирования, и применяемую нотацию (язык). Язык моделирования имеет свой синтаксис (условные обозначения различных элементов и правила их сочетания) и семантику (правила толкования моделей и их элементов).

Методология включает в себя:

Метод моделирования — способ представления реального объекта с помощью объектов модели.

Процедуру — последовательность шагов по сбору и обработке информации.

Нотацию — язык моделирования. Каждый язык имеет свой синтаксис — условные обозначения элементов и правила их сочетания, а также семантику — правила толкования моделей и их элементов.

В основе методологии моделирования могут лежать три подхода:

1. Структурный подход рассматривает систему как набор элементов, подсистем и отношений между ними. Используется для организационного развития предприятий и компаний: ищет способы оптимизации, разрабатывает рабочие регламенты и должностные инструкции. Методологии: SADT, DFD, WFD.
2. Объектно-ориентированный подход рассматривает систему как набор взаимодействующих объектов. Объекты — предметы, которые преобразуются при выполнении процессов. При объектно-ориентированном подходе сначала выделяются объекты, а затем действия, в которых они участвуют. Подход используется для визуализации, конструирования и документирования. Методология: BAAM.
3. Интегрированный подход объединяет структурный и объектно-ориентированный подходы. Даёт полное и комплексное представление о моделируемом объекте. Методология: ARIS.

Разберём особенности популярных методологий моделирования.

SADT — методология структурного анализа и проектирования, разработанная Дугласом Россом в 1969-1973 годах. Объединяет и организует диаграммы в иерархические древовидные структуры — чем выше уровень диаграммы, тем она менее детализирована.

Диаграммы SADT состоят из: блоков, которые изображают активность моделируемой системы; дуг, которые связывают блоки вместе и отображают взаимодействия.

Методология применяется на ранних этапах создания системы для определения требований к ней. В США SADT успешно использовалась в военных и коммерческих организациях для долгосрочного стратегического планирования и управления финансами.

Особенности методологии SADT:

* Универсальность — может использоваться для проектирования сложных систем любого назначения: управления и контроля, телефонных сетей, учёт материально-технических ресурсов.
* Способность отражать такие системные характеристики, как управление, обратная связь и исполнители.
* Наличие процедур для поддержки коллективной работы.
* Возможность использования на ранних этапах создания системы и сочетания с другими структурными методами проектирования.

Самая распространённая нотация — IDEF0.

DFD — методология потоков данных. Описывает внешние по отношению к системе источники и адресаты, логические функции, потоки и хранилища данных. Может быть представлена в виде графического структурного анализа или диаграммы. На диаграмме отображают работы, которые входят в состав описываемого бизнес-процесса, а также входы и выхода каждой из них.

Методология применяется для моделирования информационных систем и выявления проблем документооборота. Описывает любые действия: процесс продажи или отгрузки товаров, работу с заявками, закупку сырья. DFD помогает понять, из чего должна состоять информационная система и как автоматизировать бизнес-процессы.

Особенности методологии DFD:

* описывает не столько бизнес-процессы, сколько движения потоков данных;
* процессы не существуют сами по себе, поэтому результат должен куда-то передаваться;
* используется при разработке программного обеспечения;
* нет ограничения по количеству элементов, которые могут находиться на одной диаграмме.

Самые распространённые нотации — Эд Йордана и Тома де Марко.

WFD — методология потоков данных. Описывает бизнес-процессы нижнего уровня, где возникает необходимость показать временную последовательность выполнения работ.

Методология применяется для моделирования таких бизнес-процессов компаний как: «Выставление счетов», «Подготовка договора», «Изготовление детали».

Особенности методологии WFD:

* Использует дополнительные объекты для описания процессов: логических операторов, события начала и окончания процесса, а также элементы, показывающие временные задержки.
* Показывает альтернативы, которые происходят в процессе. Например, с помощью методологии вы можете описать ситуацию, когда договор на меньшую сумму согласуется с одной группой сотрудников, а на большую — с другой группой по более сложной технологии.
* Стрелки между операциями бизнес-процесса обозначают не потоки объектов, последовательность выполнения работ.

Самая распространённая нотация — IDEF3.

ARIS — одновременно и методология, и программный продукт для моделирования бизнес-процессов организации. Методология ARIS разработана профессором Августом Шеером в 1990-х годах. Она представляет собой современный подход к структурированному описанию деятельности компании, представлению её в виде взаимосвязанных графических диаграмм, удобных для понимания и анализа.

Методология применяется на крупных или длительных проектах, а также предприятиях с достаточным оборотом денежных средств. Это обусловлено стоимостью внедрения и трудозатратами по сопровождению и поддержке. ARIS подходит для управленческого консалтинга, внедрения систем управления качеством, анализа и оптимизации бизнес-процессов. Она позволяет классифицировать и структурировать операционные риски, вести документооборот.

Особенности методологии ARIS:

* основывается на концепции интеграции, предлагает целостный взгляд на процессы;
* рассматривает и представляет любую организацию как единую систему;
* насчитывает более 80 моделей, поэтому для осмысленного применения требуется время;
* использует разные уровни описания: что система должна знать, какие у неё пути реализации, а также программные и технические средства.
* её внедрению должна предшествовать «ручная» проектно-аналитическая работа;
* главное преимущество — высокая степень визуализации бизнес-моделей.

Самые распространённые нотации — EPC, UML и BPMN.

BAAM — методология описания деятельности. Включает в себя шесть бизнес-моделей: ESM, BCM, BPM, BFM, BOM, ERM. С их помощью последовательно описывает функции, бизнес-процессы, организационные и структурные особенности компаний, её подразделения, а также материальные и информационные потоки между ними. Методология представляет собой схему, на которой вместо работ отображаются структурные подразделения и взаимодействия между ними.

Методология применяется для описания бизнес-процессов в крупных компаниях. Отображает подразделения и должности, которые есть в организации, а также связи линейного и функционального подчинения. Помогает проектировать базы данных.

Особенности методологии BAAM:

* описывает подразделения компании и потоки между ними;
* описывает бизнес-процессы отдельных подразделений;
* формирует управляющие работы, а также состояния, характеризующие начало и конец каждой работы;
* описывает должности организации;
* определяет структуру информации, которая необходима для бизнес-процессов.

Самые распространённые нотации — Нотация Питера Чена, нотация Гордона Эвереста Crow’s Foot.

Нотации — графические модели, которые используются для фиксации бизнес-процессов. Помогают наглядно представить алгоритм действий. Выше мы перечислили десять нотаций для разных методологий, но самые популярные из них — IDEF0, EPC, BPMN. Сравним их.

|  |  |  |  |
| --- | --- | --- | --- |
| **Критерий сравнения** | **IDEF0** | **EPC** | **BPMN** |
| **Принцип построения диаграммы** | Принцип доминирования | Временная последовательность выполнения процедур | Временная последовательность выполнения процедур |
| **Описание процедуры процесса** | Объект на диаграмме | Объект на диаграмме | Объект на диаграмме |
| **Модель отражает** | Структуру системы, функции, потоки ресурсов и информации | Структуру системы, функции, потоки ресурсов и информации | Функции системы, внутренние процессы |
| **Графические элементы** | Прямоугольники — действия и этапы.  Стрелки — ресурсы и исполнители | Фигуры разных цветов. Розовые — события, зелёные — функции, жёлтые — исполнители, серые — ресурсы, оранжевые — ИС.  Соединительные элементы — стрелки и разделители «и», «или» | Задача — прямоугольник, событие — круг, поток — стрелка. Также есть сноски и базы данных |
| **Достоинство** | Высокая степень детализации. Можно создать модель, которая будет учитывать практически все ресурсы, всех сотрудников | Простота восприятия | Простота восприятия.  Подходит для описания внутренних бизнес-процессов компании |
| **Недостаток** | Модель занимает много места | Приходится создавать события даже для незначительных этапов | Зациклена на бизнес-процессах, не подходит для описания структуры |
| **Сфера применения** | Долгосрочное планирование, управление финансами | Описание технологических процессов предприятия — выставление счетов, отгрузки товаров и т.д. | Управленческий консалтинг, внедрение систем управления качеством, оптимизация бизнес-процессов |

**К популярным нотациям моделирования бизнес-процессов относятся:**

1. **IDEF0** — нотация графического моделирования, которая используется для функционального моделирования.
2. **EPC** — графический стандарт моделирования бизнес-процессов, нотация класса WorkFlow.
3. **BPMN** — нотация моделирования бизнес-процессов, один из наиболее распространённых методов описания бизнес-процессов.
4. **UML** — язык графического описания для объектного моделирования в области разработки программного обеспечения.
5. **DFD** — методология графического структурного анализа, которая описывает внешние по отношению к информационной системе источники и адресаты данных, логические функции, потоки данных и хранилища данных.

## **Инструменты. Сравнительный анализ инструментов моделирования.**

На сегодняшний день спрос на BPM–системы обусловлен широким спектром их применения. Так нотация BPMN позволяет создавать "процессные картинки", проектировать автоматизацию бизнес-процессов, а если наибольший интерес представляет концепция непосредственно исполняемых бизнес-процессов, то BPMN нет реальной альтернативы.

Наиболее известные программные продукты: ELMA BPM, ARIS Business Architect, Oracle BPM Suite.

ARIS (Architectureof Integrated Information Systems) – тиражируемый программный продукт для моделирования бизнес-процессов организаций, использующий нотации eEPC (extended Event-driven Process Chain), ERM (Entity-Relationship Model), UML (Unified Modeling Language). ARIS предоставляет визуальный инструментарий для обеспечения наглядности моделей, с возможностью создания сценариев автоматизации, составления аналитических отчётов, нормативных документов, новых моделей. Каждый сценарий представляет собой подпрограмму, запускаемую в ARIS Business Architect или непосредственно на сервере ARIS.

Oracle BPM Suite – компонент портфеля решений Oracle Fusion Middleware. Этот продукт предназначен специально для управления интерактивными бизнес-процессами, реализуемых с участием бизнес-пользователей. Oracle BPM располагает средой моделирования, которая поддерживает одновременную работу бизнес-пользователей и сотрудников ИТ, сервер исполнения и среду для взаимодействия с конечными пользователями. Он совместим со стандартами Business Process Modeling Notation (BPMN) и XML Process Definition Language (XPDL).

ELMA BPM – работа данной системы основана на простой идее: на этапе внедрения системы осуществляется моделирование бизнес-процессов организации с помощью наглядных диаграмм смоделированных по стандарту BPMN. Затем эти диаграммы загружаются в компьютерную систему и становятся исполнимыми, то есть программа позволяет отследить и проконтролировать исполнение бизнес-процессов в реальной практике работы предприятия.

Все эти программные продукты обрабатывают полный жизненный цикл бизнес-процессов: моделирование, исполнение и улучшение.

Можно вспомнить программный продукт BPwin, который также применяют для моделирования бизнес-процессов, использующий различные нотации (IDEF0, IDEF3 и DFD). При всех его плюсах, существует значительный минус – этот программный продукт не динамичен, все схемы бизнес-процессов, построенные в данной программе нельзя анализировать в реальном времени.

Существует и ряд других программных продуктов, на которых мы останавливаться не будем.

Если сравнить три программных продукта, то ARIS Business Architect, Oracle BPM Suite и ELMA BPM имеют практически одинаковые характеристики. Но существование бесплатной версии системы ELMA делает её оптимальным решением для изучения моделирования бизнес-процессов в университете.

Изображение выглядит как текст, число, снимок экрана, Шрифт

Автоматически созданное описание

Программа «Business Studio» — это разработка, предназначенная для комплексной работы с бизнес-процессами, а также для формирования различных документов, отчетов, бизнес-моделей и многого другого. Продукт полностью локализован на русском языке, что делает очень удобной работу для русскоговорящих специалистов. Также в нем предусмотрена интеграция с такими системами как 1С, MS Excel и настраиваемый импорт в другие системы. Для знакомства с возможностями продукта есть демо-версия, а для разовых решений есть временные лицензии. Пользователи системы отмечают ее много функциональность, но вместе с тем и перегруженность функциями и правами пользователей. Разберем плюсы и минусы данной программы: + русскоязычное ПО; + широкий функционал; + автоматическая сборка отчетности; – слишком широкий функционал редактирования моделей; – не отслеживается изменение модели на смежных уровнях.

Visual Paradigm — это CASE средство для моделирования. Оно обеспечивает непрерывную работу над проектом внедрения или реинжиниринга. Кроме выше указанных нотаций поддерживаются UML, SysML, DFD и ERD. В дополнение к поддержке моделирования, он предоставляет возможности генерации отчетов и разработки кода, включая генерацию кода. Он может перепроектировать инженерные схемы из кода и предоставлять возможность двустороннего проектирования для различных языков программирования. Для решения базовых некоммерческих задач есть бесплатная версия. + генерация кода к модели; + генерация документации; + автоматическое редактирование декомпозиций при редактировании одного из уровней; + большое количество других нотаций; – нет русскоязычной версии.

Microsoft Visio — это графический редактор, поддерживающий множество нотаций моделирования. Средство подходит для создания отдельных диаграмм, но не для целостного проекта. С его помощью можно быстро и наглядно сделать какую-либо диаграмму. Так как оно не является специализированным ПО для описания бизнес-процессов, поэтому его функциональные возможности весьма ограничены и не позволяют проводить глубокий анализ и оценку моделей, разрабатывать какие-либо регламентирующие документы, а также автоматически вносить изменения в модели. + удобное редактирование моделей; – отсутствие декомпозиции процесса; – отсутствие генерации документации. Каждая из анализируемых программ имеет свои достоинства, которые определяют область их применения. Так, например, для ознакомления с проектированием в различных нотациях подойдет MS Visio, а для проектирования на предприятии реальных бизнес-процессов можно выбрать Business Studio или Visual Paradigm. Выбор зависит от специфики предприятия и желаемых результатов проектирования: преимущественно русскоязычная компания, которую удовлетворит минимальный проектный пакет, выберет Business Studio; более широкая в языковом плане компания, которой требуется более детальное описание бизнес-процессов и документирование этапов проекта, остановится на Visual Paradigm.

# Билет №19.

**Теория.** Сущность, сильные и слабые стороны основных видов структур организации. Проблема адекватности структуры и симптомы структурного несоответствия.

**Сущность, сильные и слабые стороны основных видов структур организации.**

Рассматриваются 6 видом структур: гибридная, модульная, горизонтальная, матричная, дивизиональная, функциональная (вертикальная).

1. Функциональная (вертикальная).

ФУНКЦИОНАЛЬНАЯ СТРУКТУРА - это деление организации на отдельные элементы, каждый из которых имеет свою четко определенную конкретную задачу и обязанности. Традиционные функциональные блоки - это отделы производства, маркетинга и финансов. Если объем организации велик, то основные функциональные отделы подразделяются на более мелкие вторичные подразделения. Функциональную структуру целесообразно использовать в организациях, выпускающих относительно ограниченную номенклатуру продукции и действующих в стабильных внешних условиях.

Изображение выглядит как текст, диаграмма, снимок экрана, Шрифт

Автоматически созданное описание

Сильные стороны:

1. Сосредоточение работников одной функции (задачи, роли) в одном месте, откуда – экономичность, отсутствие дублирования функций внутри организации и расходов

2. Рост профессионального мастерства работников одной или близких профессий, задач.

3. Эффективность, удобство при производстве изделий ограниченной номенклатуры.

Слабые стороны:

1. Медленная реакция на изменения в среде, т.к. требуется координация отдельных функциональных подразделений.

2. Накопление нерешенных вопросов на верхних уровнях иерархии

3. Медленная реализация инноваций, требующих, как правило, привлечения специалистов разного профиля.

4. Вероятность противопоставления целей подразделений целям организации

1. Дивизиональная структура

Деление организации на элементы и блоки происходит по видам товаров или услуг, группам покупателей или географическим регионам.

1. В продуктовой структуре полномочия по руководству производством и сбытом передаются одному руководителю, который являются ответственным за данный тип продукции (стратегическую бизнес-единицу). Продуктовая структура позволяет крупной фирме уделять конкретному продукту столько же внимания, сколько ему уделяет небольшая фирма.

Недостаток продуктовой структуры - увеличение затрат из-за дублирования одних и тех же видов работ для различных видов продукции. Может возникнуть конфликт между подразделениями, производящими близкие по назначению продукты (услуги)

Достоинство – гибкость, т.к. каждая единица – не велика, а принятие решений децентрализованное в отличие от функциональной структуры, в которой все решения, касающиеся нескольких функций адресуются «наверх».

Например, структура Johnson & Johnson включает 180 (!) независимых единиц бизнеса. Штаб квартира корпорации осуществляет управление дивизиональными подразделениями, имеющими собственную специфическую структуру. Другой пример – Microsoft, в которой принятие ключевых решений по бизнес-единицам осуществляют в продуктовых подразделениях : поставки корпоративным клиентам, отделом розничных продаж, отделом разработки ПО для игр, группой продаж для вспомогательных служб (бухгалтеров, малого бизнеса, интернет-провайдеров) и др.

2. Структура, ориентированная на потребителя, все подразделения группируются вокруг определенных групп потребителей. Этот вид структуры ассоциируется с предпринимательством, а также со сферой образования, где организации имеют не только традиционные общеобразовательные программы, но и спец. подразделения обучения взрослых (например, по заочной, вечерней формам), повышения квалификации специалистов, самообразования и др.

3. Региональная организационная структура, т.е. по месту расположения ее подразделений. Эта структура облегчает решение проблем, связанных с местным законодательством, обычаями и нуждами потребителей. При появлении зарубежных филиалов могут быть созданы соответствующие структуры для компаний, функционирующих на международном уровне.

Например, Apple Computer имеет региональную структуру, которая облегчает производство и доставку компьютеров по всему миру: Apple Americas (Канада, Латинская Америка, страны Карибского бассейна), Apple Europe, Apple Pacific (Азия, Австралия, Япония).

Достоинство – приспособление к специфическим потребностям региона.

Недостаток – работники отождествляют себя скорее с региональными, чем с общекорпоративными целями. Горизонтальная координация выражена гораздо больше, чем координация между региональными структурами или между региональными структурами и штаб-квартирой.

1. **Матричная структура**

Мультифокусная организация может быть одновременно ориентирована на продукт, функцию, регион. То есть для достижения целей такой организации необходимы одновременно как специальные знания, так и инновации (межпредметные знания). Пример: ИМИСП

В матричной структуре используются одновременно связи по вертикали и по горизонтали. Менеджеры по функциям и менеджеры по проекту обладают равной властью в организации, а исполнители отчитываются и перед теми и перед другими.

Изображение выглядит как снимок экрана, текст, диаграмма, линия

Автоматически созданное описание

Сильные стороны • Ресурсы (люди, техника) могут быстро распределяться между различными направлениями работы в соответствии с меняющимися требования среды. • Предоставляет возможность персоналу совершенствовать как специальное знание, так и знания, необходимые для выполнения новых проектов. • Лучше всего матрица работает в организациях средних размеров, производящих несколько видов услуг (товаров)

Слабые стороны • Двойное подчинение вызывает напряжения, путаницу, конфликты, много времени уходит на заседания и работу конфликтных комиссий • Предъявляет требования к навыкам сотрудничества и общения менеджеров и специалистов между собой

Пример – компания Worldwide Steel. 170 лет компания работала в условиях функциональной структуры. Чтобы отвечать современным требованиям пришлось осваивать сотни марок сталей ежемесячно. Появились 4 бизнес-менеджера: по штамповочной линии, прокатного стана, производства колес и валов, сталеплавильного производства. Расширились координационные встречи для принятия решений по отдельным видам продукции и работе функциональных подразделений. Молодых менеджеров стали привлекать к «матричным дискуссиям», что готовило их к будущей работе. Откуда – экономичность и гибкость в работе

Условия создания матрицы

1. Ограниченные ресурсы должны использоваться рационально. Например, в средней (по размерам) организации специалисты работают в различных проектах или по различным направлениям по мере необходимости, т.е. нет постоянных специалистов, ответственных за конкретные направления в проектах.

2. Внешняя среда предъявляет разные требования, например, одновременно и к техническим знаниям, и к скорости обновления продуктового портфеля компании. Откуда необходима структура двойственной власти (функциональной и горизонтальной).

3. Необходимо принимать судьбоносные решения в условиях сложности и неопределенности внешней среды. Частые изменения и взаимозависимость между подразделениями требуют координации и обмена информацией (знаниями) как по вертикали, так и по горизонтали.

4. Из-за сложности уравновешивания матрицы могут быть функциональные матрицы (большая часть властных полномочий передана руководителям функциональных отделов) и продуктовые матрицы (большую власть имеют менеджеры проектов, продукт-менеджеры, а руководители отделов направляют специалистов и проводят экспертизу проектов).

1. **Горизонтальная структура**

Горизонтальная структура – это объединение сотрудников вокруг конкретных бизнес-процессов.

Организации переходят к горизонтальной структуре путем реинжиниринга бизнес-процессов – замене вертикальной структуры горизонтальными производственными процессами.

Процесс – совокупность взаимосвязанных задач и видов деятельности, которые преобразуют входные ресурсы в выходные ресурсы, имеющие ценность для внешнего или внутреннего потребителя.

Команды, выполняющие целостные процессы, ориентированы на конечный результат, удовлетворяющий запросы потребителей, например, заказ на разработку нового продукта (услуги). Все работники, участвующие в конкретном процессе, более доступны друг для друга, лучше координируют усилия по обслуживанию клиентов.

Характеристики горизонтальной структуры организации

1. Стираются границы между подразделениями. У каждого процесса – свой руководитель. Например, вместо отраслевых отделов структурируются процессы: 1) развития бизнеса – создания новых продуктов (анализ рынка – исследование - планирование работы, контроль - потребитель); 2) снабжения – логистики (анализ потоков – снабжение – материалопоток – дистрибъюция - потребитель) и т.п.

2. Основа организации – самостоятельные команды, ответственные за конкретные процессы. Члены команд принимают решения самостоятельно, взаимозаменяемы. Повышается скорость реакции на возникающие новые требования.

3. Руководители процессов несут ответственность за их выполнение и качество

4. Эффективность работы измеряется удовлетворенностью клиентов и финансовыми издержками.

5. Формируется культура доверия, сотрудничества и ориентации на постоянные улучшения

Сильные стороны: гибкость и быстрота реакции на изменения потребностей заказчиков, повышается автономия и инициатива работников, акцент смещается в сторону командной работы и сотрудничества, улучшается психологический климат в организации (разделение ответственности за принятые решения)

Слабые стороны: определение процессов, их описание требует значительных затрат труда и времени, требуется изменение культуры, которое не может пройти безболезненно (демократический руководитель воспринимается как слабый, а менеджерам не нравится заниматься коучингом и консультированием сотрудников), повышаются требования к квалификации работников для выполнения многих функций в команде для обеспечения взаимозаменяемости, необходимо четкое определение критериев качества для внешних и внутренних потребителей.

1. **Модульная структура**

Модульная структура – это объединение самостоятельных компаний, координация деятельности которых осуществляется из небольшой штаб-квартиры. Головная организация оставляет за собой процессы, которые выполняет на мировом уровне и которые трудно воспроизвести, а другие виды деятельности перепоручает другим, осуществляя за ними контроль

В этой структуре функции по исследованию, проектированию, маркетингу, производству, сбыту могут быть переданы по аутсорсингу независимым компаниям, которые с помощью партнерских договоров и ИТ связаны с головным офисом.

Аутсорсинг – стратегия отказа от собственного производства и расширения количества приобретаемых у внешних поставщиков товаров (услуг) высокого качества и по низким ценам. Как правило, приобретаемые услуги не связаны с ключевыми компетенциями компании. Например, обувь Nike производится на азиатских фабриках, однако Nike удерживает лидерские позиции в моделировании обуви и ее распределении – двух ключевых направлениях деятельности.

В пределе - широко разбросанная сеть производителей и дистрибьюторов может работать как одна целостная компания. Традиционная иерархия заменяется на соблюдение законов свободного рынка (дешевле купить, дороже продать)

Сильные стороны: маленькие организации могут стать глобальными, используя ресурсы всего мира; есть возможность получать прибыль без крупных инвестиций в предприятия, оборудование и торговые помещения (ИКЕА); позволяет быстро разрабатывать товары (услуги) и выводить их на рынки; структура организации может быстро меняться; сокращаются административные расходы (управленческие кадры сосредоточены на ключевых направлениях деятельности).

Слабые стороны: отсутствие контроля за многими видами работ; много усилий на поддержание отношений с партнерами; риск, связанный с выходом партнера из бизнеса; приверженность работников и культура слабые, т.к. сотрудники работают на контрактной основе (работники понимают, что они могут быть легко заменены), откуда повышается текучесть кадров

**Изображение выглядит как текст, снимок экрана, диаграмма, линия

Автоматически созданное описание**

Головной офис занимается покупкой прав и поиском авторов. Издательский модуль обеспечивает право на издание иллюстраций, изготовление макетов, написание рекламных текстов и подготовку рукописи к верстке. Служба дизайна готовит макет книги и верстает его. Верстка передается в типографский модуль, где книги печатают и отправляют заказчику. PR-фирма занимается продвижением издания, контактами с книжными магазинами, организацией отзывов на издание в газетах и журналах.

1. **Гибридная структура**

Гибридная структура – такая, с помощью которой организации пытаются избежать недостатков других структур и которая сочетает в себе характеристики функциональной, продуктовой, региональной, горизонтальной и модульной структур, что позволяет уйти от недостатков, присущих отдельным видам структур.

1.Чаще объединяются характеристики функциональной и дивизионной структур. Функции, необходимые для работы по конкретным направлениям децентрализуются и передаются самостоятельным подразделениям. Другие функции, требующиеся для обслуживания дивизиональных подразделений, остаются неизменными и не выносятся за рамки головного офиса, т.е. остаются в рамках функциональной структуры.

2. Сочетание функциональной и горизонтальной структур: пример компании Форд, которая после увеличения жалоб по сервису перестроила сервисную службу, часть функций была передана командам, отвечающим за определенное направление (снабжение, сбор информации о проблемах ремонта, техническое обеспечение сервисных подразделений)

**Изображение выглядит как текст, снимок экрана, диаграмма, Шрифт

Автоматически созданное описание**

**Проблема адекватности структуры и симптомы структурного несоответствия.**

Проблема: Нахождение правильного соотношения между контролем по вертикали и координацией по горизонтали. Цели вертикального контроля – производительность, контроль и стабильность. Типичная структура – функциональная, необходима специализация задач для эффективного использования ограниченных ресурсов. Цели горизонтальной координации – обучение, инновации, гибкость. Типичные структуры – горизонтальная и модульная. Горизонтальная структура необходима для реализации нововведений и содействия научению, однако эта структура дорогая. Модульная структура – адаптивная, позволяет создавать или ликвидировать модули в зависимости от изменений на рынке. Промежуточное положение занимают функциональная структура с горизонтальными связями, матричная и дивизиональная структура.

**Симптомы структурного несоответствия**

1. Решения принимаются с задержками или недостаточно качественные. ЛПР перегружены, т.к. вынуждены решать слишком много проблем. Делегирование полномочий развито недостаточно. Качество решений низкое, т.к. информация поступает не по адресу, либо нет баланса между вертикальными и горизонтальными информационными потоками 2. Организация не производит инновации в ответ на изменения в окружающей среде. Причина – отсутствие координации подразделений по горизонтали, например, должно быть согласовано выявление потребностей клиентов отделом маркетинга и создание новых товаров (услуг) отделом разработок и исследований. 3. Слишком много конфликтов. Структура должна способствовать объединению конкурирующих целей разных подразделений. В этом смысле акцент на горизонтальные связи и автономию подразделений не всегда правилен

Дополнительно:

Энциклопедический словарь: организация – 1) внутренняя упорядоченность, согласованность, взаимодействие частей целого; 2) совокупность процессов или действий, ведущих к образованию связей между частями целого; 3) объединение людей, совместно реализующих программу (цель) и действующих на основе определенных правил и процедур.

Таким образом, в общем случае организация – это упорядоченное состояние элементов целого и процесс по их упорядочению в целесообразное единство

Индукция – движение мысли от единичного к всеобщему, от знания меньшей степени общности к обобщенному знанию.

На основе анализа степени организованности отношений между частями систем выделяют следующую их иерархию: 1. Простые структуры: а) статические структуры – взаимосвязи между элементами, которые можно рассматривать как фактическую основу всякого знания (атомы в молекуле, карты Земли, Солнечной системы, скелет человека); б) простые динамические структуры с детерминированным движением, например, часовой механизм, двигатели, динамо-машины, теоретические знания в области физики, химии, экономики; в) саморегулируемые организации с обратной связью, например, термостат, простейшие автоматические устройства, поддерживающие заданное состояние. 2. Сложные структуры: а) открытые самонастраивающиеся организации – уровень, на котором происходит отделение органической жизни от неживой материи. Пример – клетка, огонь, реки; б) растительный мир – возникает разделение труда между клетками, формирующими различные «клеточные сообщества», например, листья, зерна; в) живые организмы – в отличие от растений формируются специальные информационные рецепторы (глаза, уши и т.п.), развивается нервная система, организующая информацию и регулирующая поведение животного. 3. Сверхсложные структуры: а) человек как организация, обладающая сознанием, языком, способностями самопознания. Человек не только знает нечто, но и знает, что знает; б) социальные организации – общественные образования, целенаправленно координирующие деятельность людей. Объект исследования – не индивид, а его роли. Социальную организацию можно определить как ряд ролей, связанных через систему коммуникаций; в) непознанные (до поры до времени) и непознаваемые организации – антимиры, черные дыры, соседние галактики

Дополнительные:

# **Вопрос 1. Базовые подходы к хранению и извлечению информации. Управление большими данными: сбор, оценка качества и подготовка.**

Какая бы модель не была использована для хранения и обработки данных необходимо, чтобы выполнялись **правила полноты, непротиворечивости и целостности данных**.

***Полнота данных*** — база данных должна обеспечивать полное и адекватное описание предметной области. При этом должен соблюдаться принцип минимальной избыточности. Особое внимание на полноту обращается на этапе проектирования базы данных.

***Непротиворечивость данных*** — данные, которые хранятся в базе данных, должны проверяться на правильность при вводе, существует запрет на дублирование данных.

***Целостность данных:***

1. при описании связей должна обеспечиваться правильность ссылок между таблицами, что обеспечивается каскадным обновлением и удалением
2. блокировка модифицируемых записей, при одновременной работе с БД
3. механизм транзакций (последовательность операций над БД), позволяющий вернуться на несколько шагов назад, отменив последние действия, осуществив «откат»

Основываясь на физическом представлении организации хранения данных, можно выделить следующие виды архитектуры для хранения данных:

***Локальные базы данных***

Располагаются на компьютере, на котором работает пользователь. Вся информация используется в монопольном режиме. Пользователь сам регулирует доступ к данным.

***Централизованные базы данных***

Централизованная база данных хранится на центральном компьютере, пользователи и прикладные программы имеют удаленный доступ к базе данных. Преимущества централизованной БД — минимальные затраты на корректировку. Такая система предпочтительна, если важны требования к безопасности и целостности данных. Недостатком является сложность в обслуживании, увеличение времени отклика, затраты на передачу данных, неисправность центральной системы выводит из рабочего состояния всю сеть.

Централизованные базы данных реализуются на базе двух архитектур с сетевым доступом:

* Архитектура «файл – сервер» предполагает выделение одной из машин в сети в качестве центральной (сервер файлов), на которой хранится совместно используемая централизованная база данных. Остальные машины сети исполняют роль рабочих станций, на которых в основном и производится обработка данных, получаемых в виде файлов базы данных в соответствии с запросами пользователей;
* Архитектура «клиент – сервер» стала стандартом для современных СУБД, когда сервер владеет и распоряжается информационными ресурсами системы, а клиент пользуется ими. Центральная машина (сервер базы данных) помимо хранения базы данных обеспечивает выполнение основного объема обработки данных. Запрос клиента (рабочей станции) порождает поиск и извлечение данных на сервере, которые затем транспортируются по сети к клиенту (в отличие от передаваемых файлов в предыдущей архитектуре).

***Распределенные базы данных***

Распределенная база данных предполагает хранение и управление данными в нескольких узлах компьютерной сети и передачу данных между ними в процессе выполнения запросов. На разных компьютерах могут храниться не только различные таблицы, но и разные фрагменты одной огромной таблицы. При этом для пользователя не имеет значения, как организовано хранение данных.

***Извлечение данных*** предполагает извлечение информации из неорганизованных источников данных. Извлеченные данные впоследствии импортируются в промежуточную область реляционной базы данных. Исходная система запрашивает данные через интерфейсы прикладного программирования, и применяется логика извлечения. Благодаря этому процессу данные теперь готовы к прохождению фазы преобразования процесса ETL

**Виды данных:**

1. **Неструктурированные данные -** Аудио, электронная почта, геоточки, данные с датчиков и камер - это распространенные примеры, и обычно они поступают из Интернета вещей (IoT). Компании должны сначала выполнить операции по подготовке и очистке данных, такие как удаление повторяющихся результатов, удаление лишних символов и определение того, как обрабатывать пропущенные значения, прежде чем извлекать неструктурированные данные.
2. **Структурированные данные -** Структурированные данные хранятся и управляются в транзакционной системе стандартизированным образом. **Строки в таблице базы данных SQL представляют структурированные данные**. Предприятия обычно извлекают информацию из исходной системы при работе со структурированными данными.

**Методы извлечения данных:**

1. ***Web Scraping*** - Веб-скрапинг используется для сбора данных из различных онлайн-источников, таких как веб-сайты электронной коммерции, новостные сайты и платформы социальных сетей. Программное обеспечение для очистки веб-страниц получает доступ к веб-страницам, анализирует содержимое HTML или XML и извлекает определенные элементы данных.
2. ***Извлечение на основе*** ***API*** - Многие веб-сервисы предоставляют API, которые позволяют разработчикам получать данные из приложений в структурированном формате. Извлечение на основе API включает отправку HTTP-запросов к этим API и последующее получение данных. Это надежный и структурированный способ извлечения данных из онлайн-источников, таких как платформы социальных сетей, метеорологические службы или поставщики финансовых данных.
3. ***Извлечение текста*** (обработка естественного языка – НЛП) - Методы извлечения текста часто используют обработку естественного языка (NLP) для извлечения информации из неструктурированных текстовых данных, таких как документы, электронные письма или сообщения в социальных сетях. Методы НЛП включают распознавание именованных объектов (NER) для извлечения таких объектов, как имена, даты и местоположения, анализ настроений и классификацию текста для извлечения информации из текста.
4. ***OCR*** - Оптическое распознавание символов (OCR) преобразует печатный или рукописный текст из документов, изображений или отсканированных страниц в машиночитаемые и редактируемые текстовые данные. Программное обеспечение OCR анализирует обработанные изображения для распознавания и преобразования текстового содержимого в машиночитаемые символы. Механизмы оптического распознавания символов используют различные методы для определения чувств, включая распознавание образов, извлечение признаков и алгоритмы машинного обучения.
5. ***Разбор документа или анализ документов*** — это когда компьютерная программа или система извлекает структурированную информацию из неструктурированных или полуструктурированных документов. Эти документы могут быть в различных форматах, например PDF-файлы, Файлы Word, HTML-страницы, электронные письма или рукописные заметки. Система синтаксического анализа определяет структуру документа. Затем он извлекает соответствующие элементы данных, включая имена, адреса, даты, номера счетов и описания продуктов, на основе определенных ключевых слов, регулярных выражений или других методов сопоставления с образцом.

**Типы извлечения данных:**

1. Полное извлечение - Полное извлечение или полная загрузка или обновление извлекает все данные из исходной системы за одну операцию. Вы можете **использовать этот метод, когда исходные данные меняются нечасто и необходима полная и актуальная копия данных**. Однако полное извлечение данных может быть ресурсоемким, особенно для больших наборов данных, поскольку при этом извлекаются все данные независимо от того, изменились ли данные с момента предыдущего извлечения.
2. Инкрементное извлечение - также называемое дельта-извлечением или сбор измененных данных (CDC), **используется для извлечения только тех данных, которые изменились с момента последнего извлечения**. Это лучший выбор при работе с часто меняющимися источниками данных, такими как транзакционные базы данных. Кроме того, это более эффективно, чем полное извлечение, поскольку уменьшает объем передаваемых и обрабатываемых данных
3. Ручное извлечение – метод при котором копируют и вставляют данные из документов, электронных таблиц или веб-страниц в другое приложение или базу данных. Однако ручное извлечение **требует много времени, подвержено ошибкам и неизбежно непригодно для крупномасштабных задач извлечения данных**. Тем не менее, это может быть полезно для периодического или специального извлечения данных, когда автоматизация невозможна.

**Оценка качества** включается в себя такие элементы как:

* **Полнота** - включает в себя оценку наличия пропущенных значений и атрибутов, которые могут быть не представлены в данных
* **Точность -** включает в себя проверку наличия ошибок, неточностей, дубликатов и неправильно заполненных или некорректных значений
* **Согласованность -** включает в себя проверку соответствия типов данных, форматов и правил целостности данных
* **Достоверность -** включает в себя проверку аутентичности и правдоподобности данных, а также их соответствия фактическим событиям или явлениям

**Подготовка данных** представляет собой процесс преобразования данных для дальнейшего анализа, обработки и использования. Включает в себя:

1. Очистка данных – удаление дубликатов, исправление ошибок, обработка пропущенных значений, удаление выбросов
2. Интеграция данных – объединение данных, согласование структуры данных
3. Преобразование данных – преобразование типов данных, нормализация данных, преобразование формата данных
4. Уменьшение размера данных – выборка данных, агрегация данных
5. Разбиение данных – разделение данных для построения (обучающая и тестовая выборка) и оценки производительности (тренировочная, валидационная и тестовая выборки) модели

Вопрос 2. Процессный подход к управлению. Основные понятия процессного подхода, определение бизнес-процесса, методология описания бизнес-процессов, классификация процессов.

Методики проектирования и автоматизации бизнес-процессов являются в настоящее время одним из важнейших инструментов повышения эффективности бизнеса. Многие крупные российские компании используют программные средства для описания и оптимизации бизнес-процессов. Использование подобных методик и программных средств имеет своей конечной целью реорганизацию бизнес-процессов и, как следствие, увеличение выручки, сокращение затрат на производство продукции и услуг, повышение качества продукции, оптимальное использование оборотного капитала, внедрение систем автоматизации и многое другое.

Так или иначе, выполняемые в организациях проекты связаны с разработкой и внедрением новых систем управления или их элементов. Как правило, руководители организаций ожидают от внедрения значительного улучшения деятельности, например, сокращения затрат. Иногда практический результат от внедрения систем трудно измерим. Обычно в большой компании всё начинает двигаться не сразу. На первых порах польза может быть малозаметна или вообще отсутствовать. Но со временем, когда процесс налаживается, появляется и результат. Ведь "хорошо" сразу бывает редко, обычно "хорошо" нужно подождать. Большие финансовые вложения в системы управления и отсутствие видимого результата приводят как к формированию негативного отношения, так и к дискредитации философии управления на основе процессного подхода.

Основополагающей базой современных подходов к управлению является процессный подход, который предполагает определение набора бизнес-процессов, выполняемых в организации, и дальнейшую работу с ними.

Так что же такое **процессный подход**?Обратимся к официальным определениям МС ИСО 9000:2000. Под процессом здесь понимается "совокупность взаимосвязанных и взаимодействующих видов деятельности, преобразующая входы в выходы, представляющие ценность для клиента". В определении МС ИСО 9000:2000 под процессом можно понимать любую деятельность, использующую определенные ресурсы (персонал, информацию, материальные ресурсы, инфраструктуру, технологии) и служащую для получения определенных выходов. Такое определение процесса является достаточно общим. Под него подпадает любое подразделение организации. Действительно, в каждом подразделении выполняются определенные работы, расходуются ресурсы, используется оборудование. На выходе подразделения получаем определенный результат: обработанные документы, готовую продукцию, услуги и т.д.

В связи с вышесказанным, в настоящее время у руководителей организаций и специалистов наблюдается путаница в определениях функций, бизнес-процессов и процессного подхода к управлению. Поэтому, прежде чем перейти к определению процессного подхода, мы рассмотрим наиболее распространенные заблуждения, касающиеся бизнес-процессов и процессного подхода к управлению, дадим определение процессного подхода и акцентируем внимание на ключевых аспектах его внедрения.

Так, например, преподаватель одного из ведущих высших учебных заведений России раскрывает свое понятие процессного подхода, которому учит студентов. Автор осознает "необходимость перехода от функционального подхода к управлению, функциональной декомпозиции (стандарт IDEF0, продукт BPWin) – к процессному (моделирование бизнес-процессов в ARIS), как это делают сейчас все". Приведенный тезис красноречиво говорит о непонимании автором процессного подхода. Во-первых, как BPWin, так и ARIS являются инструментальными средствами для описания деятельности организации. В основе обеих систем лежат свои нотации описания процессов. В BPWin – это IDEF0, IDEF3 и DFD, в ARIS – еЕРС (расширение IDEF3) и др. Все эти нотации служат исключительно как средства для описания деятельности организации. Ни в одну из этих систем не заложены принципы управления организацией на основе процессного подхода. В документации к системам они не сформулированы, не даны четкие методики внедрения процессного подхода к управлению и т.д. Многие заблуждаются, говоря об ARIS как о новом, современном процессном подходе, а о BPWin – как об устаревшем функциональном. Подобные рассуждения показывают некомпетентность в данной области. Такая ситуация, к сожалению, во многом сложилась вследствие неадекватной деятельности российских и зарубежных консалтинговых компаний.

Определение бизнес-процесса

Рассмотрим основные определения, которые нам понадобятся в дальнейшем:

**Бизнес-процесс** – целенаправленная последовательность процедур, предназначенная для получения заданного конечного результата.

**Процессный подход** – применение для управления деятельностью и ресурсами организации системы взаимосвязанных процессов.

**Сеть бизнес-процессов организации** – совокупность взаимосвязанных и взаимодействующих бизнес-процессов, включающих все функции, выполняемые в подразделениях организации.

**Автоматизация бизнес-процессов** – реализация системы управления бизнес-процессами с помощью информационных технологий.

**Workflow** – это система обеспечения выполнения задач, поставленных перед исполнителями в рамках процессного управления. Т.е. **Workflow** – система, координирующая выполнение операций, составляющих бизнес-процессы.

Понятие бизнес-процесса является центральным в рамках **Процессного подхода к управлению**, который приобретает все большую популярность в последнее время. Данный подход рассматривает организацию как систему взаимосвязанных процессов и при грамотном применении предоставляет компании неоценимые преимущества, такие как возможность повышения прозрачности структуры организации, выделение зон ответственности и создание системы мотивации сотрудников.

**Автоматизация бизнес-процессов**осуществляется с помощью проекта по внедрению информационной системы, позволяющей автоматизировать не только учетные функции, но и бизнес-логику, вопросы взаимодействия между сотрудниками, выполняющими разные функции в рамках единого бизнес-процесса. Автоматизация бизнес-процессов обязательно должна решать вопросы взаимодействия.

Автоматизация бизнес-процессов в системах класса ERP может быть построена на методологии процессного управления и информационной подсистеме **workflow**, позволяющей обеспечивать четкую и бесперебойную передачу рабочей информации между исполнителями процессов. Информация должна быть передана вовремя, с известным предыдущим результатом и в полном объеме. Пути передачи информации и ее протекания должны настраиваться архитектором (дизайнером) бизнес-процессов. Также система должна иметь возможность автоматически выбирать путь протекания в зависимости от выставленных пользователем параметров процесса или учетных данных в системе.

Другим вариантом данного подхода могут являться продукты класса [BPMS](http://bpms.ru/intro/index.html). Примером такой системы является [ELMA BPM](https://www.elma-bpm.ru). В совокупности с учетной системой, например, "1С: Управление производственным предприятием", данный продукт позволяет выстроить (смоделировать) управление процессами любого производственного или торгового предприятия, и автоматизировать эти процессы.

Основой для определения бизнес-процессов является деятельность, выполняемая подразделениями организации. Важно понимать, что бизнес-процесс является объектом управления и его представление в виде графической схемы является упрощенным, ограниченным по возможностям представлением.

**Методология создания модели бизнес-процесса (нотация описания)** – совокупность способов, при помощи которых объекты реального мира (например, деятельность организации) и связи между ними представляются в виде модели.

Любая методология (методика) включает три основных составляющих:

1. Теоретическая база.
2. Описание шагов, необходимых для получения заданного результата.
3. Рекомендации по использованию как отдельно, так и в составе группы методик.

Если в основу методологии заложена теоретическая база (например, математическая модель, способная предсказать поведение системы), то ее наличие делает методологию более серьезной и обоснованной. Однако и в случае отсутствия теории (отсутствия математической модели) методологии также могут успешно применяться. Основное в методологии – дать пользователю практическую последовательность шагов, которые приводят к заданному результату. Именно способность получать результат с заданными параметрами характеризует эффективность методологии. Методологии могут использоваться как отдельно, так и в ряду других методологий. Примером такой ситуации может быть проект реорганизации бизнес-процессов, когда методология создания моделей процессов является одной из применяемых методологий: управления проектом, анализа процессов, управления качеством и т.д.

Выделяют три основных способа описания бизнес-процессов:

1. Текстовое.
2. Табличное.
3. Графическое.

Первые два из них используются в настоящее время все реже, поскольку существенно уступают графическому представлению. Оно является наиболее простым для восприятия из всех трех, но в то же время позволяет анализировать бизнес-процессы с целью их последующей оптимизации, а в этом и заключается предназначение описания бизнес-процессов.

Коротко рассмотрим историю развития методологий моделирования бизнес-процессов.

Основой методологий IDEF0 и IDEF3, используемых в настоящее время для моделирования бизнес-процессов, явились методология SADT и алгоритмические языки, использовавшиеся для разработки программного обеспечения. Методология SADT была разработана частной американской корпорацией и затем в рамках программы Министерства обороны США была преобразована в методологию IDEF0, утвержденную как федеральный стандарт США. Появление методологии IDEF0 было предопределено тенденциями развития вычислительных средств – мощных машин (Mainframe) и появлением подходов MRP. Планирование материальных ресурсов для обеспечения производства (подход MRP) требовал выполнения сложных, многовариантных расчетов по обеспечению организации материальными ресурсами для производства готовой продукции. Использование подхода MRP, попытки автоматизации производства при помощи вычислительных машин привели к необходимости описывать деятельность организаций еще на стадии проектирования систем. Кроме того, задачи создания сложных систем управления (в том числе военного назначения) требовали соответствующих инструментов разработки. Необходимость создания методологий моделирования процессов была обусловлена практической необходимостью. Для моделирования деятельности организаций на верхнем уровне использовалась методология SADT, затем IDEF0. С начала 70-х годов ничего принципиально лучшего, чем IDEF0 для описания процессов на верхнем уровне, на наш взгляд, не было предложено. Исключение составляет подход UML (Unified Modeling Language – универсальный язык моделирования), но он предназначен для моделирования работы объектно-ориентированного программного обеспечения, а не бизнес-процессов организации.

После появления персональных компьютеров стали разрабатываться различные инструментальные средства (программные продукты) для моделирования бизнес-процессов. Кроме средств моделирования процессов, активно развивалось направление моделирования данных. Появлялись программные средства, в основном ориентированные на разработку моделей данных организаций и настройку промышленных баз данных. Такие программные продукты получили название CASE-систем. Среди наиболее известных продуктов для моделирования бизнес-процессов можно назвать Design/IDEF, BPWin, CASE-аналитик (в России), Silverrun, Designer-2000 и т.д.

В настоящее время на рынке присутствует несколько методологий. Часть из них основана на государственных стандартах, часть – на корпоративных разработках компаний, часть – выдвинута отдельными авторами. Целесообразно классифицировать существующие методологии по трем категориям:

1. Методологии ведения проекта.
2. Методологии моделирования и анализа бизнес-процессов.
3. Методологии использования программных продуктов для моделирования бизнес-процессов в проекте.

Последовательно рассмотрим каждую из трех групп методологий.

В настоящее время существует несколько достаточно четко идентифицируемых методологий ведения проектов, связанных с изменением бизнес-процессов, существующих в организации. Одним из известных подходов, является методология Хаммера и Чампи, известная как "реинжиниринг бизнес-процессов". Реинжиниринг, по Хаммеру и Чампи, – это "фундаментальное переосмысление и радикальное перепроектирование деловых процессов для достижения резких, скачкообразных улучшений в решающих, современных показателях деятельности компании, таких как стоимость, сервис и темпы". Основой указанного подхода является рассмотрение деятельности организации "с чистого листа" и разработка новых, более эффективных бизнес-процессов. Методология Хаммера и Чампи развивается уже более 10 лет. Из аналитических материалов зарубежной прессы известно, что 80–90% проектов, заявленных как проекты реинжиниринга бизнес-процессов, потерпели неудачу. На наш взгляд, проблемы здесь следует искать не в самой методологии Хаммера и Чампи, а в способах управления организацией, в частности, в заинтересованности руководителей верхнего уровня и их активном участии в проекте. По нашему мнению, для сегодняшнего момента можно было бы переформулировать определение реинжиниринга бизнес-процессов как деятельность, основанную на представлении организации в виде ряда взаимосвязанных бизнес-процессов и направленную на их регулярный анализ и улучшение.

Кроме методологии Хаммера и Чампи, существуют и другие методологии, не имеющие однозначного авторства, но принадлежащие отдельным компаниям, например, методологии выполнения проектов по внедрению систем автоматизации Oracle, SAP R/3, BAAN, RUP компании Rational и др.

Из последних следует отметить методологии, предлагаемые к всеобщему использованию в виде международных стандартов, как например МС ИСО 9000:2000. Заметим, что в нем регламентированы требования к системе менеджмента качества. Использование этого стандарта в качестве руководства по внедрению процессного подхода требует его квалифицированной интерпретации и конкретизации.

К второй группе методологий относятся методологии моделирования и анализа бизнес-процессов. В настоящее время существует несколько базовых способов описания процессов, основанных как на стандартах (IDEF0), так и на общепринятых подходах (DFD). Кроме того, существует ряд нотаций (методологий) описания процессов, предложенных отдельными компаниями – разработчиками программных продуктов. К числу последних относятся методологии ARIS (еЕРС) компании IDS Scheer AG, Германия.

К третьей группе методологий относятся методологии использования программных продуктов для создания моделей бизнес-процессов. Этот вариант мы и будем рассматривать в рамках нашего курса на примере современной нотации BPMN и использование её в системе ELMA. Первые две группы методологий, это просто перенос процессов на бумагу. Реальная польза компаниям идет лишь от исполняемых бизнес-процессах, которые работают. Следует отметить, что знать нотацию и уметь ее эффективно использовать на практике – далеко не одно и то же. Современные средства моделирования настолько сложны в применении, что требуют разработки специальных методик их применения в проекте.

Резюмируя обсуждение методологий, следует отметить, что методологии являются не самоцелью, а лишь средством достижения целей проекта.

Итак, **Моделирование** – это процесс отражения реальной деятельности организации при помощи специальной методологии. Важно понимать, что процесс моделирования является субъективным. Дело в том, что 80% информации для формирования моделей поступает от интервьюируемых сотрудников и руководителей организации. При этом субъективными являются как мнение сотрудников о реальном ходе работ, так и взгляд на процессы аналитика, проводившего интервью. Опыт показывает, что степень субъективности полученных моделей может стать серьезным препятствием для дальнейшего их использования.

**Модель "как есть"** (от "asis" – англ.) – это модель бизнес-процесса, построенная на основе субъективного видения бизнес-процесса, существующего в организации. При построении модели "как есть" важно помнить, во-первых, о субъективности, во-вторых, об актуальности модели. Дело в том, что в крупных организациях постоянно происходят изменения. Модель процессов может стать неактуальной (несоответствующей) уже через несколько месяцев после ее создания. Поэтому описание процесса должно использоваться в рабочих документах по процессу, постоянно подвергаться корректировке с целью обеспечения соответствия реальной деятельности. К сожалению, специалисты, привлекаемые для работ по описанию процессов, считают, что модели процессов ценны сами по себе и содержат информацию по улучшению деятельности.

Важнейшим понятием является эффективность бизнес-процесса. Под **эффективностью бизнес-процесса**, как правило, понимается отношение конечного результата (выхода) процесса к затраченным на его получение ресурсам. Эффективность может измеряться на основе различных показателей. Для каждого показателя могут быть рассчитаны допустимые или целевые значения – критерии. Важно отметить, что на практике недостаточно определить перечень показателей оценки эффективности бизнес-процесса. Важно разработать также методику их измерения. При этом методики, по возможности, не должны изменяться, так как результат измерения используется для оценки сегодняшнего состояния дел по сравнению с предыдущими периодами. При изменении методик измерения и расчета показателей меняется результат и, соответственно, гораздо труднее реально оценить ход бизнес-процесса.

### Классификация и взаимосвязь процессов в бизнесе

При выделении процессов в любой организации возникает ряд вопросов:

* Какие процессы должны быть в организации? Список обязательных процессов?
* Сколько процессов должно быть в организации?
* Кто такой "владелец" процесса, каковы его права и обязанности?
* Как обеспечить взаимосвязь процессов в единую сеть?

Существуют некоторые общие правила выделения процессов в организации. По характеру деятельности и отношению к создаваемому продукту, процессы организации могут быть разделены на четыре основных типа охватывающих:

1. Основные бизнес-процессы.
2. Вспомогательные бизнес-процессы.
3. Бизнес-процессы управления.
4. Бизнес-процессы развития.

К **основным процессам** организации, как правило, относятся процессы производства, сбыта и снабжения. Примеры основных процессов: маркетинг, закупки, производство, хранение, поставка продукции, сервисное обслуживание и другие, связанные с продукцией.

**Вспомогательные процессы** напрямую не добавляют стоимости и являются по своей сути затратными. Примеры: подготовка кадров, сервисное обслуживание оборудования, обеспечение связью, IT-обеспечение, административно-хозяйственное обеспечение, финансовое и бухгалтерское обеспечение деятельности организации, обеспечение безопасности, другие процессы. Критерий выделения вспомогательного процесса – использование результатов этого процесса многими функциональными подразделениями и процессами.

При выделении **процесса управления** деятельностью необходимо измерение его результативности и эффективности. Для принятия решения по вопросам деятельности в масштабах организации генеральный директор часто создает аппарат управления (отдел стратегического развития, планово-экономический отдел, административный отдел), который подчиняется ему напрямую и должен быть учтен при выделении процесса.

Характерные признаки этих категорий процессов приведены в Таблица 1.

Таблица 1. Классификация процессов предприятия

| **Типы процессов** | **Характерные признаки** | **Клиенты** |
| --- | --- | --- |
| Основные процессы (процессы основной деятельности) | * назначение процессов – создание основных продуктов; * результат – основной продукт или полуфабрикат для его изготовления; * процессы лежат на пути создания основных продуктов; * процессы добавляют продукту ценность для потребителя. | * внешние клиенты; * конечные потребители; * внутренние клиенты – другие процессы организации. |
| Вспомогательные процессы | * + назначение процессов – обеспечение деятельности основных процессов;   + результат – ресурсы для основных процессов;   + деятельность процессов не касается основных продуктов; * процессы добавляют продукту стоимость. | Внутренние клиенты – другие процессы организации. |
| Процессы управления организацией | * назначение процесса – управление деятельностью всей организации; * результат – деятельность всей организации. | * собственники (инвесторы); * потребители (клиенты); * персонал (сотрудники); * поставщики и субподрядчики; * общество (внешняя среда). |

Графическое отображение классификации процессов изображено на Рис. 2.

Изображение выглядит как текст, снимок экрана, диаграмма, Прямоугольник

Автоматически созданное описание

Рис. 2. Графическое отображение классификации процессов

**Сетью бизнес-процессов** организации называется совокупность взаимосвязанных и взаимодействующих бизнес-процессов, включающих все функции, выполняемые в подразделениях организации.

Сеть процессов является результатом, который может быть использован для дальнейших работ по улучшению системы управления организации, например, внедрению системы менеджмента качества, соответствующей требованиям МС ИСО 9001:2000. Возможно и другое использование сети процессов – выбор приоритетных процессов для организации, детальное моделирование и анализ выбранных процессов, последующая реорганизация (реинжиниринг) процессов. Важно то, что как руководители, так и исполнители имеют комплексную картину процессов организации и могут принимать адекватные решения.

Вопрос 3. Архитектура предприятия: понятия, эволюция подходов, компоненты архитектуры.

**Архитектура предприятия** – это структура и организация бизнес-процессов, информационных систем, технологий и ресурсов предприятия, которые взаимодействуют между собой для достижения стратегических целей и обеспечения эффективной работы организации. Она описывает на уровне предприятия в целом то, как реализуются основные функции организации, включая организационные и функциональные структуры, роли и ответственности.

Архитектура предприятия **определяет**, как различные компоненты предприятия взаимодействуют друг с другом, какие данные и информация используются, какие технологии применяются, какие бизнес-процессы выполняются и как они связаны с целями и стратегией предприятия.

Архитектура предприятия **включает в себя** не только информационные системы и технологии, но также охватывает организационную структуру, бизнес-процессы, роли и обязанности сотрудников, а также взаимодействие с внешними стейкхолдерами, такими как клиенты, партнеры и поставщики.

**Цель архитектуры предприятия** – обеспечить согласованность, эффективность и гибкость работы предприятия, а также обеспечить его способность адаптироваться к изменяющимся условиям рынка и технологий.

В ранних работах ИТ-архитектура **понималась в основном как технологическая архитектура** или архитектура, определяющая инфраструктуру информационной системы. Работы по описанию архитектуры сводились к рассмотрению исключительно технических аспектов. Такой подход хоть и позволяет добиться определенных преимуществ, например, снизить стоимость закупок и эксплуатации информационных систем и уменьшить затраты на разработку приложений и обучение персонала. Однако он является заведомо ограниченным, так как не подразумевает ориентацию на решение бизнес-задач, которые как известно могут меняться весьма кардинально, требуя от организации большой гибкости в части архитектуры (в том числе и в сфере информационных технологий).

Следующей ступенью развития оказалось понятие корпоративной информационно-технологической архитектуры масштаба предприятия (EWITA - Enterprise-Wide Information Technology Architecture). Даже название уже указывает на то, что она **сочетает в себе как технологические, так и аспекты, связанные с данными**. Основное направление работ при этом состоит в совместном использовании общих данных, исключении дублирования бизнес-функций, координации управления пользователями, ресурсами, информационной безопасностью за счет улучшений в управлении портфелем прикладных систем. Архитектура данного этапа эволюции **описывает то, как компоненты информационной системы связаны между собой**; точно так же бизнес-архитектура описывает то, как элементы бизнеса связаны между собой. Такой подход обеспечивает более эффективное взаимодействие различных структурных подразделений организации за счет совместного доступа к информации различных подразделений, а также внешних организаций (клиентов, партнеров, поставщиков, т. н. «микросреды бизнеса»). Получаемые преимущества относятся прежде всего к оперативной деятельности организации.

Очевидным логическим продолжением данной траектории развития явилось **добавление к технологическим и информационным аспектам описания бизнеса организации**. В результате появилось понятие архитектуры предприятия, которая объединяет корпоративную ИТ-архитектуру масштаба предприятия с бизнес-архитектурой и позволяет обеспечить переход на стратегический уровень управления.

В реальности все три аспекта представляют собой лишь взгляды с разных точек зрения на одну и ту же организацию, при этом они связаны между собой неразрывно. Значит совершенно логично, что они должны быть объединены в одну модель, которая бы учитывала и вполне понятно показывала, как связаны друг с другом все элементы ведения бизнеса, в том числе и связанные с информационными технологиями. Особенно стоит отметить, что с точки зрения современных архитектурных подходов, нет особой разницы, какую именно организацию рассматривать, коммерческую или нет. Это достигается за счет того, что бизнес-архитектура обеспечивает гибкость моделирования, поскольку включает в себя описание миссии и целей рассматриваемой организации.

Сама концепция архитектуры предприятия явилась результатом поиска некоторого целостного подхода, который обеспечил бы «взгляд на организацию в целом», с учетом всех возможных измерений, хотя учет большего количества измерений предполагает и рост сложности представлений об архитектуре

Выделяют следующий набор компонентов архитектуры.

* Двигатели архитектуры (Architecture Drivers) отражают внешние стимулы изменения архитектуры: бизнес-стимулы и технические стимулы. В качестве бизнес–стимулов может выступать новое законодательство, новые инициативы администрации, ассигнования для ускорения развития отдельных сфер, рыночные силы. В роли технических двигателей могут выступать новое и улучшенное программное обеспечение, аппаратные средства ЭВМ и их комбинации.
* Стратегическое направление (Strategic Direction) - руководство для разработки целевой архитектуры, которое содержит видение миссии предприятия, принципы его построения, цели и объекты предприятия.
* Текущая архитектура (Carrent Architecture) определяет архитектуры предприятия "как есть" и состоит из двух частей: текущая бизнес-архитектура и техническая архитектура (данные, приложения и технологии). Она отражает текущие возможности и технологии, а также служит объектом для дальнейшего расширения.
* Целевая архитектура (Target Architecture) определяет архитектуру предприятия "как должно быть построено" и состоит из двух частей: целевая бизнес-архитектура и техническая архитектура (т.е. данные, приложения и технологии). Она представляет будущие возможности и технологии, которые являются результатом улучшения проекта поддержки изменяющихся бизнес - потребностей.
* Переходные процессы (Transitional Processes) поддерживают переход от текущей архитектуры к целевой архитектуре. Критические переходные процессы для предприятия включают планирование инвестиций в сферу ИТ, планирование перехода, управление конфигурацией, контроль и управление проектом.
* Архитектурные сегменты (Architectural Segments) отражают ориентацию отдельных частей общей архитектуры на главные бизнес - области.
* Архитектурные модели (Architectural Models) определяют бизнес - модели и конструкторские (технические) модели, которые отражают все необходимые сегменты для полного описания предприятия.
* Стандарты (Standards) включают все стандарты, руководящие принципы (руководящие материалы), а также передовой опыт. Примерами стандартов являются:
  + стандарты безопасности;
  + стандарты данных относятся к данным, метаданным и другим связанным структурам;
  + стандарты приложений относятся к прикладному ПО;
  + стандарты технологий относятся к операционным системам и аппаратным платформам.