**Лекция 2. Технологии проектирования ИС**

**2.1 Основные определения**

Современные информационные технологии предоставляют широкий набор способов реализации ИС. выбор которых осуществляется на основе требований со стороны предполагаемых пользователей, которые, как правило, изменяются в процессе разработки.

Под проектом будем понимать про ектно-конструкторскую и технологическую документацию, в которой представлено описание проектных решений по созданию и эксплуатации ИС в конкретной программно-технической среде.

Под проектированием ИС понимается процесс преобразования исходной информации об объекте, методах и опыте проектирования объектов аналогичного назначения в соответствии с ГОСТом в проект ИС. С этой точки зрения проектирование ИС сводится к последовательной формализации проектных решений на различных стадиях жизненного цикла ИС: планирования и анализа требований, технического и рабочего проектирования, внедрения и эксплуатации.

Объектами проектирования ИС являются отдельные элементы или комплексы
функциональных и обеспечивающих частей. Так. функциональными элементами в соответствии с традиционной декомпозицией выступают задачи, комплексы задач и функции управления. В составе обеспечивающей части ИС объектами проектирования служат элементы и комплексы информационного, программного, технического и других видов обеспечения системы.

В качестве субъекта проектирования И С выступают коллективы специалистов, которые осуществляют проектную деятельность, как правило, в составе специализизированной (проектной) организации, и организация - заказчик, для которой необходимо разработать ИС. Масштабы разрабатываемых систем определяют состав и количество участников процесса проектирования. При большом объеме и жестких сроках выполнения проектных работ в разработке системы может принимать участие несколько проектных коллективов (орга-
низаций - разработчиков). В этом случае выделяется головная организация, которая координирует деятельность всех организаций-соисполнителей.

Форма участия соисполнителей в разработке проекта системы может быть различной. Наиболее распространенной является форма, при которой каждый соисполнитель выполняет проектные работы от начала до конца для какой-либо части разрабатываемой системы. Обычно это бывает функциональная подсистема или взаимосвязанный комплекс задач управления. Реже встречается форма, при которой некоторые соисполнители выполняют работы на отдельных этапах процесса проектирования. Возможен вариант, при котором функ-
ции заказчика и разработчика совмещаются, т. е. АИС проектируется собственными силами.

Осуществление проектирования ИС предполагает использование проектировщиками определенной технологии проектирования, соответствующей масштабу и особенностям разрабатываемого проекта.

Технология проектирования ИС — это совокупность методологии и средств проектирования ИС. а также методов и средств его организации (управление процессом создания и модернизации проекта ИС).

В основе технологии проектирования лежит технологический процесс, который определяет действия, их последовательность, требуемые состав исполнителей, средства и ресурсы.

Технологический процесс проектирования ИС в целом делится на совокупность последовательно-параллельных, связанных и соподчиненных цепочек действий, каждое из которых может иметь свой предмет. Действия, которые выполняются при проектировании ИС. Могут быть определены как неделимые технологические операции или как подпроцессы технологических операций. Все действия делятся на собственно проектировочные, которые формируют или модифицируют результаты проектирования, и оценочные действия, которые вырабатывают по установленным критериям оценку результатов проектирования.

Таким образом, технология проектирования задается регламентированной последовательностью технологических операций, выполняемых на основе того или иного метода, в результате чего становится ясным, не только что должно быть сделано для создания проекта, но и как, кем и в какой последовательности.

Предметом любой выбираемой технологии проектирования должно служить отражение взаимосвязанных процессов проектирования на всех стадиях жизненного цикла ИС. К основным требованиям, предъявляемым к выбираемой технологии проектирования, относятся следующие:

• созданный проект должен отвечать требованиям заказчика;

• максимальное отражение всех этапов жизненного цикла проекта;

• обеспечение минимальных трудовых и стоимостных затрат на проектирование и сопровождение проекта;

• технология должна быть основой связи между проектированием и сопровождением проекта;

• рост производительности труда проектировщика;

• надежность процесса проектирования и эксплуатации проекта;

• простое ведение проектной документации.

Основу технологии проектирования ИС составляет методология, которая определяет сущность, основные отличительные технологические особенности.

Методология проектирования предполагает наличие некоторой концепции, принци
пов проектирования, реализуемых набором методов, которые, в свою очередь, должны поддерживаться некоторыми средствами.

Организация проектирования предполагает определение методов взаимодействия проектировщиков между собой и с заказчиком в процессе создания проекта ИС. которые могут также поддерживаться набором специфических средств.

**2.2 Классификация методов проектирования систем**

Методы проектирования ИС можно классифицировать по степени использования
средств автоматизации, типовых проектных решений, адаптивности к предполагаемым изменениям.

Так. **по степени автоматизации** методы проектирования разделяются на:

**ручное**, при котором проектирование компонентов ИС осуществляется без использования специальных инструментальных программных средств, а программирование — на алгоритмических языках;

**компьютерное**, при котором производится генерация или конфигурирование (настройка) проектных решений на основе использования специальных инструментальных программных средств.

**По степени использования типовых проектных решений** различают следующие методы проектирования:

оригинальное (индивидуальное), когда проектные решения разрабатываются «с нуля» в соответствии с требованиями к АИС. Характеризуется тем. что все виды проектных работ ориентированы на создание индивидуальных для каждого объекта проектов, которые в максимальной степени отражают все его особенности;

типовое, предполагающее конфигурирование ИС из готовых типовых проектных решений (программных модулей). Выполняется на основе опыта, полученного при разработке индивидуальных проектов. Типовые проекты, как обобщение опыта для некоторых групп организационно-экономических систем или видов работ, в каждом конкретном случае связаны со множеством специфических особенностей и различаются по степени охвата функций управления, выполняемым работам и разрабатываемой проектной документации.

**По степени адаптивности проектных решений** выделяют методы:

**реконструкции**, когда адаптация проектных решений выполняется путем переработки соответствующих компонентов (перепрограммирования программных модулей);

**параметризации**, когда проектные решения настраиваются (генерируются) в соответствии с изменяемыми параметрами;

**реструктуризации** модели, когда изменяется модель проблемной области, на основе которой автоматически заново генерируются проектные решения.

Сочетание различных признаков классификации методов обусловливает характер используемых технологий проектирования ИС, среди которых выделяют два основных класса: каноническую и индустриальную технологии.

**Индустриальная технология** проектирования, в свою очередь, разбивается на два подкласса: автоматизированное (использование CASE-технологий) и типовое (параметрически-ориентированное или модельно-ориентированное) проектирование. Использование индустриальных технологий не исключает использования в отдельных случаях канонических.

Для конкретных видов технологий проектирования свойственно применение определенных средств разработки ИС, которые поддерживают выполнение как отдельных проектных работ, этапов, так и их совокупностей. Поэтому перед разработчиками ИС, как правило, стоит задача выбора средств проектирования, которые по своим характеристикам в наибольшей степени соответствуют требованиям конкретного предприятия.

Таблица 2.1 Характеристики Kлaccoв технологий проектирования

|  |  |  |  |
| --- | --- | --- | --- |
| Класстехнологии | Степеньавтоматизации | Степеньтипизации | Степеньадаптивности |
| Каноническое | Ручное | Оригинальное | Реконструкция |
| Индустриальноеавтоматизированное | Компьютерное | Оригинальное | Реструктуризация модели(генерация ИС) |
| Индустриальноетиповое | Компьютерное | Типовое сборочное | Параметризация и рест-руктуризация модели(конфигурация ИС) |

Средства проектирования должны быть:

• инвариантны к объекту проектирования (в своем классе);

• охватывать в совокупности все этапы жизненного цикла ИС;

• технически, программно и информационно совместимыми;

• простыми в освоении и применении;

• экономически целесообразными.

Средства проектирования ИС можно разделить на два класса: без использования ЭВМ, с использованием ЭВМ.

**Средства проектирования без использования ЭВМ** применяются на всех стадиях и этапах. Как правило, это средства организационно-методического обеспечения операций и в первую очередь различные стандарты, регламентирующие процесс проектирования систем. Сюда же относятся единая система классификации и кодирования информации, унифицированная система документации, модели описания и анализа потоков информации и т. п.

**Средства проектирования с использованием ЭВМ** могут применяться как на отдельных, так и на всех стадиях и этапах процесса проектирования ИС и соответственно поддерживают разработку документов, разделов, проекта системы в целом. Все множество средств проектирования с использованием ЭВМ делят на четыре класса.

**1. Операционные средства**, которые поддерживают проектирование операций обработки информации. К данному подклассу средств относятся алгоритмические языки, библиотеки стандартных программ и классов объектов, макрогенераторы, генераторы программ типовых операций обработки данных и т. п.. а также средства расширения функций операционных систем (утилиты). В данный класс включаются также такие простейшие инструментальные средства проектирования, как средства для тестирования и отладки программ, поддержки процесса документирования проекта и т. п. Особенность последних программ заключается в
том. что с их помощью повышается производительность труда проектировщиков, но не разрабатывается законченное проектное решение.

Таким образом, средства данного подкласса поддерживают отдельные операции проектирования ИС и могут применяться независимо друг от друга.

**2. Средства, поддерживающие проектирование отдельных компонентов**. К данному подклассу относятся средства общесистемного назначения:

• системы управления базами данных (СУБД);

• методо-ориентированные пакеты прикладных программ (решение задач дискретного программирования, математической статистики и т. п.);

• табличные процессоры;

• статистические ППП;

• оболочки экспертных систем;

• графические редакторы;

• текстовые редакторы;

• интегрированные ППП (интерактивная среда с встроенными диалоговыми возможностями, позволяющая интегрировать, вышеперечисленные программные средства).

Для перечисленных средств характерно их использование для. разработки технологических подсистем ИС: ввода информации, организации хранения и доступа к данным, вычислений, анализа и отображения данных, принятия решений.

**3. Средства, поддерживающие проектирование разделов проекта**. В этом подклассе выделяют функциональные средства проектирования.

Функциональные средства направлены на разработку автоматизированных систем, реализующих функции, комплексы задач и задачи управления. Разнообразие предметных областей порождает многообразие средств данного подкласса, ориентированных на тип организационной системы (промышленная, непромышленная сферы), уровень управления (например, предприятие, цех. отдел, участок, рабочее место), функцию управления (планирование, учет и т.п.).

К функциональным средствам проектирования систем обработки информации относятся типовые проектные решения, функциональные пакеты прикладных программ, типовые проекты

**4. Средства, поддерживающие разработку** на стадиях и этапах процесса проекти-
рования. К данному классу относятся средства автоматизации проектирования ИС (CASE-средства).

Современные CASE-средства. в свою очередь, классифицируются в основном по двум признакам:

1) по охватываемым этапам процесса разработки ИС;

2) по степени интегрированности:

• отдельные локальные средства (tools);

• набор неинтегрированных средств, охватывающих большинство этапов разработки ИС (toolkit);

• полностью интегрированные средства, связанные общей базой проектных данных

— репозиторием (workbench).

**Лекция 3. СОЗДАНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ**

Когда организация изменяет свою ИС, в ее разработке, внедрении и использовании
приходится участвовать специалистам различного профиля. В этом разделе обсуждается: какие этапы проходит ИС в своем становлении, что должно быть сделано на каждом этапе?

**3.1. Жизненный цикл разработки ИС**

Одним из базовых понятий методологии проектирования ИС является понятие жизненного цикла разработки информационных систем (ЖЦ ИС). ЖЦ ИС - это непрерывный процесс, который начинается с момента принятия решения о необходимости ее создания и заканчивается в момент ее полного изъятия из эксплуатации.

Основным нормативным документом, регламентирующим ЖЦ ПО. является международный стандарт ISO/TEC 12207 (ISO - International Organization of Standardization - Международная организация по стандартизации. IEC - International Electrotechnical Commission - Международная комиссия по электротехнике). Он определяет структуру ЖЦ, содержащую процессы, действия и задачи, которые должны быть выполнены во время создания ИС.

Структура ЖЦ ПО по стандарту ISO/TEC 12207 базируется на трех группах процессов:

• основных процессах ЖЦ ИС (приобретение, поставка, разработка, эксплуатация, сопровождение);

• вспомогательных процессах, обеспечивающих выполнение основных процессов (документирование, управление конфигурацией, обеспечение качества, верификация, аттестация, оценка, аудит, решение проблем);

• организационных процессах (управление проектами, создание инфраструктуры проекта, определение, оценка и улучшение самого ЖЦ. обучение).

**Разработка** включает в себя все работы по созданию ИС и его компонентов в соответствии с заданными требованиями, включая оформление проектной и эксплуатационной документации, подготовку материалов, необходимых для проверки работоспособности и соответствующего качества программных продуктов, материалов, необходимых для организации обучения персонала и т.д. Разработка ИС включает в себя, как правило, анализ, проектирование и реализацию (программирование).

**Эксплуатация** включает в себя работы по внедрению компонентов ИС в эксплуатацию, в том числе конфигурирование базы данных и рабочих мест пользователей, обеспечение эксплуатационной документацией, проведение обучения персонала, и т.д. и непосредственно эксплуатацию, в том числе локализацию проблем и устранение причин их возникновения, модификацию ИС в рамках установленного регламента, подготовку предложений по совер-шенствованию, развитию и модернизации системы.

**Управление проектом** связано с вопросами планирования и организации работ, создания коллективов разработчиков и контроля за сроками и качеством выполняемых работ. **Техническое и организационное обеспечение проекта** включает выбор методов и инструментальных средств для реализации проекта, определение методов описания промежуточных состояний разработки, разработку методов и средств испытаний ИС. обучение персонала и т.п.
**Обеспечение качества** проекта связано с проблемами верификации, проверки и тестирования ИС. **Верификация** - это процесс определения того, отвечает ли текущее состояние разработки, достигнутое на данном этапе, требованиям этого этапа. **Проверка** позволяет оценить соответствие параметров разработки с исходными требованиями. Проверка частично совпадает с тестированием, которое связано с идентификацией различий между действительными и ожидаемыми результатами и оценкой соответствия характеристик ПО (программного обеспечения) исходным требованиям. В процессе реализации проекта важное место занимают вопросы идентификации, описания и контроля конфигурации отдельных компонентов и всей системы в целом.

**Управление конфигурацией** является одним из вспомогательных процессов, поддерживающих основные процессы жизненного цикла ИС. прежде всего процессы разработки и сопровождения ИС. При создании проектов сложных ИС. состоящих из многих компонентов, каждый из которых может иметь разновидности или версии, возникает проблема учета их связей и функций, создания унифицированной структуры и обеспечения развития всей си-
стемы. Управление конфигурацией позволяет организовать, систематически учитывать и контролировать внесение изменений в ИС на всех стадиях ЖЦ.

Каждый процесс характеризуется определенными задачами и методами их решения, исходными данными, полученными на предыдушем этапе, и результатами. Результатами анализа в частности являются функциональные модели, информационные модели и соответствующие им диаграммы. ЖЦ ИС носит итерационный характер: результаты очередного этапа часто вызывают изменения в проектных решениях, выработанных на более ранних этапах.

Изменения в ИС могут колебаться от незначительных поправок до установки полностью новой системы. Независимо от масштабов изменений большинство компаний проходит через **жизненный цикл разработки ИС.**

**3.1.1. Причины изменения ИС в организациях**

Компании обычно меняют свои ИС по одной из следующих причин.

**Измененияе потребностей пользователей или предприятия**. Увеличение конкуренции, рост предприятия или объединение с другими, новые законы, изменения в локальных, региональных или глобальных взаимоотношениях могут изменить структуру и намерения организации. Чтобы соответствовать новым требованиям, ИС должна тоже измениться.

**Изменение технологии**. По мере того как технологии развиваются и становятся более дешевыми, у организации появляется возможность иметь систему, более отвечающую ее нуждам.

**Улучшение деловых процессов**. Многие компании устанавливают новые ИС, стремясь повысить эффективность своей деятельности. Например, сократить время ожидания клиентов.

**Получение преимуществ в конкурентной борьбе**. Улучшение качества, увеличение количества и скорости выработки информации может привести к улучшениям в продукте или услуге или помочь сократить затраты.

**Прирост производительности**. Компьютеры автоматизируют многие канцелярские и рутинные работы. Экспертные системы дают многим людям доступ к знаниям экспертов.

**Рост**. Растущие компании нуждаются либо в переделке, либо в замене своих ИС.

**Сокращение**. Часто компании переходят от централизованных систем с большими ЭВМ к ПК, работающим в сети, чтобы получить преимущество в отношении це-
на/производительность.

**Улучшение качества**. Трудно улучшить качество продуктов и услуг без улучшений в системе, от которой компании получают данные, измеряющие и оценивающие качество. Чтобы что-то менять, надо знать, что именно имеет недостаточное качество.

**3.1.2. Принципы разработки системы**

**Привлечь владельца и пользователей к процессу разработки**. Этот принцип обязательно следует выполнять для успешной разработки системы. Лица, ответственные за разработку системы, должны уделять время владельцу и пользователям, настаивая на их участии в согласовании всех решений, с ними связанных. Поэтому методологии уменьшают риск, связанный с сокращением функциональности системы и ошибками.

**Использовать подход, направленный на разрешение проблем**. Методология является подходом к построению систем на основе разрешения проблем. Классический подход к проблемной ориентации описывается следующим образом:

• изучается и понимается проблема и содержание системы;

• определяются требования приемлемого решения;

• определяются варианты решений и выбирается лучшее;

• проектируется и применяется решение;

• наблюдаются и оцениваются воздействие решений и соответствующим образом
усовершенствуются решения.

Среди начинающих разработчиков систем имеется тенденция игнорировать или сокращать этапы решения проблемы, что приводит к неадекватному пониманию и решению задач.

**Установить этапы и виды деятельности**. Большинство методологий состоят из этапов. В классическом виде жизненный цикл состоит из четырех этапов: обзор системы, анализ системы, проектирование системы, применение системы. Пятый этап - поддержка системы усовершенствует полученную систему, повторяя предыщушие четыре этапа в меньшем объеме с целью улучшения и усовершенств ования системы. Этапы обычно делятся на действия и задачи, которые легче управляются и выполняются.

**Установить стандарты для согласования разработки и документации**. Стандарты разработки системы обычно описывают:

• виды работ;

• ответственности;

• руководство или требования по документированию;

• проверку качества.

**Рассматривать систему как инвестиционный проект**. Действительно, система представляет собой инвестирование средств. Когда рассматриваются направления инвестирования, выбор следует делать на основе выполнения анализа стоимость- эффективность нескольких возможных вариантов решения проблемы. Стоимость эффективность определяется как результат, полученный вычислением баланса между стоимостью разработки и функционирования системы и выгоды, полученной от системы.

**Не следует пренебрегать пересмотром масштаба системы**. Значительным преимуществом поэтапного подхода разработки системы является возможность пересматривать технико-экономическое обоснование. При длительных разработках своевременное прекращение проекта менее убыточно, нежели продолжение разработки и применение системы, не обеспечивающей необходимый уровень эффекта от внедрения. Большинству аналитиков не удается выдержать запланированную стоимость и график выполнения работ в случае увеличения
масштаба системы. В результате аналитики часто и без необходимости принимают на себя ответственность по превышению стоимости и корректировке графика работ.

В методологию процесса разработки встраиваются контрольные точки, в которых стоимость сравнивается с некоторой величиной, представляющей стоимостную границу возможного продолжения разработки. В каждой контрольной точке следует принимать решение о прекращении работ, пересмотре стоимости и графика работ, частичном сокращении масштаба системы или выделении дополнительных средств.

**Разделяй и властвуй.** Все системы являются частью больших систем, называемых суперсистемами. С другой стороны, любая система может рассматриваться состоящей из меньших частей, называемых подсистемами. В процессе разработки система разбивается (декомпозируется) на подсистемы с целью упрощения изучения декомпозированных проблем. В результате декомпозиции большой проблемы (системы) на более мелкие части (подсистемы) появляется возможность упростить процесс анализа и решения проблем.

**Проектировать систему с учетом ее развития и изменений**. Многие системные аналитики рассматривают только требования сегодняшнего дня в процессе разработки системы. "Энтропия" - термин, используемый учеными для описания естественного и неизбежного процесса распада всех систем. Таким образом, в процессе поддержки неизбежно наступает момент, когда эксплуатационные затраты на поддержку системы превосходят стоимость разработки новой или усов ершенствования действующей системы, и на смену устаревшей системы следует разрабатывать новую систему. Системы, которые разрабатываются только с
учетом требований сегодняшнего дня. трудно поддаются модификации в ответ на вновь возникающие требования системы. Современные инструментальные и технологические средства позволяют проектировать систему, которая может расти и развиваться вместе с расширением и изменением требований системы. В этой связи требование гибкости и адаптивности должны быть встроены в систему.

**3.1.3. Вспомогательные виды деятельности**

В процессе жизненного цикла разработки системы существуют виды деятельности, которые следует выполнять на протяжении всех этапов ЖЦ. К таким видам деятельности относятся:

сбор информации — процесс использования обследований, интервью, встреч, анкетирования, создания прототипов и других технологий для сбора информации о системе, ее требованиях и предпочтениях заинтересованных лиц;

документирование и презентации — средства взаимодействия различных категорий заинтересованных лиц. вовлеченных в процесс разработки системы. Документирование — деятельность, направленная на запись фактов и спецификаций системы. Презентация — вид деятельности, направленный на представление документации о системе, оформленной в интересах различных категорий заинтересованных лиц. Презентация может выполняться в виде со-
общения или представляться в виде записки. Управление версиями документации при разработке сложных систем становится критическим фактором, поскольку необходимо учитывать множество различных версий документации, находящихся в различных стадиях разработки. Так например, может возникнуть необходимость учета текущей применяемой версии системы, предыдущей, а возможно, и новой разрабатываемой версии;

**оценивание и измерение** — деятельность, направленная на повышение качества и производительности системы. Оценивание — вычисление примерного времени, трудоемкости, стоимостей и величины выгоды разрабатываемой системы, используемое для получения достоверных данных. Измерение направлено на получение и анализ оценок производительности и качества деятельности разработчика (и иногда стоимости). Некоторые аналитики опасаются высказывать оценки из-за неуверенности и неопределенности имеющихся данных.
Более опытные аналитики привлекают для оценивания собственный и коллективный опыт, основанный на выполнении предыдущих проектов, и постоянно усовершенствуют систему оценок. Измерения становятся особенно актуальными из-за серьезного влияния производительности и качества разработки проблем на разработку системы;

**технико-экономический анализ** представляет собой меру того, насколько выгодной и полезной в организации может быть разработка информационной системы;

**управление проектом и процессом разработки.** В проектирование информационной системы вовлекается коллектив сотрудников, состоящий из аналитиков, программистов, пользователей и других профессионалов в области информационных систем, работающих как одна команда. Управление проектом представляет собой постоянную деятельность, в процессе которой аналитик планирует, распределяет обязанности, направляет и контролирует вы-
полнение разработки системы в рамках выделенного времени и бюджета. Большинство проектов, которые заканчиваются неудачей, характеризуются недостаточным руководством и управлением. Плохое управление выражается невыполненными или неустановленными требованиями, превышением бюджета и задержками в разработке. Жизненный цикл разработки системы обеспечивает структурную основу для управления проектом системы. Управление процессом разработки имеет своей целью стандартизацию подхода и средств проектирова-
ния, а также способа распространения готовой системы. Управление процессом разработки - деятельность, устанавливающая стандарты для выполнения этапов жизненного цикла, состав методов и моделей проектирования, использование инструментальных средств, а также требования к содержанию и оформлению результатов, которые производятся в процессе проектирования.

**3.1.4. Содержание основных этапов жизненного никла**

Рассмотрим содержание основных этапов жизненного цикла разработки ИС. изображенных на рис. 3.1.

Суть этапа **анализа системы** - сбор информации, необходимой для приобретения или разработки новой системы.Разработка системы начинается с предварительного определения основных проблем и задач ИС. а также функций, с помощью которых разрешаются выявленные проблемы. Описание функций выполняется на языке производственных (описание процессов предметной области), функциональных (описание формы обрабатываемых документов) и технических (аппаратное, программное, лингвистическое обеспечение) требований. Во многих организациях в результате предварительного изучения ситуации готовится формальный запрос на проектирование и разработку ИС. Чтобы максимизировать использование
ограниченных ресурсов, требования к разработке системы ранжируются по степени важности. После начального этапа производится исследование используемой в данный момент ИС, чтобы определить ее природу, нацеленность и понять сильные и слабые стороны, определяются и документируются информационные потребности пользователей системы и менеджеров. Все это используется для разработки и документирования требований к системе. Далее требования к системе используются для выбора между разработкой и приобретением ИС. Руководству предоставляется отчет об анализе системы.



Рис. 3.1 Жизненный цикл разработки ИС

На этапе **концептуальной разработки** компания решает, как удовлетворить свои потребности. Первая задача - определить и оценить возможные альтернативы. Если одна из альтернатив принята руководством для разработки, то формируется план разработки ИС, содержащий следующие ключевые элементы:

• определение масштаба проекта, в рамках которого анализируются основные функции и взаимодействие с внешними системами;

• более полное изучение формулировки проблем;

• уточняются необходимые процессы обработки информации и ограничения;

• рассматривается ресурсное обеспечение разработки: люди, время, финансовые средства для разработки и функционирования ИС. Ресурсное обеспечение исчисляется по аналогии с известными разработками;

• готовится график выполнения различных этапов проекта.

В заключение этапа анализа готовятся детальные спецификации, описывающие логическую модель ИС: схемы и структуры данных всех уровней, модульность ИС. документацию логической структуры, скрипты для создания объектов БД.

**Физическая разработка** - это перевод общих, ориентированных на пользователя требований, сформулированных на этапе концептуальной разработки, в детальные спецификации, которые могут быть использованы при кодировании и тестировании компьютерных программ. Разрабатываются входные и выходные документы, пишутся компьютерные программы, создаются файлы, разрабатываются процедуры, в новую систему встраиваются способы
ее контроля. Разработанные компоненты тестируются и отлаживаются.

**Внедрение** - это этап, на котором соединяются воедино все элементы системы и начинается ее функционирование. Это очень ответственный и сложный этап, поэтому подготавливается и строго выполняется план внедрения. Как часть внедрения, устанавливается и тестируется все новое оборудование и программы, нанимается или обучается персонал, тестируются (и возможно корректируются) новые процедуры обработки данных, отрабатываются стандарты и способы контроля новой ИС, делается подробное документирование. В конце
этого этапа происходит демонтаж старой системы и переход на новую.

**Эксплуатация и обслуживание**. После того как система заработала, она изучается на предмет обнаружения и исправления недостатков разработки. В течение своей жизни система периодически пересматривается. Изменения в нее вносятся по мере возникновения проблем или появления новых потребностей. Если требуются существенные изменения системы, по сути означающие ее замену, то цикл разработки начинается сначала.

В дополнение к этим пяти этапам на протяжении всего цикла разработки ИС производится планирование, управление поведенческими реакциями пользователей на изменения и оценка текущих возможностей всего проекта.

***Участники***

Для успешной разработки ИС должны сотрудничать очень многие люди.

Во-первых, это **руководство** организации. Разработка не может быть успешной без ясной позиции высшего руководства по поводу ее необходимости. Обеспечение поддержки, установка целей разработки, анализ работы отдела автоматизации, определение политики по выбору системы, участие в принятии важнейших решений, выделение средств, назначения на ключевые посты - основные функции менеджеров при создании ИС.

Роль **пользователей** (бухгалтеров, управленцев, других служащих, которые будут использовать систему) определяется тремя моментами. Они должны обозначить свои информационные потребности, они должны участвовать в самой разработке ИС и играть активную роль в оценке результатов ее деятельности.

Поскольку разработка ИС не признает границ между подразделениями и функциями, организации учреждают **руководящий комитет**. Его задачи: планирование и контроль за разработками, установление стандартов и политики, координация действий. В его состав включаются представители всех вовлеченных в процесс разработки сторон.

**Группа разработчиков** каждого проекта объединяет специалистов по ИС. менеджеров, бухгалтеров, аудиторов, других пользователей. В задачу группы входит планирование каждого отдельного проекта, обеспечение его своевременного и эффективного выполнения, взаимодействие с руководящим комитетом, аккумуляция и использование экспертных знаний, необходимых для разработки. Чтобы выполнить эти задачи, группа должна регулярно общаться с пользователями ИС и устраивать обсуждения новых идей и продвижений в выпол-
нении проекта.

**Системные аналитики** исследуют существующую систему, разрабатывают новую и готовят спецификации для программистов.

**Программисты** разрабатывают новые и модифицируют существующие компьютерные программы.

***Планирование разработки системы***

Планирование ИС представляет собой принятие решений по следующим вопросам:

• что представляют собой основные задачи, которые должна выполнять ИС для до-
стижения целей системы;

• за что возлагается ответственность на подразделение автоматизации и пользовате лей, и кто будет выполнять разработку ИС;

• когда может быть завершена разработка ИС;

• какая технология будет использована для разработки системы и какие технологии
должны использоваться для выполнения разработки качественно и в срок.

Одновременно должны быть определены желаемые результаты в терминах изменений выполняемых операций.

Если ИС плохо спланирована, то ее использование и изменение может оказаться очень неудобным и дорогим процессом. Поэтому до того как приступить к разработке, проводится предварительное планирование, которое призвано добиться:

*последовательности*. Цели и задачи системы должны соответствовать общему стратегическому плану организации;

*эффективности*. Система должна быть построена и работать максимально эффективно, а ее подсистемы должны быть скоординированы;

*использования передовых технологий*. Должны быть приняты решения, на основе каких технологиях будет строиться система;

*сокращения затрат*. Необходимо избегать дублирования, никому не нужных работ, перерасхода бюджета и времени. Построенная система должна быть проста в эксплуатации;

*адаптивности*. Руководство должно быть готово к новым потребностям, которые могут возникнуть в будущем, а работники - к будущим изменениям системы.

Когда усилия по разработке плохо спланированы, компании приходится чаще возвращаться к уже пройденным этапам, что приводит к задержкам и повышению затрат. На рис. 4.2 показаны шаги процесса разработки и причины, приводящие к нежелательным возвратам на более ранние стадии.

Каскадный подход хорошо зарекомендовал себя при построении ИС, для которых в самом начале разработки можно достаточно точно и полно сформулировать все требования, с тем чтобы предоставить разработчикам свободу реализовать их как можно лучше с технической точки зрения.



Рис, 3.2, Причины для возврата на предыдущие шаги цикла разработки ИС

В эту категорию попадают сложные расчетные системы, системы реального времени и другие подобные задачи. Однако в процессе использования этого подхода обнаружился ряд его недостатков, вызванных прежде всего тем. что реальный процесс создания ПО никогда полностью не укладывался в такую жесткую схему. В процессе создания ПО постоянно возникала потребность в возврате к предыдущим этапам и уточнении или пересмотре ранее принятых решений.

Основным недостатком каскадного подхода является существенное запаздывание с получением результатов. Согласование результатов с пользователями производится только в точках, планируемых после завершения каждого этапа работ, требования к ИС "заморожены" в виде технического задания на все время ее создания. Таким образом, пользователи могут внести свои замечания только после того, как работа над системой будет полностью завершена. В случае неточного изложения требований или их изменения в течение длительного периода создания ПО пользователи получают систему, не удовлетворяющую их потребностям. Модели (как функциональные, так и информационные) автоматизируемого объекта могут устареть одновременно с их утверждением.

Для преодоления перечисленных проблем была предложена спиральная модель ЖЦ (рис. 3.3). делающая упор на начальные этапы ЖЦ: анализ и проектирование. На этих этапах реализуемость технических решений проверяется путем создания прототипов. Каждый виток спирали соответствует созданию фрагмента или версии ИС. на нем уточняются цели и характеристики проекта, определяется его качество и планируются работы следующего витка спирали. Таким образом углубляются и последовательно конкретизируются детали проекта и в результате выбирается обоснованный вариант, который доводится до реализации.





Разработка итерациями отражает объективно существующий спиральный цикл создания системы. Неполное завершение работ на каждом этапе позволяет переходить на следующий этап, не дожидаясь полного завершения работы на текущем. При итеративном способе разработки недостающую работу можно будет выполнить на следующей итерации. Главная же задача - как можно быстрее показать пользователям системы работоспособный продукт, тем
самым активизируя процесс уточнения и дополнения требований.

Основная проблема спирального цикла - определение момента перехода на следующий этап. Для ее решения необходимо ввести временные ограничения на каждый из этапов жизненного цикла. Переход осуществляется в соответствии с планом, даже если не вся запланированная работа закончена. План составляется на основе статистических данных, полученных в предыдущих проектах, и личного опыта разработчиков. На этапе предварительного планирования составляются планы двух типов.

**План разработки проекта** - основная составляющая планирования ИС. Он содержит анализ затрат и доходов, требования к разработке и применению, включая требования к аппаратному и программному обеспечению, финансированию, кадрам, а также расписание мероприятий, необходимых для разработки. Деятельность по разработке проекта должна быть представлена графиком работ. График выполнения проекта представляет собой сжатое описание основных шагов разработки, определяющих сроки их выполнения и исполнителей. Множество инструментальных средств управления проектом могут применяться для разработки графика и отслеживания отклонений от него. Большинство проектов ИС используют диаграмму Ганта (Gantt chart), изображенный на рис.3.4.



Рис. 3.4, Представление графика работ диаграммой Ганта

Диаграмма Ганта представляет график визуально, показывая задачи вместе с запланированным и действительным сроками начала и завершения работ, а также требуемыми ресурсами. Задачи могут выполняться параллельно и быть последовательнымими. Требования ресурсов могут быть установлены в терминах человеко-месяцев, денежных средств, времени начала для некоторых задач. Диаграмма Ганта представляет собой отличный инструмент для взаимодействия
между группами разработчиков, выявления проблем и принятия решений о корректировке планов. Беглый взгляд на диаграмму Ганта показывает, выполняется ли проект по графику или с отставанием. На совещаниях разработчиков диаграмма Ганта является эффективным средством демонстрации успеха прогресса, выявления проблем и понимания необходимости перераспределения ресурсов. Аналогично технологии управления, диаграмма Ганта зависит от качества данных, описывающих график. Если начало работ не определено, то трудно сказать когда задача будет завершена.

Управление проектом ИС часто обнаруживает противоречие между масштабом системы и качеством разработки, с одной стороны, и выполнением графика работ - с другой стороны. Участники часто бывают сверхоптимистичны относительно длительности работ над проектом, в основном из-за этого проекты не учитывают сложность системы. Возможными вариантами решения на подобного рода проблемы являются изменение масштаба системы, изменение графика или выделение дополнительных ресурсов.

**Главный план** - это долгосрочный документ, определяющий, из каких блоков будет состоять система, как она будет разрабатываться, кто этим будет заниматься, как будут приобретаться необходимые ресурсы. Главный план должен также определять приоритеты частей всего проекта и описывать критерии, использованные при назначении приоритетов. Главный план готовится высшим руководством, а не компьютерными специалистами. Горизонт планирования обычно составляет около 5 лет, но этот план обновляется периодически,
по крайней мере раз в год.

**3.2. Анализ системы**

При появлении необходимости в новой ИС или улучшении старой подготавливается письменный запрос на разработку системы, который описывает текущие проблемы, причины необходимости изменений, цели предлагаемой системы, предполагаемые затраты и выгоды. Этот анализ проводится группой разработчиков. При этом анализ проходит пять стадий, цели которых показаны ниже (табл.3.1).

Таблица 3.1,

Шаги анализа системы



**3.2.1.Начальное исследование**

**Начальное исследование** проводится для анализа проблемной ситуации.Основная
цель этапа - выяснить точную природу проблемы, приведшей к необходимости разработки новой ИС. и получить поддержку руководства для продолжения разработки. Проблемы представляют собой нежелаемые ситуации, которые предотвращает организация для полного достижения своих целей и задач. Проблемы могут быть текущими, предполагаемыми или ожидаемыми. Хорошая формулировка проблемы включает следующие элементы:

— **список наблюдаемых симптомов** (выявленных несоответствий характеристик системы современным требованиям), установленных в измеряемой форме. Чем более определенные симптомы, тем большая вероятность того, что проблема может быть решена;

— **список предполагаемых мероприятий**, установленных как задачи или приложения, способствующие решению проблем или устранению симптомов;

— **предварительную оценку** требуемых ресурсов обычно установленных в терминах финансовых затрат.

Таблица 3.2,

Примеры формулировки проблем



Первый шаг в проверке правильности формулировки проблемы состоит в сравнении симптомов с выявленными задачами. Каждому симптому должна соответствовать одна или несколько задач.

На основе комбинации выявленных элементов должна быть определена цель всего проекта. Новая ИС бывает полезна, когда проблема является результатом недостатка информации, недоступности данных или неподобающей их обработки. Если проект одобряется, то готовится предложение о проведении анализа системы, ему назначается приоритет, он попадает в главный план, и группа разработчиков начинает обследование системы. Целью анализа является преобразование общих, неясных знаний о требованиях к будущей системе в точные (по возможности) определения. На этом этапе определяются:

• архитектура системы, ее функции, внешние условия, распределение функций между аппаратурой и ПО;

• интерфейсы и распределение функций между человеком и системой;

• требования к программным и информационным компонентам ПО, необходимые аппаратные ресурсы, требования к БД, физические характеристики компонентов ПО, их интерфейсы.

**3.2.2. Обследование системы**

На этапе обследования системы выполняется интенсивное изучение текущей ИС. Обследование может выполняться неделю или месяцы, в зависимости от сложности и задач системы. На этом этапе следует:

— получить полное понимание операций, выполняемых компанией, ее политики и процедур, потоков данных и информации, сильных и слабых сторон существующей ИС. характеристику имеющегося персонала, используемое программное и аппаратное обеспечение;

— сделать предварительную опенку текущих и будущих информационных потребностей компании и определить природу необходимых изменений.

— наладить рабочие взаимоотношения с пользователями ИС;

— собрать данные, показывающие потребности пользователей, провести технико-
экономическое обоснование и дать рекомендации руководству.

Внутренние данные о системе собираются как на основе анализа циркулирующей документации, так и в процессе взаимодействия со служащими компании. Внешними источниками информации о системе служат консультанты, клиенты, поставщики, профессиональные организации и правительственные учреждения. Достоинства и недостатки основных методов сбора информации представлены в таблице 4.3.

**Таблица 3.3**

Достоинства и недостатки методов сбора данных о системе



Собранная информация должна быть документирована для дальнейшего использования. Эффективный способ документировать систему в процессе анализа. - создать модель системы. Для этой цели используются различные методики: диаграммы потоков данных (DFD - dataflow diagrams) для описания документооборота в компьютерных системах и в организации, блок-схемы процедур, моделирование данных (ERD- entity relation diagrams), прототи-
пирование и др.

По завершении сбора данных информация подвергается анализу с целью выделить
сильные и слабые стороны системы. В заключение готовится отчет об обследовании системы.

**3.2.3. Техникоэкономическое обоснование**

Информация, собранная на этапе обследования системы, используется для проведения более детального технико-экономического обоснования (feasibility study, business case). Это самый трудный этап в создании системы. При работе с масштабными системами для него даже часто создают специальную группу, в которую входят руководители, бухгалтеры, пользователи, специалисты по компьютерам. Поскольку технико-экономическое обоснование выполняется в начале цикла разработки системы, часто исследование не учитывает некоторые
проблемы или параметры, которые могут значительно изменить ожидаемый масштаб системы. Естественно, что обоснование требует времени и средств и для очевидных проектов не выполняется. Стоимость работ по технико-экономическому обоснованию должна составлять не более 5-10% от стоимости разработки системы.

Руководящий комитет использует отчет этого этапа для принятия решения о том. следует ли дальше продолжать проект. Поэтому здесь подлежат рассмотрению пять важных аспектов.

*Технический аспект*. Может ли ИС быть создана и внедрена с использованием существующих технологий? Кроме того, рассматривается вопрос о подготовленности и наличии опыта у персонала выполнить разработку предлагаемого решения. Как правило, аналитик доказывает техническую обоснованность, сравнивая с решением аналогичных проблем. Эффективным является также использование прототипов, физического и аналитического моделирования.

*Организационный аспект*. Совпадает ли система со стратегическими целями организации? Если ответ отрицательный, то возможно лучше будет потратить средства на другие проекты.

*Операционный аспект*. Может ли система быть применимой в среде пользователей? В какой степени предлагаемое решение выполняет требования пользователей и что пользователи думают о таком решении? Возможно, существуют этические соображения применения компьютерных технологий, например, руководитель страдает компьютерофобией. Такие нематериальные факторы могут вызывать неудачу применения системы так же как технологи-
ческие и экономические. Ряд аналитиков называют этот фактор политическим.

*Временные возможности*. Может ли ИС быть создана в отведенное время?

*Экономические возможности*. Покрывают ли выгоды от ИС расходы времени, средств и других необходимых для нее ресурсов? Основой для анализа служит модель планирования инвестиций в основные средства, в рамках которой оцениваются денежные величины всех доходов и затрат в различные периоды времени и определяются период окупаемости, чистый дисконтированный доход и внутренняя норма рентабельности. Часто определить денежное выражение бывает трудно, и в этом случае оно подлежит оценке. Во внимание принимаются как осязаемые, так и неосязаемые выгоды: экономия затрат, улучшение обслужи-
вания клиентов, производительность, улучшение принятия решений и обработки данных, лучший контроль, большее удовлетворение служащих своей работой и т.д. Затраты обычно подсчитать проще. Это расходы на оборудование, программное обеспечение, разработку системы, установку и обслуживание. Расходы, связанные с кадрами, - на обучение, оплату обслуживающего персонала.

*Экономические возможности*. Покрывают ли выгоды от ИС расходы времени, средств и других необходимых для нее ресурсов? Основой для анализа служит **модель планирования инвестиций** **в основные средства**, в рамках которой оцениваются денежные величины всех доходов и затрат в различные периоды времени и определяются период окупаемости, чистый дисконтированный доход и внутренняя норма рентабельности. Часто определить денежное выражение бывает трудно, и в этом случае оно подлежит оценке. Во внимание принимаются как осязаемые, так и неосязаемые выгоды: экономия затрат, улучшение обслуживания клиентов, производительность, улучшение принятия решений и обработки данных, лучший контроль, большее удовлетворение служащих своей работой и т.д. Затраты обычно подсчитать проще. Это расходы на оборудование, программное обеспечение, разработку системы, установку и обслуживание. Расходы, связанные с кадрами, - на обучение, оплату обслуживающего персонала.

Предполагая, что одно либо несколько осуществимых решений существуют, аналитик готовит отчет технико-экономического обоснования, который описывает несколько альтернатив и рекомендует направление реализации. Структура типичного отчета представляется следующим образом.

Титульная страница – наименование проекта, заголовок проекта, автор(ы), дата.

Оглавление – список глав с номерами страниц.

Определение проблемы – ясное, сжатое сжатое до одной страницы определение проблемы.

Резюме руководству - ясное, сжатое, до одной страницы резюме, полученные результаты и рекомендации. В резюме включаются основные источники информации, рассматриваются принятые и отклоненные альтернативы. Указывются стоимость, преимущества, ограничения и запланированное время реализации рекомендованной альтернативы.

Метод изучения: описание подхода и процедур, используемых в процессе изучения. Упоминаются источники и ссылки; указываются ключевые разработчики и кратко описывается существующая система. Многие детали помещаются в приложении.

Анализ: представляется анализ высокого уровня логической системы. Включает развернутую формулировку целей системы, ограничений и масштаба системы; включает логическую модель (*DFD,ERD*) и возможно словарь данных предлагаемой системы; указываются основные взаимосвязи с другими системами.

Рассматриваемые альтернативы – включается записка о техническом, экономическом, операционном и организационном аспектах, ориентировочный план применения, диаграмма потоков данных высокого уровня для каждой рассматриваемой альтернативы.

Рекомендации – устанавливается рекомендованное направление разработки проекта. Представляется материал для поддержки и обоснования рекомендаций. В особенности представляется анализ стоимость/эффективность.

План разработки – включает проектный план и проектную стоимость для каждого этапа жизненного цикла в предположении следования рекомендациям. Для этапа анализа указывается точная оценка времени и стоимости.

Приложение – рисунки, графики, обзор, интервью, диаграммы, записки, ссылки, контакты, и т.п.

***3.2.4. Определение информационных потребностей и требований к системе***

После признания проекта осуществимым, компания подробно определяет информационные потребности и документирует требования к системе. Даже для относительно простых ИС определение информационных потребностей может быть очень сложным процессом из-за огромного количества и разнообразия информации, которая требует определения. Кроме того, для служащих очень часто представляет трудность определить свои информационные потребности или они определяют их неточно. На рис. 3.5 в юмористической форме отражены проблемы, связанные с этим этапом.

Очень часто после завершения разработки системы пользователи заявляют, что сделано то, что они просили, но не то, что им действительно необходимо. При всей своей сложности данный этап является еще и очень ответственным, т.к. исправление некачественных решений, допущенных на этом этапе, предполагает впоследствии большие затраты.

Следующий шаг - **определение целей системы**. Как правило, в качестве целей конкретизируются такие показатели, как полезность производимой информации, надежность, доступность, своевременность получаемой информации, наличие достаточной мощности, экономическая эффективность, гибкость, безопасность, управляемость, контролируемость ИС, улучшение обслуживания клиентов.



|  |  |  |  |
| --- | --- | --- | --- |
| Как предлагают пользователи и менеджеры | Как преподносится высшему руководству | Как планируется группой разработчиков | Как утверждается руководящим комитетом |



|  |  |  |  |
| --- | --- | --- | --- |
| Как разработана главным аналитиком | Как написана программистами | Как установлена для использования | Что на самом деле нужно пользователям |

*Рис. 3.5 Коммуникационные проблемы анализа системы.*

И, наконец, **определение ограничений**. Успех разработки ИС часто зависит от понимания ограничений, в которых работает компания: требования законодательства, политика руководства, недостаток квалифицированного персонала, возможности и настроения пользователей, доступные технологии, ограниченные финансовые ресурсы.

Для **определения требований к системе** используются различные методики: опрос пользователей об их потребностях, анализ существующих систем, исследование их применения, использование прототипа новой системы (табл.3.4.).

Результатом этапа является детальное описание требований к новой ИС и определение полного набора функций системы. Определить, как она должна это делать - задача последующих этапов разработки системы.

*Таблица 3.4.*

*Возможное содержание требований к системе*

|  |  |
| --- | --- |
| Процессы | Описание всех процессов в новой системе, включая распределение обязанностей исполнителей |
| Элементы данных | Описание элементов данных, включая имена, форматы, источники и предназначение |
| Структуры данных | Предварительные структуры данных, показывающие, как элементы данных будут организованы в логических записях |
| Выходы | Образцы выходных документов с объяснением их назначения, частоты использования, предназначения |
| Входы | Образцы первичных документов и объяснение их содержания, источников и ответственных за них лиц |
| Документация | Описание, как новая система и ее подсистемы должны работать |
| Ограничения | Описание таких ограничений, как сроки выполнения работ, меры безопасности, ограничения в кадрах и др |
| Контроль | Описание мер контроля, позволяющих гарантировать точность и надежность входов, выходов и обработки данных |
| Реорганизация | Требуемые реорганизационные меры - увеличение или сокращение штатов, добавление новых функций, создание подразделений |

Подготовленные требования подлежат обсуждению с заинтересованными сторонами - пользователями и менеджерами и подписываются руководством, если достигнуто согласие. Для обеспечения такого обсуждения часто подготавливается описательный документ “нетехнического” характера.

***3.2.5. Подготовка отчета***

Анализ системы завершается подготовкой отчета, который суммирует и документирует все проделанные шаги и служит источником информации для разработки системы.

Решение о продолжении работ принимается по крайней мере три раза: После начального исследования, после исследования возможностей и после завершения всего анализа системы.

Разработчики новой системы часто испытывают давление со стороны руководства компании, которое вынуждает их пропускать этап анализа системы и сразу приступать к написанию программ. В больших и сложных организациях это практически всегда приводит к появлению дорогой системы, которая не нужна ее пользователям и поэтому отторгается.

**3.3. Концептуальная разработка**

Фаза концептуальной разработки ИС предназначается для реализации требований пользователей и решения других проблем, обнаруженных на этапе анализа системы. В какой-то момент во время концептуальной разработки должно быть принято решение о выборе между покупкой и разработкой программного обеспечения.

***3.3.1. Оценка альтернатив***

Существует много способов реализации различных компонентов ИС, поэтому ее разработка сопровождается рядом решений по выбору. Примеры приведены в таблице 3.5.

*Таблица 3.5.*

*Альтернативы для некоторых компонентов ИС*

|  |  |
| --- | --- |
| **Компонент** | **Альтернативы разработки** |
| Коммуникационный канал | Телефонная линия, коаксиальный кабель, оптоволокно, спутниковый канал |
| Обработка данных | Централизованная, децентрализованная, распределенная |
| Хранение данных | Лента, гибкие или жесткие диски, твердая копия |
| Структура данных | Файловая или база данных |
| Доступ к файлам | Прямой или последовательный |
| Ввод данных | С клавиатуры, голосом, EDI, OCR, POS |
| Место обработки | В компании или в сторонней организации |
| Обработка на | Большой ЭВМ, микро или персональной ЭВМ |
| Вывод информации | На монитор, на бумагу, на оборотный документ, голосом |
| Обработка операций | Ручная, пакетная, онлайновая в реальном времени |
| Приобретение программ | Готовые, модифицированные, разработанные |

Группа разработчиков должна определить возможные альтернативы и оценить каждую из них по отношению к тому, как хорошо они удовлетворяют целям организации и системы (1), как они удовлетворяют потребности пользователей(2), являются ли они экономически осуществимыми (3), какими достоинствами и недостатками они обладают (4). Окончательное решение по выбору должен принимать руководящий комитет.

***3.3.2. Подготовка спецификаций и отчета***

После выбора альтернатив разработки группа разработчиков определяет **концептуальные спецификации** для таких элементов как:

*выходы*. Поскольку система разрабатывается для удовлетворения нужд пользователей, работа начинается с определения спецификаций выходов системы, которые для каждого производимого отчета включают: частоту его использования (ежедневно или еженедельно), его содержание, внешнего вида, на каком носителе предоставляться (экранная или твердая копия);

*хранение данных*. Описывается структура данных: последовательный или прямой доступ, файловая организация или база данных, размеры и типы полей;

*входы*. Какие данные должны вводиться, где, когда и как данные должны собираться;

*процедуры обработки*. Как обрабатывать введенные и хранящиеся данные для получения необходимых отчетов. Также определяется частота и последовательность обработки. Предусматриваются процедуры контроля, аудита и обеспечения безопасности.

В заключение группа разработчиков готовит **отчет о концептуальной разработке**, который будет использован: как руководство на этапе физической разработки(1), для оценки, насколько удовлетворены требования пользователей(2), и чтобы помочь руководящему комитету оценить ход работ(3). Главное его содержание - полное описание спецификаций одного или нескольких предлагаемых вариантов. Окончательный выбор варианта ложится на плечи руководящего комитета.

**3.4. Физическая разработка**

На этапе физической разработки широкие, ориентированные на пользователя концептуальные требования к ИС переводятся в детальные спецификации, которые затем используются для программирования и отладки программ.

***3.4.1. Разработка выходов***

Цель этого этапа - определить назначение, формат, содержание, расписание, в соответствии с которыми будут создаваться отчеты, выходные документы и экранные формы. Вот некоторые наиболее важные вопросы разработки выходных документов: использование (кто, почему и для принятия каких решений нуждается в отчетах), носитель (бумага, бланк, экран, голос, файл и т.п.), формат (таблица, текст, график), адресат (место назначения), доступ (кто имеет право знакомиться), детализация (определяет объем объяснений), расписание (в какое время производить).

Как правило выходы принадлежат одной из следующих категорий:

1. **Регулярные отчеты** имеют заранее определенное содержание и формат и готовятся регулярно. Например, ежемесячный отчет о производительности, еженедельный отчет о продажах, ежегодный финансовый отчет.
2. **Специальный анализ** не имеет заранее определенной структуры и готовится по запросам руководителей. Например, это может быть ответ на вопрос, какой из трех новых продуктов приносит наибольшую прибыль?
3. **Отчеты о необычных обстоятельствах** имеют заранее определенное содержание, но готовятся только в ответ на необычные условия. Примеры: отсутствие на рабочем месте большого количества служащих, превышение затрат, недостаток материально- производственных запасов.
4. **Отчеты по требованию** имеют предопределенное содержание и формат, но готовятся только по требованию.

Всякий разработанный на этом этапе отчет должен быть подвергнут оценке пользователей, чтобы гарантировать его полноту, актуальность и полезность.

***3.4.2. Разработка файлов и базы данных***

Основные задачи этого этапа - моделирование данных и создание базы данных. При этом окончательно решаются вопросы о носителе данных (жесткий, гибкий, оптический диск или магнитная лента), об организации доступа (прямой, последовательный, индексно-последовательный), о режиме обработки (пакетный или в реальном времени), о процедурах эксплуатации; оцениваются ожидаемые размеры файлов, скорость их обработки, процент ежегодного обновления записей и другая динамика базы данных.

Очень важно на этом этапе позаботиться о совместимости форматов данных с уже созданными и работающими ИС компании, а может быть и других организаций. В противном случае в будущем могут возникнуть проблемы интеграции баз данных.

***3.4.3. Разработка входов***

Задача этапа - полностью определить входные данные и оптимальные методы их ввода. Здесь рассмотрению подлежат: способ ввода каждого документа (клавиатура, сканер торгового терминала, OCR - устройство распознавания символов, голос), источник (где данные появляются), формат (первичный или оборотный документ, экранная форма), тип данных, оценка объема, частоты и затрат на ввод данных, требования к подготовке обслуживающего персонала, необходимые меры контроля ошибок и их исправления.

Все большее распространение в ИС получает использование автоматизированного ввода, основанного на оборотных документах с последующим их сканированием. Поэтому на этапе разработки входов проектируются формы таких документов. Вот некоторые принципы их разработки: данные, которые могут быть заранее определены, должны встраиваться в документ сразу, а не заполняться (название компании, нумерация, текущий год); выделение линиями, шрифтом, цветом или штриховкой различных частей документа; логичная группировка данных по частям документа; ясные инструкции по заполнению; подбор размера, цветов, толщины и плотности бумаги для облегчения работы с документом и т.д.

Если используется ввод с клавиатуры, то большое значение приобретает разработка экранных форм, которая позволяет обеспечить ускорение ввода данных и избежать возникновения ошибок. Приведем некоторые правила:

* ввод данных осуществляетсяв том же порядке, в котором они появляются на исходном документе;
* минимизация ввода данных (например, автоматический вывод названия по номеру счета или табельного номера по фамилии работника);
* заполнение слева направо и сверху вниз;
* ограничение количества данных в одном кадре;
* автоматическая проверка правильности ввода (например, сверка почтового индекса и названия города) и т.д.

***3.4.4. Разработка программ и процедур***

**Разработка программ** - наиболее длительный этап во всем цикле разработки ИС. Этой работой всегда занимаются специалисты по программированию, и здесь существует своя специфика - планирование разработки, выбор инструментальных средств и языка программирования, распределение обязанностей в группе разработчиков, использование различных методик разработки программ, неизбежный этап отладки отдельных программ и программной системы в целом. Процесс разработки программ должен быть тщательно спланирован, разбит на параллельные процессы (обычно это разработка подсистем) и этапы подготовки программной системы в целом (исследовательский и испытательный, прототипы, рабочая версия), когда эксперты группы разработчиков могут оценить ход и направление проводимых работ.

Основная цель проектирования программного обеспечения - в определении автономных, независимых модулей. Степень независимости модуля обратно пропорциональна количеству элементов данных (или соединений), образующих поток между модулем и остальной частью системы. Следовательно, акцент этого шага должен быть сориентирован на увеличение независимости модулей, выявляя и реструктуризируя модули с чрезмерными потоками данных.

Вторая цель проектирования состоит в том, чтобы определить взаимосвязанные модули, которые исполняют одну законченную функцию. Акцент этого шага состоит в объединении модулей, которые исполняют общую функцию, объединяя функции для уменьшения количества интерфейсов, перемещая модули для увеличения эффективности системы.

С использованием результатов предыдущих двух шагов готовится заключительная версия диаграммы структуры программного обеспечения. В завершение этапа проектирования программного обеспечения выполняется анализ преобразования, целью которого является группирование вместе модулей (или процессов), которые обрабатывают определенный набор данных или определенную структуру данных. Например процессы, которые принимают данные по учету движения материально-производственных запасов, изменяют уровень запасов и изменяют данные по общей структуре запасов, вероятно, связаны. Прежде всего, для модулей определяются вход, выход и преобразование (модификация данных) и координация (управление). Группирование модулей для формирования структуры управления может вовлекать обозначение одного модуля как управляющего или создание нового управляющего модуля. Зависимые модули называются подчиненными.

Далее выполняняется операционный анализ, цель которого состоит в том, чтобы сгруппировать все модули (или процессы) вызванные одной операцией для формирования операционного центра. Например, все задачи, выполняемые при получении заказа от поставщика, связаны между собой. Часто центр управления представляет собой модуль управления. На основании выполненного проекта программного обеспечения разрабатываются физические модули, сгруппированные для эффективного выполнения на компьютере. Часто выполняется процедурный анализ для определения процедур, которые должны быть сгруппированы в пределах одного модуля, чтобы избежать серьезных ошибок выполнения и/или испытаний.

Управлению процессом разработки программ следует уделять самое серьезное внимание. В этой связи для каждой задачи должна быть вычислена оценка длительности разработки следующим образом:

оценивается минимальное количество необходимого времени на выполнение задачи – **OT**. Эта величина подразумевает отсутствие задержек и прерываний в разработке задач, например, таких как болезни;

оценивается максимальное количество необходимого времени на выполнение задачи - **PT**, в предположении, что случается все, что может случиться плохого;

устанавливается наиболее вероятное время – **MLT**- необходимое для выполнения задачи.

Тогда ожидаемое продолжительность задачи – **ED** - можно определить по следующей формуле:

**ED=(OT+(4xMLT)+PT)/6.**

Эта формула обеспечивает взвешенное среднее различных оценок.

**Процедуры** - это функциональные режимы по эксплуатации системы. Они должны обеспечивать вопросы подготовки ввода данных, обработки операций, обнаружения и исправления ошибок, выверку балансов, доступ к базе данных, подготовку и предоставление отчетов, инструкции оператора ЭВМ. Процедуры могут принимать форму руководств по системе, инструкций пользователя, обучающих материалов, кадров подсистемы помощи. Они могут готовиться как группой разработчиков, так и пользователями системы.

***3.4.5. Разработка методов контроля и подготовка отчета***

Без обеспечения качественных, точных и корректных данных на входе системы, а также надежной и правильной их обработки получаемый на выходе ИС результат не будет полезным, а может даже оказаться вредным. Методы контроля предназначены для обеспечения целесообразности, эффективности и точности обработки данных, предотвращения ошибки, обнаружения и исправления их в момент появления. В выработке методов контроля большую роль играют специалисты предприятия - менеджеры и особенно бухгалтеры, т.к. именно они до тонкостей представляют себе деятельность фирмы и в состоянии определить мероприятия, обеспечивающие эффективный контроль за данными. Вот наиболее важные аспекты контроля.

*Законность*. Например, как ИС может убедиться, что денежные выплаты вносятся на правильные счета?

*Авторизация*. Действительно ли деятельность по вводу, обработке, сохранению и выводу информации производится не посторонними, а уполномоченными людьми?

*Точность*. Проверяются ли вводимые данные на точность? Какие меры принимаются, чтобы обеспечить обработку данных без потерь?

*Доступ*. Как предотвращается незаконный доступ к данным?

*Нумерация*. Перенумерованы ли документы, чтобы предотвратить ошибочное или намеренное их использование в посторонних целях и чтобы своевременно обнаруживать пропажу или кражу документов?

*Возможность проверки*. Могут ли данные операций быть прослежены от первичных документов до окончательного вывода (и наоборот)?

По окончании физической разработки готовится отчет, который служит базовым документом для принятия руководством ответственного решения о начале фазы внедрения.

**3.5. Внедрение**

Внедрение системы - это процесс установки аппаратного и программного обеспечения и начало реальной работы ИС.

***3.5.1. Планирование внедрения***

В **плане внедрения** указываются мероприятия по внедрению, для каждого из которых определены ожидаемые даты завершения, оценки затрат и ответственные исполнители. Выполнение плана внедрения контролируется руководящим комитетом.

Группа, занимающаяся внедрением, должна определить **факторы риска**, понижающие вероятность успешного внедрения, и план должен содержать меры или стратегию по блокированию каждого такого фактора.

Например, один из таких факторов состоит в том, что изменения в ИС могут потребовать и изменений в существующей организационной структуре компании. Может понадобиться создание новых подразделений, а существующие могут ликвидироваться за ненадобностью или сокращаться в размерах. Как правило, изменяются структура и статус отдела, занимающегося обработкой данных.

***3.5.2. Безопасность***

Цель безопасности состоит в том, чтобы защитить аппаратные средства ЭВМ, программное обеспечение, данные и другие ресурсы системы от неавторизованного, некорректного или нежелательного доступа, использования, модификации или воровства. В традиционной информационной системе, построенной на основе централизованной универсальнй ЭВМ компьютер и большинство периферии расположены в помещениях ограниченного доступа. Взаимодействие большого числа пользователей и физически небезопасной периферии, кабелей, линий связи и пунктов доступа делает современные системы соблазнительной целью. Интернет усложняет эту проблему.

*Угрозы безопасности*

Предмет безопасности рассматривается, если стоимость нарушения безопасности превышает ценность этого предмета. Совершенная информационная безопасность системы может представлять собой недостижимую цель.

Нарушения могут быть получены от людей, аппаратных средств ЭВМ и/или программного обеспечения.

**Люди**. С недавних пор хакеры и взломщики стали широко известны. Первоначально хакеры были опытные программисты, виртуозно создающие изящное программное обеспечения. Сегодня, однако, термин как правило применяется к тем, кто незаконно проникает в компьютерные системы. В популярном использовании слово "хакер" (более общий термин) применяется и в хорошем и в злонамеренном смысле.

Помимо хакеров и взломщиков, такие члены организации как служащие, бывшие служащие, консультанты, клиенты и заказчики совершают большинство нарушений безопасности. В отличие от хакеров сотрудники имеют относительно свободный доступ к системе. Промышленные шпионы, как известно, предлогают деньги в обмен на соответствующую информацию или программное обеспечение. Рассерженные служащие информационных систем (действующие и бывшие) особенно опасны.

Даже честные сотрудники могут представлять риск безопасности. Люди не всегда осторожны в отношении защиты своих паролей, кодов безопасности, номеров телефона, оборудования и рабочих мест.

**Аппаратные средства ЭВМ.** Персональный компьютер или рабочая станция - одно из самых слабых звеньев в безопасности. Пользователи обмениваются данными в Интернет, совместно используют общее программное обеспечение и разделяют общую периферию, что может составлять угрозу безопасности. Неавторизованный доступ к общим файлам на серверах и периферии (магнитная лента, принтеры, графопостроители и.т.д) вредит безопасности.

Физическая сеть также уязвима. Злоумышленники, как известно, подключаются к кабелю или телефонной линии или перехватывают спутниковые и микроволновые коммуникации. Телефонный доступ является особенно трудным для контроля, потому что входящий звонок может исходить отовсюду. Фактически хакеры и взломщики иногда запускают программы, которые последовательно набирают тысячи телефонных номеров, и отмечают те из них, которые возвращают модемный тон. Эти номера позже используются как возможные для доступа в систему.

Воровство ноутбуков является возрастающей проблемой. Кроме материальной ценности компьютера и программного обеспечения, жесткий диск может содержать корпоративные данные, пароли, коды доступа и другую важную информацию.

**Программное обеспечение.** Ошибки выполнения и неточно введенные данные, произведенные пользователями, представляют особую проблему для проектирования сетевой безопасности, потому что обнаружение момента появления данных является очень трудной задачей при параллельной работе нескольких пользователей и нескольких активных задачах. Кроме того, неавторизованный доступ создает трудности обеспечения целостности базы данных, особенно в случае модификации ее содержания.

Иные проблемы программного обеспечения могут оказаться более серьезными. "Часовая бомба" (*time bomb*) - программа, которая выполняется в определенный момент времени или при наступлении определенных условий. "Троянская конь" (*Trojan horse*) – кажущаяся безопасной программа, которая приглашает пользователю запустить ее. Некоторые подобные программы стирают оперативную память, разрушают жесткий диск или выборочные данные.

"Кролик" (*rabbit*) - программа, которая размножает себя до тех пор, пока память не будет заполнена и никакие другие программы не смогут выполняться.

Вирус - программа, которая способна воспроизводить себя и распространяется между компьютерами. Подобно биологическому двойнику, вирус - паразит, который присоединяет себя к другой программе, чтобы существовать и размножиться. Вирусы, как правило, проникают на другие компьютеры через инфицированные дискеты или полученные по сети копии инфицированных программ.

*Меры противодействия*

Имеется ряд средств и технологий противодействия угрозе безопасности.

**Физическая безопасность.** Физическая безопасность имеет отношение к запрещению физического доступа к системе, предотвращению физического разрушения системы. Доступ в запрещенную зону может контролироваться по удостоверениям личности, бейджам, ключам или персональным идентификационным номерам (PIN). Камеры наблюдения становятся все более и более обычным средством. Современные биометрические устройства могут использоваться для индивидуальной идентификации через сканирование сетчатки глаза, анализа отпечатков пальцев, голоса или анализа подписи.

Интернет - существенный источник нарушений безопасности. Многие организации используют firewalls для изоляции их внутренней сети от сети Интернет (или от других общественных сетей). *Firewall* - набор аппаратных средств ЭВМ, программного обеспечения и данных, который располагается между внутренней сетью и Интернетом, экранирует все поступающие и/или исходящие транзакции и позволяет проходить только авторизованным транзакциям. Часто firewall устанавливается на физически отдельном компьютере с общим хостом (компьютер, который связан с Интернетом) за пределами firewall и внутреннего сервера внутри firewall. Дополнительно критическое программное обеспечение может быть размещено в ядре операционной системы или разделено для усложнения неавторизованного доступа.

План мероприятий необходим для таких событий, как пожар, наводнение, землетрясение или отключение электрического питания. Может быть необходим контроль окружающей среды для регулирования температуры, влажности, пыли и.т.д. Резервные копии всего программного обеспечения и данных и запас компонентов аппаратных средств ЭВМ - важные элементы плана мероприятий.

**Логическая безопасность.** Логическая безопасность осуществляется системой в процессе ее функционирования. Например, в большинстве систем, использующих сеть, каждому пользователю присваивается уникальное имя и пароль. В некоторых ситуациях требуются дополнительные пароли для доступа к критическим данным или для запуска программы. Часто привилегии доступа назначаются по уровням, для большинства пользователей устанавливаются права доступа только читать (*Read Only*), меньшей группе назначаются полномочия изменять (*Modify*) некоторые данные, и только несколько человек - администраторов системы (*administrator*) - имеют статус доступа для изменения всех данных. Как правило, операционная система проверяет профиль (*profile)* пользователя или матрицу управления доступом (*access control matrix*) для проверки привилегий данного пользователя.

Только указание действительного имени пользователя и пароля еще не означает, что пользователь – именно тот за кого себя выдает. Установление подлинности, процесс проверки идентичности пользователя часто основывается на подтверждении дополнительной информации (например PIN или девичьей фамилии матери) или измерениях биометрических устройств.

Обратный вызов (*callback)* - другое полезное средство опознания пользователя. После регистрации пользователя с удаленной рабочей станции хост проверяет имя пользователя и пароль, разрывает связь, находит номер телефона рабочей станции пользователя и затем соединяется с ней.

Вирусы бывает трудно обнаружить или удалить, так что лучшей защитой от них является предотвращение. Персоналу не следует применять свободное программное обеспечение (на дискете, CD-ROM или через сеть), пока не станет известно, что источник является безопасным. Антивирусное программное обеспечение предназначено для иденитификации определенного образца кода (сигнатуры) и выдачи соответствующего сообщения при обнаружении вируса. Во многих системах антивирусное программное обеспечение выполняется непрерывно в фоновом режиме.

Другие методы предназначены для обеспечения восстановления информации или документации, когда безопасность нарушается. Протокол транзакций представлеят собой список всех недавних танзакций системы. Специальное программное обеспечение сравнивает содержание файлов или записей до и после транзакции и выводит обнаруженные отличия.

**Персональная безопасность.** Люди являются причиной большинства проблем безопасности. Следовательно, хотя это дорого и иногда спорно, такие средства управления персоналом как просмотр кандидатуры перед приемом на работу, периодические проверки и ротация назначения работы, необходимы. Основной принцип бухгалтерского учета предполагает, что никому единолично нельзя позволять размещать заказ и оплачивать счет. Аналогично системы часто разрабатываются, определяя ответственность таких связанных функций, как вход данных и их проверка различными подразделениями.

Стандартные процедуры, политика и руководства по безопасности являются важной частью любого плана мероприятий безопасности. Служащие должны знать, как выполняются процедуры безопасности.

Например, стандартные процедуры могут вынуждать пользователей периодически изменять их пароли. Программное обеспечение установки пароля может быть разработано таким образом, чтобы помочь пользователю выбрать лучший пароль, отклоняя простые пароли, устанавливая их минимальную длину, требуя комбинацию символов и цифр, и.т.д.

**Шифрование.** Для того чтобы сделать критическую информацию трудночитаемой, даже если сообщение перехвачено, данные могут быть зашифрованы (преобразованы в секретный код), переданы и затем расшифрованы на другом конце линии. DES *(Data Encryption Standart*) - cтандарт шифрования данных - рассматривается как очень трудный для взлома. Система ключей DES, содержашая личный и общий ключи, используется для безопасной передачи правительственных и большинства электронных передач фондов. Другой популярный ключевой алгоритм шифрования PGP (*Pretty Good Privacy*) был создан без правительственной поддержки и доступен в Интернете.

Исходя из названия система использует два ключа. Общий ключ получателя, который свободно распространяется, используется для шифрования сообщения. При получении сообщения, оно может быть расшифровано только при помощи личного секретного ключа.

**Проектирование безопасности**

Каждая организация должна установить стандарты безопасности и инструкции, которые применяются в информационной системе. Такие стандарты помогают не допустить нарушения безопасности в процессе разработки системы и обеспечивают проектировщика принципами безопасности.

Важный элемент любого набора стандартов безопасности - признание того, что не все системы или даже не все компоненты системы, требуют одного уровня безопасности Стандарты должны определять несколько уровней рисков безопасности и предлагать уровни предосторожности в соответствии с риском.

Цель выбора принципов защиты состоит в том, чтобы сбалансировать стоимость и риск. В процессе определения проблемы и стадии анализа подверженность нарушения безопасности и риски должны быть идентифицированы. Затраты, связанные с соответствующими контрмерами, должны быть частью анализа стоимость/эффективность. В конце стадии анализа необходимые меры безопасности должны быть зафиксированы в спецификации требований.

Безопасность не может быть добавлена на систему; она должна быть разработана в системе. План мероприятий безопасности на уровне системы должен быть создан на ранних стадиях процесса проектирования. Как только подход к безопасности системы выбран, соответствующие возможности безопасности должны быть разработаны на уровне соответствующих аппаратных средств ЭВМ, программного обеспечения, данных и процедур. Фактически любой компонент системы может представлять риск безопасности. Следовательно, безопасность - важное часть в проекте почти каждого компонента системы.

Во время функционирования и стадии сопровождения системы контроль системы играет важную роль в поддержке ее безопасности, в том числе и для обеспечения раннего предупреждение проблем безопасности.

***3.5.3. Тестирование программ и подготовка оборудования***

Процесс подготовки программ может длиться от нескольких дней до нескольких лет, в зависимости от сложности программной системы. Обычно в этом процессе можно выделить семь шагов.

*1. Определение потребностей пользователей*. Это консультации с пользователями, результат которых - соглашение о требованиях к программному обеспечению.

*2. Разработка плана*, определяющего этапы и сроки выполнения работ.

*3. Написание кода программ*. Часто именно этот этап по ошибке называют программированием. Но любой реально действующий программист скажет, что это только часть работы.

*4. Тестирование программ*. Процесс поиска и устранения ошибок в коде программы называется ее **отладкой**. Для этого используются различные методы. В некоторые из них вовлечены не только программисты. Один из таких методов, **настольные испытания**, состоит в логическом и визуальном инспектировании работы системы будущими участниками ее эксплуатации или просто сторонними лицами. Кроме того, при проверке используются **тестовые данные**, которые должны моделировать различные ситуации обработки, в том числе содержать различные ошибки, которые могут произойти в процессе реальной эксплуатации системы. В разработке таких тестов принимают участие специалисты организации. Большие программные системы, состоящие из подсистем (или модулей), тестируются, как правило, в три стадии: проверка отдельных модулей, проверка связей между модулями, проверка работы тестируемой программы совместно с другими программными приложениями. Некоторые разработчики программ считают, что на тестирование, отладку и переделку программ уходит от 20 до 30% затрат на всю разработку.

*5. Документирование программ*. Без качественной и подробной документации даже очень квалифицированным программистам потребуются огромные усилия, если понадобится модификация программ или исправление ошибок в них. Документирование, сделанное вовремя, в дальнейшем поможет сберечь средства.

*6. Обучение пользователей*. Эту стадию обычно начинают в конце тестирования программ. При этом часто используют документацию - руководства пользователя.

*7. Установка и использование программной системы*. Процесс исправления существующих программ называют **обслуживанием программ** или их сопровождением (*program maintenance*). Необходимость в этом появляется периодически при пересмотре отчетов, изменений в первичных документах, необходимости хранить в файлах дополнительную информацию, изменении налоговых ставок, обнаружении ошибок или при переводе системы на новые компьютеры.

Можно считать аксиомой, что всякая сложная программная система содержит ошибки. Многие из этих ошибок проявляются на этапе внедрения, т.к. именно здесь программа начинает работать в реальных условиях. Поэтому для их устранения может понадобиться определенное время и средства.

Любой компьютер, будь то ПК, миникомпьютер или большая ЭВМ, требует **подготовки оборудования** к эксплуатации. Большая система может потребовать подготовки помещения, обеспечения надежного питания, каналов связи, установки фальшпола, оборудования, регулирующего влажность и температуру, специального освещения. Важность такого объекта может потребовать мер безопасности - противопожарного оснащения, аварийного источника питания. Все это может занять определенное время, и данный этап начинается заранее, до момента установки.

***3.5.4. Выбор и обучение персонала***

Служащие могут наниматься за пределами компании или быть найденными среди персонала организации. Второй вариант обычно является менее дорогой и более эффективной альтернативой, поскольку служащие компании уже работали в ней и лучше понимают специфику и потребности организации. Кроме того, продвижения по службе, связанные с этим вариантом, обычно вызывают положительную реакцию в смысле лояльности и настроения работников.

В любом случае при внедрении новой ИС существует проблема обучения персонала работе с новым оборудованием. (Даже среди тех, кто уже имеет навыки обращения с компьютерами, до трети (по американским исследованиям) чувствуют необходимость более адекватного обучения работе с новым оборудованием.) Во-первых, качественное обучение требует много времени и денег, поэтому компании часто экономят на нем, используя небольшое число высококвалифицированного персонала. С другой стороны, те, кто хорошо разбирается в системе, настолько заняты текущими проблемами, что просто не имеют возможности выступать в качестве инструкторов. Однако, если работающий с системой персонал недостаточно обучен, эффект от применения ИС может быть значительно ниже ожидаемого, в том числе и экономический эффект. Скрытые затраты, связанные с постоянной помощью более квалифицированных сослуживцев, уменьшают общую производительность и увеличивают затраты компании.

Эффективное обучение должно быть нацелено не только на приобретение навыков работы с программным и аппаратным обеспечением, необходимых для использования новой ИС, служащие должны быть ознакомлены и с новой политикой, и с новыми операциями, связанными с процессом функционирования системы. Обучение должно быть спланировано так, чтобы к этапу тестирования и перехода на новую систему, служащие уже понимали ее работу и роль в организации.

***3.5.5. Документирование системы***

Как уже отмечалось, документирование системы играет очень важную роль в процессе ее дальнейшей эксплуатации и выполняется на всех этапах жизненного цикла разработки системы. Документация по новой системе бывает трех типов.

*Документация для разработчиков*. Включает описание системы, разработанные модели (DFD, ERD и др.), образцы выходов, входов, форматов файлов и баз данных, блок-схемы программ и их исходные тексты, результаты тестирования.

*Документация для операторов*. Расписание операций, доступные файлы и базы данных, правила поддержания безопасности и целостности, сохранности файлов и оборудования.

*Документация для пользователей*. Составляется для обучения пользователей работе с ИС и содержит руководство по проведению процедур и учебные материалы.

***3.5.6. Тестирование системы***

**Тестирование** - это комплексная проверка системы в условиях, приближенных к реальности. Проверке подвергается использование первичных документов и отчетов, рабочие и контрольные процедуры, процедуры обработки и программы. Система тестируется на предельных нагрузках и в условиях возможных мешающих факторов. Существуют три основные формы тестирования:

* **пошаговое тестирование** - последовательная проверка процедур и логики программ предпринимается на этапе разработки системы группой разработчиков и пользователей системы. Упор делается на проверку входов, файлов, выходов и потоков данных в организации. Пошаговая проверка, проводимая программистами, сосредотачивается на структуре программного кода;
* **обработка тестовых операций** нацелена на проверку правильности работы программы. Обрабатываются как правильные данные, так и данные, содержащие ошибки, чтобы убедиться в том, что ошибки будут обнаружены системой и правильно обработаны. Чтобы оценить результаты такого тестирования, заранее определяется желаемая реакция системы;
* **обработка реальных операций** использует не модельные, а копии реальных данных об операциях. Этот тип проверки обычно проводится пользователями системы, знающими процедуры компании и определяющими, правильно ли система себя повела при обработке.

Даже программное обеспечение, купленное в готовом виде, должно быть протестировано перед установкой. Невыполнение этих требований дорого обходится организациям, доверяющим свои деньги и репутацию системам, которые не подверглись тщательной проверке.

***3.5.7. Переход на новую систему***

Переход на новую систему - это процесс смены старой ИС на новую. Это неизбежный этап, имеющий свои особенности. Заменены или преобразованы должны быть многие элементы: программное и аппаратное обеспечение, файлы данных и процедуры. Этот процесс можно считать завершенным, когда новая ИС становится обычной работающей частью деятельности организации. При таком переходе может быть применен один из следующих подходов (рис.3.6)



*Рис.3.6. Методы перехода на новую систему*

**Прямой переход** означает, что старая ИС выводится из эксплуатации сразу при появлении новой. Он применяется, когда старая ИС не представляет больше ценности или новая настолько отличается от нее в лучшую сторону, что их сравнение лишено смысла. Этот подход не связан с дополнительными затратами, но является более рискованным, чем остальные, т.к. не предусматривает подстраховки на случай выявления недостатков новой системы.

**Параллельный переход** предусматривает совместное функционирование старой и новой систем в течение какого-то времени, пока новая ИС не докажет свою работоспособность. Такой метод связан с дублированием обработки всех данных компании и сопровождается большими затратами, однако является самым надежным. На практике компании подвергаются риску и испытывают проблемы в период перехода, поэтому данный метод пользуется большой популярностью.

**Последовательный переход** применяется, когда возможна замена элементов (например, подсистемы учета кадров, учета материально-производственных запасов, платежей и т.д.) ИС одного за другим. Такой подход связан со специальным проектированием новой системы, возможен не для всякой старой ИС и требует разработки временных подсистем, предназначенных специально для обеспечения совместной работы старых и новых блоков. При этом процесс перехода усложняется и требует большего времени.

**Переход по подразделениям** применяется в организациях, имеющих относительно самостоятельные и однородные подразделения (например, системы магазинов, банк со многими отделениями и т.д.), в которых предполагается устанавливать однотипные ИС, связывая их затем в единую систему. В каждом из подразделений переход может осуществляться одним из перечисленных выше методов. Преимуществами является локализация проблем перехода и возможность обучения персонала других подразделений на реально работающих системах. Недостатки - большое время перехода и необходимость создавать подсистемы для обеспечения совместной работы старых и новых ИС в разных подразделениях. С переходом на новую ИС связана еще одна проблема - **преобразование данных**. Довольно часто сложность и значение этой проблемы недооцениваются. Файлы данных могут потребовать модификации в трех аспектах:

* *смены носителя*. Файлы могут потребовать переноса, например, с магнитных лент на магнитные диски для обеспечения более быстрого доступа к ним.
* *смены содержания*. Например, для описания изменений в деятельности организации могут быть добавлены или удалены отдельные поля или таблицы.
* *смены формата*. Одно и то же содержание может храниться в разных форматах. Выбор формата файла связан с использующимися инструментальными средствами для разработки программ. Например, разные СУБД могут хранить свои файлы в разных форматах.

Проблема преобразования данных возникает потому, что новые ИС как правило решают расширенный круг задач и применяют более современные программные и аппаратные средства. Само преобразование данных состоит из нескольких этапов:

решения вопроса о том, какие файлы данных должны быть преобразованы;

проверки полноты и устранение неточностей;

собственно преобразования данных;

проверки новых файлов на предмет корректности преобразования.

Если преобразование было длительным, то новые файлы должны быть обновлены операциями, произошедшими в период преобразования.

Только после этого новая ИС сможет нормально выполнять свои обязанности, однако она должна некоторое время контролироваться, чтобы убедиться в успешности преобразования данных.

**3.6. Эксплуатация и обслуживание**

Последний шаг жизненного цикла разработки системы - ее эксплуатация и обслуживание. Чтобы убедиться, что установленная ИС удовлетворяет запланированным целям, проводится ее **инспекция после внедрения**, затем готовится отчет об инспектировании.

**Лекция 5. Базы данных**

В ИС одним из главных инструментов моделирования деятельности организации являются базы данных. В этом разделе обсуждается: Как ИС моделирует деятельность организации. Что нужно знать экспертам организации о разработке баз данных.

**5.1. Системы управления базами данных**

Данные организации, которая использует ИС, являются одним из ее главных активов. Они хранятся в **базах данных** (БД) и предоставляются пользователям, интересующимся различными вопросами. Например, менеджер по продажам может поинтересоваться списком клиентов в табличной форме, производственный менеджер - заказами, находящимися на исполнении, высшее руководство может захотеть проанализировать структуру продаж в графической форме по регионам или магазинам. Все это различные примеры **логического представления данных**. И хотя существует огромное количество разных способов просмотреть данные, все же они не хранятся в форме, сразу пригодной для каждого из таких представлений. Данные хранятся в едином **физическом представлении,** например в индексно-последовательных файлах. Одной из задач **системы управления базой данных** (СУБД или *database management* *system - DBMS*) является перевод логического представления каждого пользователя в физическое представление данных, чтобы они могли просмотреть то, что хотят (рис.5.1).

Часто физическое представление данных сильно отличается от того, как пользователи их получают. Например, баланс клиента, его имя и кредитная история могут храниться в разных файлах и даже на разных компьютерах. СУБД скрывает детали физического представления от пользователей, чтобы они могли сконцентрироваться на логических взаимоотношениях различных показателей.



*Рис. 5.1. Функция СУБД*

Часто физическое представление данных сильно отличается от того, как пользователи их получают. Например, баланс клиента, его имя и кредитная история могут храниться в разных файлах и даже на разных компьютерах. СУБД скрывает детали физического представления от пользователей, чтобы они могли сконцентрироваться на логических взаимоотношениях различных показателей.

Чтобы обеспечить предоставление нужных данных нужным пользователям и скрыть информацию от тех, кто ее не должен знать, в базах данных используется два механизма - **авторизация пользователей** (каждый, кто обращается к БД, должен назвать свое имя и свой пароль) и **справочник данных** (специальный служебный файл, который для каждого элемента данных содержит его описание, в том числе - кому разрешен его просмотр), который часто называют **словарем данных**, т.к. от его содержимого зависит, на каком языке “разговаривает” БД, как называет свои элементы.

***5.1.1. Схемы баз данных***

**Схема** описывает логическую структуру БД. Существует три уровня схем: концептуальная, внешняя и внутренняя (рис.5.2)

**Концептуальная схема** дает общий взгляд на организацию всей БД. Она перечисляет все элементы данных и отношения между ними. **Внешняя схема** состоит из набора логических представлений пользователей, каждое из которых называется подсхемой. **Внутренняя схема** представляет структуру базы данных. Она описывает, как данные на самом деле хранятся, включая информацию об указателях, индексах, длине записей и т.д. Между схемами этих уровней существует соответствие, называемое отображением. Это правила преобразования одного представления в другое. СУБД использует отображения схем для вывода и ввода данных.

При создании БД для ИС эксперты организации (менеджеры, бухгалтеры) привлекаются для разработки схем концептуального и внешнего уровня. Поэтому им важно понимать разницу между этими типами схем. Например, для описания розничных продаж база данных на концептуальном уровне может содержать информацию о клиентах, продажах, кассовых операциях, продавцах и кассирах, товарах, счетах.

На внешнем уровне из этой схемы может быть получено множество подсхем, каждая из которых предназначена для потребностей отдельных пользователей и должна скрывать от них данные, не относящиеся к выполнению их обязанностей. Так, внешняя подсхема для служащего, оформляющего заказы, может содержать данные о кредитных лимитах и текущих балансах клиентов, количестве товаров на складе и их ценах, но не будет включать информацию о закупочной стоимости товаров или текущих банковских счетах компании. Внешняя подсхема для служащих, обеспечивающих доставку товаров, будет содержать информацию об адресах клиентов, но не об их текущих балансах.



*Рис. 5.2. Три уровня схем базы данных*

***5.1.2. Использование языков СУБД***

Любая СУБД должна обеспечивать три базовых функции - создание, изменение и запросы к базе данных. Наборы команд, которые используются для выполнения этих функций, называются соответственно языком описания данных, языком управления данными и языком описания запросов. Эти языки призваны упростить работу с БД, поскольку при работе с данными позволяют оперировать именами полей, таблиц, индексов вместо описания физического расположения этих элементов.

**Язык описания данных** (ЯОД или *data definition language - DDL*) используется для построения справочника данных (1), создания или инициализации БД (2), описания логического представления для каждого пользователя (3) и определения ограничений, обеспечивающих безопасность записей и полей БД (4).

**Язык управления данными** (ЯУД или *data manipulation language - DML*) используется для обслуживания БД, т.е. выполнения операций обновления, вставки, удаления записей.

**Язык описания запросов** (ЯОЗ, или *data query language - DQL*) используется для выбора данных из БД, их упорядочения и предоставления в ответ на запрос пользователей.

***5.1.3. Функции СУБД и ее пользователей***

Обычно **пользователи** имеют доступ к языку описания запросов. Что касается ЯОД и ЯУД, то доступ к ним предоставляется только тем служащим, которые отвечают за администрирование и программирование СУБД. Это просто означает, что изменять содержимое БД могут не все пользователи, а только ответственные за это лица. Распределение обязанностей таких привилегированных пользователей - существенная часть жизни БД.

**Администратор данных** отвечает за разработку политики и процедур управления всеми данными организации, а не только данными, хранящимися в БД. Другими словами, администратор данных отвечает за информационные потребности организации и в частности решает, какие данные будут храниться в БД.

**Администратор базы данных** отвечает за координацию, контроль и управление БД. Он должен беспокоиться не только о пользователях и их информационных нуждах, но и о том, как БД работает и как в ней хранятся данные. Основные обязанности администратора БД - создание логической модели БД, установка стандартов данных, одобрение изменений в структуре БД, разработка методов выбора данных в соответствии с требованиями пользователей, определение и обслуживание физической структуры БД, ведение справочника данных, разработка и реализация методов контроля для обеспечения точности и сохранности БД. Для выполнения этих задач администратор БД использует ЯОД.

**Прикладные программисты** пишут программы, призванные взаимодействовать с БД. Они используют ЯУД для доступа и изменения содержимого БД.

**5.2. Реляционные базы данных**

СУБД характеризуются различными логическими моделями, на которых они основаны. **Модель данных** - это абстрактное представление о содержимом БД.

***5.2.1. Реляционная модель данных***

Большинство современных СУБД являются реляционными и используют **реляционную модель данных**, которая представляет данные в виде таблиц. Однако следует помнить, что в терминах реляционной модели описываются только концептуальная и внешняя схемы, а на внутреннем уровне данные хранятся не в таблицах. Каждая таблица представляет одну сущность (например, товар), каждая строка в таблице содержит данные об одном экземпляре этой сущности (например, о конкретном товаре). Каждый столбец таблицы соответствует одному атрибуту сущности (например цене, коду поставщика или количеству - таблица 5.1.).

Реляционная модель данных накладывает ограничения на структуру таблиц, которые призваны обеспечить целостность и точность трактовки содержащихся в них данных. Некоторые из них кажутся очевидными, но они полезны на этапе моделирования данных.

*Таблица 5.1.*

*Таблицы видов товаров и их поставщиков*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Код товара*** | **Описание** | *Код поставщика* | **Количество на складе** | **Цена** |
| ***1036*** | Холодильник | *10023* | 23 | 2310 |
| ***1038*** | Холодильник | *10034* | 0 | 3100 |
| ***1039*** | Стиральная машина | *10034* | 52 | 2500 |

*Таблица 5.2*

|  |  |  |
| --- | --- | --- |
| ***Код поставщика*** | **Описание** | **Адрес** |
| ***10011*** | “Горизонт” | Россия, … |
| ***10023*** | “Бирюса” | Россия, … |
| ***10034*** | BOSS | ФРГ, … |

*1. Первичный ключ должен быть единственным*. **Первичный ключ** - это атрибут, который однозначно определяет каждую строку таблицы (например, код товара может быть первичным ключом для таблицы товаров). Это ограничение еще называют **правилом целостности сущности**, т.к. если у двух экземпляров сущности один первичный ключ, то их невозможно отличить.

*2. Каждый внешний ключ должен быть либо пустым, либо соответствовать одному из значений первичного ключа в другой таблице*. **Внешний ключ** - это атрибут таблицы, который является первичным ключом другой таблицы. Внешние ключи используются для описания связей между сущностями (например, код поставщика в таблице видов товаров задает связь с таблицей поставщиков, что описывает принадлежность товаров тому или иному поставщику). Это ограничение называют **правилом ссылочной целостности** (например, если в таблице товаров указан код поставщика 1010, то это означает, что поставщик неизвестен или данные о нем утеряны, поэтому в таких случаях вместо просмотра таблицы поставщиков коду поставщика назначают пустое значение).

*3.* *Каждый столбец таблицы должен описывать свойство сущности, определенной первичным ключом* (например, в таблице товаров не должно встречаться имя покупателя, иначе получится, что этот товар предназначен только для одного покупателя).

*4.* *В каждой строке таблицы должно содержаться только одно значение* (например, таблица поставщиков содержит в каждой строке описание одного и только одного поставщика).

*5.* *В каждом столбце все значения должны быть одного типа данных* (например, цена товаров не должна указываться прописью).

*6. Порядок строк и столбцов не должен иметь значения*. Это означает, что переупорядочение строк или столбцов не должно менять смысла информации, представленной в таблице. Данное свойство означает, например, что не должно быть строк, меняющих смысл содержимого следующих за ней строк (то же самое и для столбцов). Например, в таблице поставщиков не могут содержаться строки с описанием “акционерные общества” или “частные предприятия”, которые предполагают разбиение таблицы на части. Вместо этого надо вводить дополнительный атрибут “тип предприятия”. Другой пример: холодильники в таблице товаров не должны быть упорядочены по объему камеры, вместо этого надо использовать отдельный атрибут.

***5.2.2. Нормализация***

Еще один вопрос, на который необходимо получить ответ: почему данные разбиваются на таблицы, каждая из которых представляет отдельную сущность? Чтобы на него ответить, рассмотрим альтернативный вариант - хранение данных в одной таблице 5.3.

*Таблица 5.3.*

*Хранение данных о товарах и поставщиках в общей таблице.*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***Код*** ***товара*** | **Описание** | **Количество на складе** | **Цена** | **Описание поставщика** | **Адрес** |
| ***1036*** | Холодильник | 23 | 2310 | “Бирюса” | Россия,.. |
| ***1038*** | Холодильник | 0 | 3100 | "BOSS" | ФРГ,… |
| ***1039*** | Стир. машина | 52 | 2500 | "BOSS" | ФРГ,… |

Отметим несколько недостатков такого подхода:

**Избыточность данных**. Обратите внимание, что фирма BOSS в общей таблице указана два раза. В больших базах данных при использовании такого подхода могут быть объединены десятки таблиц, и подобных повторений может быть многие тысячи. Результат - неэкономное расходование места на носителях данных и, как следствие, - большой расход времени на поиск нужной информации.

**Аномалия обновления данных**. Всякий раз, когда необходимо будет внести изменения в данные о фирме (например при изменении адреса), придется переделывать несколько записей. В варианте с двумя таблицами данные хранятся только в одном месте.

**Аномалия вставки записей**. В общей таблице отсутствует поставщик “Горизонт”, т.к. он реально не поставляет товары. Это означает, что учесть нового поставщика можно будет только после того, как в таблице появится какой-то его товар.

**Аномалия удаления**. Удаление из общей таблицы холодильника с кодом 1036 приведет к потере всех данных о поставщике “Бирюса”.

С помощью разбиения данных на таблицы эти недостатки исчезают, что дает экономию места на носителях данных и намного облегчает работу с ними. При этом само разбиение не является искусственным, а производится в соответствии с внутренней логикой объектов-сущностей, которые моделируются, поэтому облегчается не только оперирование с данными, но и общее понимание того, что содержится в базе данных. Процесс следования правилам разработки реляционных таблиц, позволяющим избежать перечисленных недостатков, называют **нормализацией**.

***5.2.3. Запросы к реляционным базам данных***

Реляционные БД выполняют три основных типа операций с таблицами при выполнении запросов:

**1. Проектирование** - создание новой таблицы путем выбора заданных столбцов из исходной таблицы (например, создание списка названий поставщиков).

**2. Выбор по условию** - создание новой таблицы путем выбора из исходной таблицы строк, удовлетворяющих заданным условиям (например, выбор из таблицы товаров холодильников, которые имеются на складе).

**3. Объединение** - создание новой таблицы путем выбора заданных столбцов из двух или более таблиц (например, создание таблицы видов товаров с указанием их цены и страны изготовления).

Важное свойство реляционной модели данных состоит в том, что результатом запроса всегда является новая таблица, и к ней может быть применен новый запрос. Это позволяет выполнять **вложенные запросы** и делает языки описания запросов очень мощным инструментом просмотра базы данных (пример простого вложенного запроса - после *объединения* таблицы товаров с указанием стран - изготовителей, можно *выбрать* товары, изготовленные за рубежом). При этом физически последовательности таблиц при выполнении вложенных запросов могут и не создаваться.

Кроме того, запросы позволяют **упорядочивать таблицы** (например, выбранные товары расположить в порядке возрастания цены), а также делать итоговые вычисления, используя операцию **группировки** (например, можно посчитать общую стоимость хранимых на складе холодильников, сгруппировав строки таблицы товаров по описанию товара).

В большинстве случаев пользователю даже не обязательно знать ЯОЗ, потому что для выбора из БД часто встречающихся, стандартных типов информации создаются специальные формы, с которыми легко работать. Однако для понимания того, что можно получить из базы данных, надо представлять, как работает язык описания запросов. Реляционные языки описания запросов можно разделить на две большие категории: **текстовые ЯОЗ** и **графические ЯОЗ**. Ниже на примерах рассматривается язык четвертого поколения SQL как наиболее широко используемый для определения информационных потребностей пользователей.

**5.3. Моделирование данных**

С точки зрения пользователя наибольшая проблема в организации и доступности информации ИС заключена в 3 вопросах:

какая информация содержится в ИС?

как информация организована?

каким образом пользователь может получить необходимую информацию?

В соответствии с действующими в настоящее время идеями разработки систем, ответы на эти вопросы лежат в плоскости моделирования данных-процесса построения концептуальной схемы, обеспечивающей единое определение данных в рамках одного предприятия, не ориентированное на какое-то конкретное их использование и не зависящее от того, как физически осуществляется хранение данных или доступ к ним.

**Моделирование данных** - это процесс определения базы данных с целью непротиворечивой интерпретации и установления взаимосвязей данных для их объединения, совместного использования и управления целостностью данных, адекватно отражающих функционирование организации. Задача, следовательно, состоит в том, чтобы надежно собирать и сохранять данные о всякой деятельности, которую организация хочет планировать, контролировать или оценивать. В процессе моделирования данных осуществляется идентификация сущностей, установление связей между сущностями и описание этих сущностей соответствующим набором атрибутов. Эксперты организации (менеджеры, бухгалтеры) обязательно вовлекаются в создание БД и при этом сталкиваются по крайней мере с двумя инструментами - REA-моделью данных и E-R-диаграммами, которые используются на этапе разработки БД.

***5.3.1. REA модель данных***

REA-модель данных была специально создана для разработки БД, предназначенных для учета операций в организациях. REA - это английские начальные буквы трех фундаментальных типов сущностей.

*Resources* **(ресурсы)**, приобретаемые и используемые организацией. Большинство ресурсов организации традиционно относят к ее активам. Это деньги, материально-производственные запасы, недвижимость и т.д. То есть активы, которые подвергаются учету. Однако для организации может быть важно учитывать и что-то другое, например заказы клиентов.

*Events* **(события)**, которые происходят в организации. В широком понимании - это любая деятельность организации, изменяющая состояние ресурсов. В REA модели событиями считаются не только традиционные бухгалтерские операции (продажи, покупки, выплата зарплаты и т.д.), но и другие операции, о которых организация хочет собрать данные (оформление заказов клиентов, стадии их прохождения и др.). Однако такие события должны напрямую влиять на ресурсы. Например, перенос данных из журнала в главную книгу не будет являться событием, если только какой-то из этих документов не рассматривается как ресурс организации, записи в котором подлежат учету.

*Agents* **(участники**) этих событий. Участники как сущности - это обычно группы людей, о которых организация собирает данные с целью лучше планировать, контролировать или оценивать их деятельность. Участники всегда вовлечены или имеют отношение к каким-то событиям. Например, продавцы совершают сделки продаж, кассиры принимают деньги, поставщики предоставляют товар, клиенты производят заказы и т.д.

Таким образом, REA-модель, с точки зрения организации, предназначена для описания учета как бухгалтерских, так и управленческих данных. Например, кроме даты продажи может учитываться и время дня, чтобы по множеству записей о продажах можно было планировать потребность в обслуживающем персонале в течение рабочего дня и тем самым получить возможность адекватно составлять скользящие графики работы служащих.

***5.3.2. E-R-диаграммы***

E-R-диаграммы (*Entity-Relationship*, Сущность-Отношение) предназначены для графического изображения схемы БД. Они показывают сущности в виде прямоугольников и отношения между ними в виде линий. Такой простой прием позволяет при разработке БД охватывать взором сложные схемы, иногда состоящие из десятков сущностей, скрывая детали, несущественные на определенном этапе. Для примера рассмотрим E-R диаграмму на рис.5.3, описывающую продажи в розничной торговле. Для упрощения в ней моделируются только два типа событий: оформление продажи и прием платежа.



*Рис.5.3 E-R- диаграмма, описывающая продажи*

REA-модель, на основе которой построена данная диаграмма, состоит из 7 сущностей, информация о которых подлежит учету. Поскольку события платежей и продаж являются сделками, то каждое из них связано с двумя участниками и одним предметом сделки. Каждое отношение на диаграмме имеет название-связку, которое позволяет просто понять его смысл (продажи кому? - клиентам, платеж в уплату за продажу и т.д. Продолжите сами для остальных отношений).

В дальнейшем каждая из сущностей будет описана в виде таблицы, содержащей ее атрибуты-столбцы и экземпляры-строки. Однако на E-R-диаграмме эти подробности скрыты, позволяя сосредоточиться на разработке концептуальной схемы БД.

На диаграмме имеются символы ‘1’ или ‘\*’ между каждой сущностью и отношением. Они описывают **характер отношения** между двумя сущностями. По характеру отношения бывают трех типов: один к одному (1:1), один ко многим (1:\*) и многие ко многим (\*:\*).

Характер отношений можно определить, рассматривая их смысл. Например отношение между кассирами и платежами состоит в том, что один кассир может принимать много платежей (\*). В то же время каждый отдельный платеж не может быть принят несколькими кассирами, а только одним (1).

Аналогично определяется характер отношения между товарами и продажами - многие ко многим (\*:\*), т.к. один вид товара может быть упомянут во многих сделках продаж (\*) и предметом каждой сделки может служить не один, а сразу несколько видов товаров (\*).

Характер отношения играет важную роль в моделировании деятельности организации с помощью реляционных БД. Чтобы это увидеть, рассмотрим возможные отношения между продажами и платежами.

Разработка E-R-диаграммы может быть разложена на 4 шага.

**Выявление необходимых сущностей**. Если мы пользуемся REA-моделью, то этот шаг всегда начинается с определения событий, связанных с деятельностью организации. Продажи и платежи, поставки и покупки, прием и доставка заказов, прием на работу и выплата вознаграждений персоналу, продвижения по службе - все это примеры событий, которые организация может учитывать для обеспечения лучшего принятия решений.

После выявления событий для каждого из них определяются ресурсы, состояние которых изменяется после того, как событие произошло.

И наконец выявляются участники каждого события. Важно помнить, что в REA-модели под термином “участник” подразумевается не конкретный человек, а его функция в организации. Поэтому один из кассиров и один из продавцов может быть одним и тем же лицом, а для учета персонала при этом потребуется выделение отдельной сущности.

При выявлении сущностей важно помнить, что в их число должны попасть только те, которые связаны с проблемами, решаемыми с помощью БД. Иначе организация рискует столкнуться с эффектом информационной перегрузки.

**Изображение сущностей на диаграмме**. Поскольку сущности делятся на три класса и события связаны как с ресурсами, так и с участниками, то прямоугольники для каждой сущности помещаются в три столбца (слева направо): столбец ресурсов, событий и участников. Это облегчает изображение отношений. Возможно, потом для удобства и придется делать перестановки, но только внутри столбцов.

**Выявление и изображение отношений на диаграмме**. При построении E-R-диаграммы не используются прямые отношения между ресурсами и участниками. Это обусловлено самой природой событий, происходящих в организации, поскольку само воздействие участника на ресурс является событием. Изображение ромбов и линий, связывающих две сущности для каждого отношения, несложная задача. Проблемы чаще возникают при подборе удачного названия для отношения. Возможно также, что на этом шаге придется переставить на диаграмме некоторые сущности, чтобы связи не перекрещивались.

**Определение характера отношений**. Характер отношений не является чем-то произвольным, а отражает их природу и определяется двумя факторами: политикой предприятия при проведении своих операций (пример - на рис. 5.4) и спецификой операций (например, отношение клиенты-продажи на рис. 5.3 - один ко многим - отражает тот факт, что любая продажа оформляется только на одного клиента).



*Рис. 5.4. Характер отношения между продажами и платежами*

Мы рассмотрели моделирование данных с помощью E-R-диаграммы на простом примере. Однако моделирование данных может быть сложным и длительным процессом, сопровождающимся преодолением проблем, связанных со спецификой отрасли и отдельного предприятия, и даже с различиями в терминологии, используемой в разных подразделениях организации.

Опишем 4 шага, необходимые для перевода E-R-диаграммы в схему реляционной базы данных.

**Создание таблиц для каждой сущности и каждого отношения (\*:\*).** Пример на рис.5.3. требует создания 9 таблиц - 7 для каждой из сущностей и 2 для отношений "продажи-товары" и "продажи-платежи". Создание таблиц для отношений (\*:\*) гарантирует выполнение требований нормализации. Каждой таблице дается имя по названию соответствующей сущности (в примере - товары, деньги, продажи, платежи, продавцы, клиенты, кассиры). Что касается имен таблиц, представляющих отношения, то они обычно составляются из имен связываемых сущностей (в примере - "продажи-товары" и "продажи-платежи"). Большинство СУБД накладывают ограничения на имена таблиц и полей, например, допускают использование только букв английского алфавита и некоторых символов.

**Определение атрибутов для каждой таблицы**. Эта работа, как и создание E-R-диаграмм, делается с помощью специалистов фирмы, знающих особенности циркулирующей в ней информации и потребности пользователей. Можно перечислить некоторые правила при определении атрибутов.

**Назначение первичных ключей каждой таблице.** Одним или двумя атрибутами в каждой таблице должны быть представлены первичные ключи. Для этого чаще всего используются цифровые ключи, например, номер счета, табельный номер работника, код товара, номер счет-фактуры и т.д. Обычно первичный ключ таблицы - это один из ее атрибутов. Однако для таблиц, представляющих отношения (\*:\*), он всегда состоит из двух атрибутов, которые представляют первичные ключи связанных этим отношением таблиц. Такой многоатрибутный ключ называется **составным ключом**. Например, первичный ключ таблицы "продажи-товары" будет состоять из атрибутов “номер счет-фактуры” (первичный ключ таблицы продаж) и “код товара” (первичный ключ таблицы товаров). Столбцы первичных ключей в таблицах 5.8 - 5.16. обозначены серым цветом.

**Назначение таблицам неключевых атрибутов**. Некоторые атрибуты (например дата и сумма продажи) нужны для правильной обработки данных и требуются для отчетности. Другие атрибуты приписываются сущностям, чтобы обеспечить информацию для более эффективного управления ресурсами организации (например, табельный номер работника нужен не только чтобы начислять комиссионные, но и чтобы оценивать работу служащих).

**Каждый атрибут в таблице должен характеризовать первичный ключ или быть внешним ключом**. Это правило уже обсуждалось, когда мы вводили понятия первичного и внешнего ключей при определении ограничений на структуру таблиц. Здесь оно означает, что вся информация, не описывающая непосредственно данную сущность, относится к другой сущности и находится в другой таблице, а внешний ключ обеспечивает ссылку на эту информацию. Например, в таблице платежей каждый платеж будет иметь атрибуты, описывающие его и только его - это номер (первичный ключ), дата, сумма платежа, но вместо указания плательщика, кассира и атрибутов счета в банке будут присутствовать внешние ключи - код клиента, табельный номер кассира и внутренний номер счета. Обратившись по этим ключам в соответствующие таблицы, всегда можно узнать, например, имя клиента и кассира, описание счета. По сути выполнение этого правила обеспечивает отсутствие избыточности в БД. В самом деле, если бы вместо кода клиента в таблице платежей присутствовало бы его описание, то фамилия Первушина, его адрес и кредитный лимит повторились бы два раза, поскольку он сделал две покупки.

**Реализация отношений (1:\*)**. К этому этапу реляционная база данных полностью моделирует только отношения (\*:\*). Отношения (1:\*) моделируются с помощью внешнего ключа. Для этого первичный ключ сущности, которая встречается в отношении один раз, используется как внешний ключ сущности множественной сущности. В нашем примере в табл.5.8. - 5.16 все внешние ключи предназначены для моделирования отношений (1:\*). Убедитесь в этом, сопоставив набор таблиц 5.8. - 5.16. с E-R диаграммой на рисунке 5.4. Это произошло потому, что на диаграмме нет отношений (1:1).

**Реализация отношений (1:1)**. Отношения (1:1) между сущностями моделируются с помощью вставки в одну из связанных таблиц внешнего ключа, который служит первичным ключом другой таблицы. В принципе, вставлять внешний ключ можно в любую из таблиц, но если мы имеем дело с отношением (1:1) между двумя событиями, то внешний ключ вставляется в таблицу более позднего события.

Итак, мы разобрали основные этапы моделирования данных с помощью реляционных баз данных. Знание основ этой работы необходимо экспертам организации, которые будут принимать участие в создании ИС в качестве консультантов и даже просто пользователей.

**5.4. Разработка баз данных**

Разработка базы данных - поэтапный процесс, в котором можно выделить 6 стадий (рис.5.5.). Экспертам предприятия приходится участвовать в этом процессе почти на всех его стадиях.

**Планирование**. Первая стадия - начальное планирование для определения потребностей и возможностей разработки новой системы. Цель - определить, является ли предлагаемая система технологически и экономически возможной. Если это так, то начинается новая стадия.



*Рис. 5.5. Стадии разработки базы данных*

**Определение требований** включает определение области применения предлагаемой базы данных, основных требований к программному и аппаратному обеспечению, а также потребностей пользователей. Область применения определяется в консультациях с руководством предприятия и отражает информационные потребности организации, ее стратегические цели и задачи. После этого собирается информация о таких факторах, как число пользователей и ожидаемый объем операций, которые используются для определения основных требований к программному и аппаратному обеспечению новой системы. Данные о потребностях пользователей собираются различными методами, например, с помощью интервью или анкетирования. Эти данные используются для предварительного определения отдельных представлений пользователей (внешних подсхем), которые бы отражали как требования обработки операций, так и требования процесса принятия решений. При разработке базы данных приходится принимать во внимание несколько требований.

**Полнота.** БД должна содержать все данные и отношения, нужные различным пользователям. Интересы пользователей и источников данных должны быть скоординированы.

**Адекватность.** Собираться и храниться должны только полезные и относящиеся к делу данные.

**Актуальность.** Хранимые данные должны постоянно обновляться, чтобы отразить текущее состояние дел.

**Точность.** В БД не должно быть ошибок и неточностей.

**Доступность.** Хранимые данные должны быть доступны для всех легальных пользователей в нужное им время.

**Эффективность.** На хранение данных должно тратиться не очень много ресурсов, а время обновления, извлечения данных и эксплуатация БД должно быть приемлемым.

**Безопасность.** БД должна быть защищена от потери данных, разрушения и несанкционированного доступа.

**Гибкость.** Возможные изменения в жизни организации не должны приводить к полной замене БД.

К сожалению, не всегда возможно добиться наилучших результатов по всем этим требованиям. Во многих случаях приходится идти на компромисс. Например, экономичность часто находится в противоречии с гибкостью и доступностью БД. Поэтому при разработке БД пытаются достичь возможного баланса между целями.

**Логическое проектирование**. На этом этапе завершается разработка внешних схем БД. Требования различных пользователей и прикладных программ переводятся на язык концептуальной схемы, используя REA-модель и E-R-диаграммы. Часто на этом этапе выделяются подсистемы будущей БД, отвечающие за различные информационные нужды, например, подсистемы продаж, закупок, кадров, производства и т.д. Это делается для удобства разработки и эксплуатации БД. Кроме того, на этом этапе определяются первичные и вторичные ключи, разрабатывается справочник данных.

**Физическое проектирование** состоит в переводе концептуальной разработки в физически существующие структуры хранения данных и работающих с ними программ. Здесь концептуальная схема переводится во внутреннюю, создается справочник данных, определяются способы физического хранения и доступа к БД, в том числе принимаются решения об использовании индексов.

**Внедрение и эксплуатация**. Внедрение состоит в том, чтобы подготовить, инициировать и запустить все процессы, связанные с эксплуатацией базы данных. Это включает преобразование существующих данных в формат файлов новой базы данных, разработку новых прикладных программ и модификацию существующих, обучение пользователей, тестирование работы БД, переход на ее использование. Стадия эксплуатации включает не просто реальное использование БД, но и наблюдение за ее работой и выявлением неудовлетворенности пользователей, чтобы определить, что необходимо усовершенствовать.По различным причинам базы данных “стареют”, и если простой модификации становится недостаточно, то возникает потребность разработки новых принципов работы. На этом жизненный цикл БД начинается сначала.

**Роль экспертов организации**. Эксперты организации должны быть вовлечены во все стадии разработки БД. На стадии планирования они предоставляют информацию для оценки возможностей и участвуют в принятии решения по этому вопросу. На стадии определения требований и логического проектирования они участвуют в определении информационных потребностей пользователей, разработке схем, словаря данных, мер контроля. Во время внедрения - в тестировании БД и прикладных программ. Наконец, при эксплуатации они используют БД и помогают принимать решения по ее управлению.

*Таблица 5.18*

 *Факторы, исследуемые при инспекции после внедрения.*

|  |  |
| --- | --- |
| **Фактор** | **Вопросы** |
| Цели и задачи | Помогает ли система организации выполнять ее цели, задачи и миссию? |
| Удовлетворение пользователей | Довольны ли пользователи системой? Что бы они хотели изменить или улучшить? |
| Выгоды | Что выиграли пользователи при применении системы?Достигнуты ли ожидаемые выгоды для организации? |
| Затраты | Сравнимы ли действительные затраты с ожидаемыми? |
| Надежность | Надежна ли система? Если система работает ненадежно, то что является причиной? |
| Точность | Производит ли система точные и полные данные? |
| Своевременность | Своевременно ли система предоставляет информацию? |
| Совместимость | Совместимы ли программное, аппаратное обеспечение, данные и процедуры с другими системами? |
| Контроль и безопасность | Защищена ли система от непреднамеренных ошибок, мошенничества и несанкционированного доступа? |
| Ошибки | Адекватны ли процедуры обработки ошибок? |
| Обучение | Достаточно ли подготовлены пользователи и персонал, чтобы поддерживать использование системы? |
| Коммуникации | Соответствуют ли коммуникационные системы потребностям ИС? |
| Организационные изменения | Полезны или вредны организационные изменения, сделанные при внедрении системы? Если вредны, то как решить эту проблему? |
| Документация | Является ли документация по системе полной и точной?  |

Последнее событие при разработке системы - утверждение пользователями отчета об инспектировании ИС. С этого момента управление системой передается в руки подразделения, занимающегося обработкой данных. Система начинает свою самостоятельную жизнь.

Однако работа над новой системой на этом не завершается. Исследования практики применения ИС говорят, что в период разработки делается только 20-30% работы, остальные 70-80% остаются на долю **обслуживания системы**. Большая часть затрат на обслуживание - это затраты на модификацию программ и обновление различных компонентов ИС. Это происходит по разным причинам, к которым можно отнести исправление не обнаруженных ранее ошибок, усовершенствования системы, изменения в структуре и деятельности организации, изменения налогового и другого законодательства и т.п.