**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ**

КАЗАНСКИЙ ГОСУДАРСТВЕННЫЙ
АРХИТЕКТУРНО-СТРОИТЕЛЬНЫЙ УНИВЕРСИТЕТ

ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ

Учебно-методическое пособие
к курсу «Проектирование информационных систем»
для студентов направления подготовки 09.03.02
«Информационные системы и технологии»

Казань
2018

УДК 338.24
ББК 65.9(2)212.14
Ш2

Ш2 Учебно-методическое пособие «Проектирование информационных систем» / Сост. Шамсутдинов Т.Ф. Казань: КГАСУ, 2018. - 110 с.

Печатается по решению Редакционно-издательского совета Казанского государственного архитектурно-строительного университета

В учебно-методическом пособии приведены основные понятия, компоненты и классификация информационных систем и пользователей, раскрываются этапы и принципы, а также особенности проектирования информационных систем; рассмотрены способы хранения, ограничения к файловым системам, преимущества файловых систем в сравнении с базами данных; рассмотрены процессы жизненного цикла разработки программного обеспечения с выделением основных, вспомогательных и организационных процессов; приведены основные подходы объектно-ориентированного программирования и принципы проектирования пользовательского интерфейса.
Пособие рассчитано на студентов 4-го курса направления 09.03.02 Информационные системы и технологии.

Рецензенты:

Профессор кафедры ИТиСАПР КГАСУ, к.т.н. Е.М. Удлер
Доцент кафедры АСОИУ КНИТУ-КАИ им. А.Н. Туполева, к.т.н. И.С. Ризаев

УДК 338.24
ББК 65.9(2)212.14

Оглавление

Тема 1. Основные понятия, компоненты и классификация информационных систем (ИС) и их пользователей

1.1 Определение информационной системы (ИС)

Информационная система – это совокупность взаимосвязанных средств, методов, позволяющих выполнять сбор, обработку и сохранять информацию, а также ее предоставлять для принятия управленческих решений.

Информационная система – это система, основанная на использовании компьютерной вычислительной техники, позволяющая выполнять процессы, связанные с хранением, поиском, обработкой и передачей больших объемов информационных данных определенной предметной области.

Информационная технология – это процесс, основанный на использовании методов и инструментов, позволяющих выполнять сбор, обрабатывать и передавать данные для того чтобы получить информацию нового качества, позволяющую описать состояние определенного объекта или явления.

Свойства информационной системы приведены в таблице 1.1.

Таблица 1.1 – Свойства информационной системы

Свойство	Характеристика
Интегрируемость	Информационную систему можно дополнить новыми компонентами, модулями или подсистемами
Масштабируемость	Информационная система позволяет расширить системные ресурсы для увеличения производительности и функциональности
Адаптивность	Информационная система может быть использована для автоматизации определенной предметной области
Управляемость	Возможность гибкого управления информационной системой
Целостность	В информационной системе все компоненты представлены в виде единого целого
Безопасность	В информационной системе предусмотрены механизмы для обеспечения безопасности данных

Информационная система может осуществлять работу по правилам разомкнутой или замкнутой системы управления.

Структурная схема разомкнутой информационной системы приведена на рисунке 1.1.

Рисунок 1.1 – Структурная схема разомкнутой информационной системы

В разомкнутой информационной системе в состав аппаратной и программной части входят подсистемы, позволяющее принимать информацию от источника входной информации, выполнять ее преобразование и хранение с последующим выводом и передачей потребителю информации.

В качестве примера разомкнутой информационной системы можно привести компьютерную справочную библиотечную систему. В этой системе каждый читатель по необходимой тематике может подобрать информацию и после ее получения завершает работу с системой.

В замкнутой информационной системе предусмотрены системы ввода, обработки и вывода информации обслуживающему информационную систему персоналу с организацией обратной связи, что видно из рисунка 1.2.

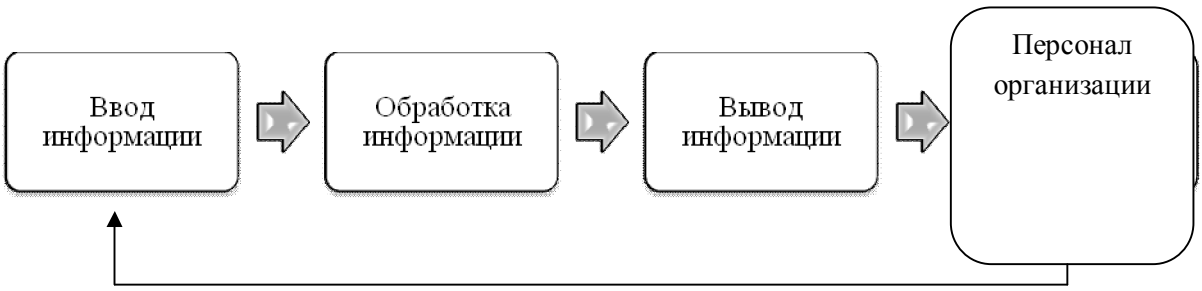

Рисунок 1.2 – Структура замкнутой информационной системы

В этом случае информационная система ориентируется на потребности определенного пользователя, то есть информация, полученная от пользователя, обрабатывается в системе, с предоставлением ему результатов.

В качестве примера замкнутой информационной системы можно привести информационную систему железнодорожной кассы. На первоначальном этапе кассир предоставляет необходимую информацию пользователю о количестве билетов, которые возможно купить. Как только билет продан покупателю, кассир изменяет статус билета. В этой замкнутой информационной системе обратная связь представлена в виде уведомления о продаже билетов.

1.2 Виды обеспечения работы информационной системы

Одной из наиболее важных частей информационной системы выступает информационное обеспечение, направленное на выдачу достоверных данных, позволяющих принимать управленческие решения.

Информационное обеспечение является совокупностью единой системы кодирования и классификации информации, представлено унифицированными документами, схемами информационных потоков находящимися в постоянном обмене, а также методологией построения баз данных.

Структура информационного обеспечения ИС приведена на рисунке 1.3.

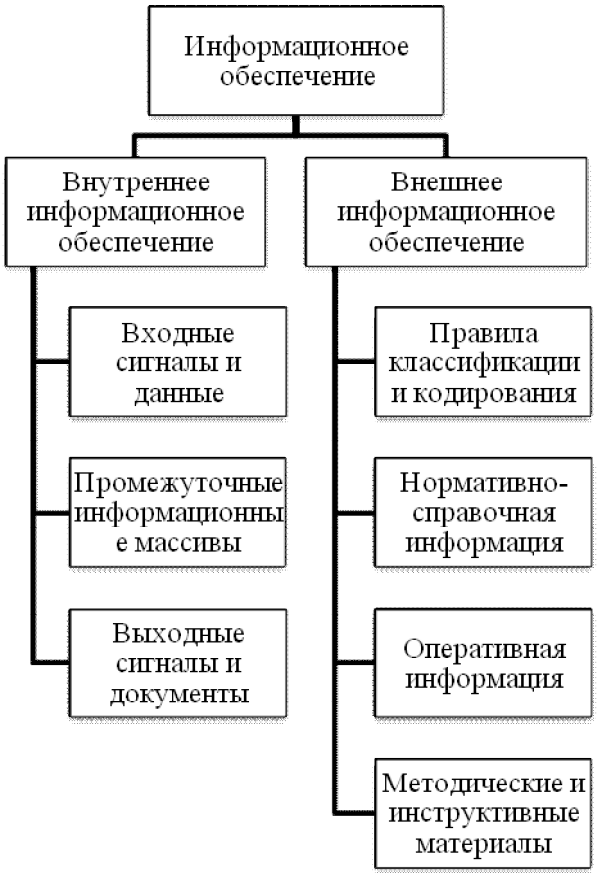

Рисунок 1.3 – Структура информационного обеспечения ИС

Унифицированные системы документооборота применяются на федеральном, региональном и муниципальном уровнях. Их главной целью является предоставление показателей, позволяющих классифицировать различные сферы общественного производства.

Схемы информационных потоков применяется для отражения маршрутов движения информации, ее объемов, мест возникновения первичной информации и применения полученных результатов. За счет анализа структуры вырабатываются мероприятия по совершенствованию всей системы управления.

Схема информационных потоков позволяет выявить объем информации, исключить дублирующиеся и не используемую информацию, ее классифицировать и рационально представить.

Методология построения баз данных основана на теоретических основах проектирования и включает в себя этапы обследования всех функциональных подразделений и создание концептуальной информационно логической модели. Для того чтобы создать информационное обеспечение определяется движение информации от момента возникновения до использования, совершенствуется система документооборота, создаются массивы информации, предусматривающие использование современного технического обеспечения [8].

Документация условно разделяется на три группы:

- общесистемную документацию, в состав которой входят отраслевые государственные стандарты по техническому обеспечению;
- специализированную документацию, в состав которой входит комплекс методических материалов позволяющих описать основные этапы разработки технического обеспечения;
- нормативно-справочная информация, позволяющая выполнять расчеты по техническому обеспечению.

Техническое обеспечение представлено набором технических средств, позволяющих осуществлять работу с информационной системы системой, а также документация по использованию этих технических средств.

Характеристики основных видов технического обеспечения приведены на рисунке 1.4.

Рисунок 1.4 – Характеристики основных видов технического обеспечения

К техническим средствам относятся персональные компьютеры различных моделей, устройства для сбора, обработки и передачи информации по линиям связи, организационная техника и устройства автоматического съема информации.

В настоящее время для организации технического обеспечения используется централизованная и децентрализованная формы. Централизованное техническое обеспечение основано на использовании в информационной системе вычислительных центров или серверов. Децентрализация технических средств основана на реализации функциональных подсистем на персональных компьютерах и на рабочих местах.

Математическое программное обеспечение представлено в виде совокупности математических моделей, методов функций алгоритмов и программ, позволяющих реализовать цели и задачи информационной системы, а также нормального функционирования набора технических средств.

В состав математического обеспечения входят средства моделирования процессов управления, позволяющие решать задачи управления, методы математического программирования и теории массового обслуживания.

Программное обеспечение представлено общественными и специальными программными продуктами, а также технической документацией.

Общественное программное обеспечение включает программы, которые ориентированы на пользователей и позволяют решать не типовые прикладные задачи по обработке информации. Они позволяют расширить функциональные возможности компьютеров, средств контроля и управления процессами обработки данных.

Специальное программное обеспечение представлено в виде программ, позволяющих решать определенные задачи пользователя. Оно представлено пакетами прикладного программного обеспечения, реализующего модели различной степени адекватности и позволяющие отразить функционирование реального объекта.

Классификация специального программного обеспечения приведена на рисунке 1.5.

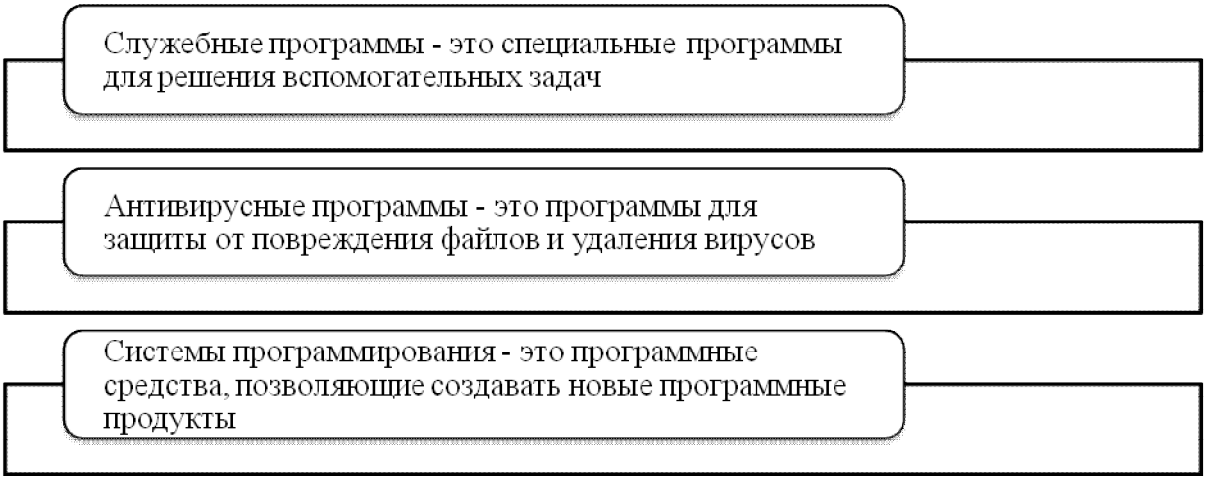

Рисунок 1.5 – Классификация специального программного обеспечения

Техническая документация на разработку программного обеспечения включает детализацию задач, задание на разработку алгоритмов, экономико-математические модели, контрольные примеры.

Организационное обеспечение является совокупностью средств и методов, позволяющих регламентировать взаимодействие между работниками и техническими средствами в процессе разработки и эксплуатации информационных систем.

Организационное обеспечение позволяет реализовать следующие функции:

- проанализировать существующую систему управления организацией, в которой будет внедрена информационная система, выявлять задачи которые подлежат автоматизации;
- подготавливать к решению задачи на персональном компьютере, в том числе разработку технического задания на проектирование информационной системы и технико-экономическое обоснование эффективности;
- разрабатывать управленческие решения, а также выбирать методологии решения задач, позволяющих повысить эффективность системы управления.

Организационное обеспечение представлено в виде совокупности правовых норм, которые определяют создание и функционирование информационной системы, регламентирует порядок получения, преобразования и использования данных.

В правовом обеспечении выделяется общая часть, которая позволяет регулировать работу любой информационной системы и локальная часть для определения функционирования определенной информационной системы.

Правовое обеспечение этапов работы информационной системы представлено статусом, правами и обязанностями, правовыми положениями и порядком создания и использования информационной системы.

1.3 Классификация пользователей и разработчиков ИС

Пользователи информационной системы разделяются на несколько категорий, среди которых можно выделить:

случайного пользователя – это пользователь, который осуществляет работу с информационной системой для удовлетворения личностных потребностей;

- конечного пользователя или потребителя информации – это лицо или группа лиц, которые осуществляют повседневную работу с информационной системой, автоматизирующей определенную

предметную область или бизнес-процесс. Они не являются специалистами по разработке программного обеспечения, а выполняют свои должностные обязанности и за счет использования ИС автоматизируют часть своей работы;

- обслуживающего персонала ИС – это специалисты по разработке программного обеспечения, администрирования ИС и базы данных, а также бизнес-аналитиков.

Администратор – это специалист, который обеспечивает защиту информационной системы, повышает ее производительность и эффективность работы, координирует процессы по выбору способов сбора обработки и предоставления информации пользователям.

Системный программист – это специалист, который осуществляет разработку и сопровождение программного и математического обеспечения информационной системы.

Прикладной программист – это специалист, который выполняет разработку программного обеспечения для обработки запросов пользователей и организации взаимодействия с базой данных.

Бизнес-аналитик – это специалист, который выполняет разработка математической модели предметной области, учитывая потребности конечных пользователей, бизнес-модели и формирует задачи для системных и прикладных программистов.

1.4 Классификация ИС

В настоящее время информационные системы классифицируют по различным признакам, имеющим отличия в ориентации, уровне управления, сфере функционирования, характере процесса управления, архитектуру, способы доступа к системе.

Рассмотрим классификацию информационных систем по целевой функции, в которой выделяют:

- экономические информационные системы (ЭИС);
- системы поддержки принятия решений (СППР);
- информационно-вычислительные системы;
- информационно-справочные системы;
- корпоративные информационные системы.

Экономические информационные системы (ЭИС) представляют собой системы направленные на предоставление и обработку данных на различных уровнях управления в области экономики. В ЭИС выделяют две группы информационных систем: ЭИС для производственной сферы и ЭИС для непроизводственной сферы.

Отдельной категорией экономических информационных систем выступают автоматизированные системы для бухгалтерского учета,

позволяющие автоматизировать ведение бухгалтерского и налогового учета.

Виды автоматизированных систем бухгалтерского учета приведены на рисунке 1.6.

Пакеты "Мини-бухгалтерия"	•Программы 1С: Бухгалтерия", "Турбо-Бухгалтер", "Фолио"
Пакеты "Интегрированная бухгалтерская система"	•Программы 1С: Предприятие, Парус, Галактика
Пакеты Комплексная система бухгалтерского учета	•АРМ главного бухгалтера, АРМ бухгалтера-расчетчика, АРМ бухгалтера по заработной плате
Корпоративные системы	•Интегрированные SAP-системы

Рисунок 1.6 – Виды автоматизированных систем бухгалтерского учета

Системы поддержки принятия решений (СППР) – это аналитический вид информационных систем, направленный на обеспечение возможностей получения представления о текущем состоянии системы управления, иметь возможность ее прогнозирования и развития.

Информационно-вычислительные системы применяют в сфере проведения научных исследований, для автоматизации сложных расчетов. Они могут выступать одним из компонентов СППР.

В настоящее время в деятельности многих предприятий свое распространение получили информационно-справочные системы (ИСС), позволяющие выполнять сбор, хранение, поиск информации, имеющей справочный характер.

Информационные системы, позволяющие автоматизировать все функции управления, охватить весь жизненный цикл деятельности предприятия, обеспечить комплексную автоматизацию бизнес-процессов получили название интегрированных ИС.

Вопросы для самопроверки по теме 1

1.Что такое информационная система?
2. Чем отличается информационная система от информационной технологи?
3. Перечислите свойства информационной системы?
4. В чем отличие замкнутой от разомкнутой информационной системы?
5. Перечислите пользователей информационной системы
6. Кто входит в группу разработчиков информационной системы?
7. На какие виды разделяют информационные системы?

Тема 2. Этапы, принципы и особенности проектирования ИС

2.1 Этапы проектирования ИС

Каждый проект независимо от объемов работ и сложности, которые необходимы для его выполнения предусматривает реализацию нескольких этапов, представленных в виде стадий развития проекта от момента его возникновения до полного завершения.

Количество этапов и их содержание определяются условиями и целями реализации проекта, а также профессиональными навыками команды разработки.

Основные этапы разработки проекта приведены на рисунке 2.1.

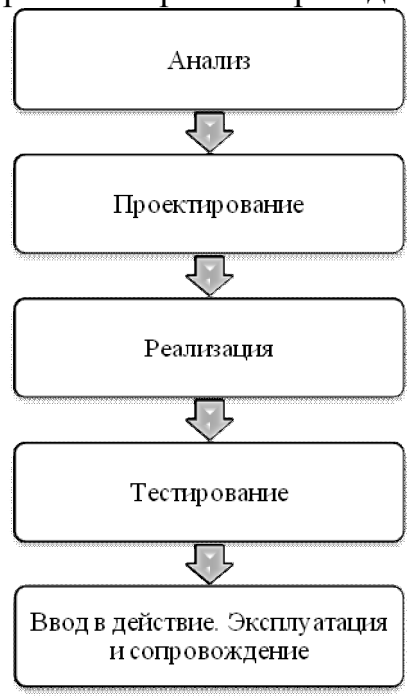

Рисунок 2.1 - Основные этапы разработки проекта

На первоначальном этапе выполняется анализ. На этом этапе выполняется формирование требований к информационной системе, выбор ответственных лиц, формализация задач.

Также на этом этапе выполняется анализ деятельности организации, формируются требования к информационной системе, корректно и точно отражаются цели и задачи организации заказчика, вместе с требованиями пользователей на этапе анализа разрабатывается логический проект информационной системы.

За счет логического проектирования определяется концептуальная модель данных, формируется список входных данных, процессы, а также список выходных данных.

На этапе анализ выполняется моделирование данных, основанное на описании объектов и их атрибутов, а также установлении связей между сущностями и представления их в виде ER-диаграммы.

Описание и документирование всех преобразований осуществляется с использованием средств моделирования, позволяющих построить модели бизнес-процессов, получить схемы информационных моделей. Конечной целью этого этапа является отражение всех бизнес-процессов, формирование оборудования и программных средств которые будут использоваться для разработки информационной системы.

Второй этап - это проектирование. На этапе проектирования выполняется формирование модели данных. В качестве исходных данных используют результаты предыдущего этапа.

Полученная в результате анализа информационная модель на первоначальном этапе преобразовывается в логическую модель, а затем физическую модель. Параллельно с разработкой схемы база данных осуществляется проектирование процессов для того чтобы получить спецификации модулей информационной системы.

Можно сказать, что эти два процесса проектирования взаимосвязаны между собой, поскольку часть бизнес-логики должна быть реализована в базе данных с формированием системы ограничений, обеспечением целостности триггеров и хранимых процедур.

При проектировании модулей выполняется определение интерфейсов программ: меню, видов окон, горячих клавиш и связанных с ними вызовов. В результате этого этапа получается схема базы данных, включающая ER-модель, разработанную на этапе анализа, набор спецификации модулей, на базе моделей функций. Также на этом этапе выбирается платформа и операционная система, система управления базами данных. Этап проектирование завершается получением технического проекта информационной системы.

На этапе реализации создаются компоненты программного обеспечения информационной системы, устанавливаются технические средства, разрабатывается эксплуатационная документация.

После этапа реализации осуществляется этап тестирования, который сопровождается выполнением набора тестов, позволяющих обнаружить отказы модулей в случае возникновения жесткий сбоев, тестов имитации отказов, наработки на отказ, системный тест позволяющий проверить функциональную систему, а также приемо-сдаточные испытания.

На пятом этапе выполняются работы по вводу в действие информационной системы, ее эксплуатации и сопровождении.

После ввода в действие информационной системы осуществляется обучение конечных пользователей, вносятся корректировки в модули, в случае их обнаружения, формируются требования к службе сопровождения системы.

2.2 Основные принципы проектирования ИС

В работах Академика Глушкова В. М. вместе с практическими рекомендациями по разработке информационной системы отражены основные принципы, приведенные на рисунке 2.2.

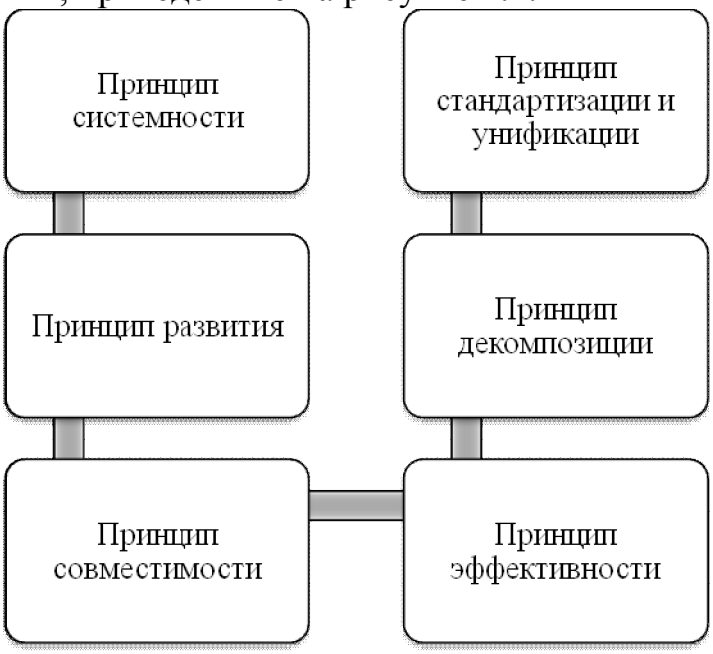

Рисунок 2.2 – Принципы проектирования ИС

Принцип системности достаточно важен на этапах разработки, функционирования и развития информационной системы. Он представляет объект автоматизации в виде единого целого. При этом устанавливаются направления деятельности объекта автоматизации, выполняемые им функции, типы связей между структурными элементами.

В основе принципа системности находится методы макро- и микроанализа объекта автоматизации, анализ структурных составляющих с целью установления функциональных характеристик и связей между элементами внешней и внутренней среды.

При проектировании информационной системы учитывается организационная структура управления объекта автоматизации, ее гибкость и возможность учитывать изменяющиеся условия внешней среды.

Практическим значением системного принципа является то, что он позволяет не только определить интересы разработчиков системы, но и на основании проведения моделирования оценить поведение разрабатываемой информационной системы.

Принцип развития основан на том что, в разрабатываемой информационной системе должна быть предусмотрена возможность постоянного обновления функций в разрезе видов ее обеспечения. Его сущностью является то, что постоянно развивающиеся управленческие и производственные процессы с течением времени усложняются и это

приводит перестройке структуры объекта автоматизации, создает необходимость увеличения производительности вычислительных мощностей, добавления новых программных и технических средств, позволяющих ускорить процессы автоматизации, расширить информационный фонд.

Информационной принцип на всестороннее изучение и детализацию информационных потоков, которые сопровождают процессы управления объекта автоматизации. При этом информация представляется в содержательном аспекте, синтаксическом, практическом. Представление информации в различных аспектах в дальнейшем используется для проектирования информационной системы, а также выбора средств для обработки и защиты информации.

Следует отметить, что в настоящее время на информационном подходе основан объектно-ориентированный метод моделирования информационных систем.

Принцип совместимости основан на организации взаимодействия информационных систем различных видов. Поэтому при проектировании обеспечивается системное единство в использовании методического инструментария для последующего решения проблем, технической и информационной совместимости разрабатываемой информационной системы с другими системами.

Принцип стандартизации и унификации основан на использовании унифицированных, типовых и стандартизированных элементов для обеспечения работы информационной системы. В основе этого принципа находится сокращение трудовых, временных и стоимостных затрат на разработку информационной системе

Принцип декомпозиции предусматривает разделение процесса разработки информационной системы на части и определение комплексов работ для ее создания. За счет этого принципа также изучаются особенности компонентов информационной системы.

Принцип эффективности направлен на соблюдение соотношения между затратами понесенными на разработку информационной системы и полученным эффектом от ее работы.

2.3 Предпроектное обследование объекта автоматизации

Предпроектное обследование объекта автоматизации относится к одному из важных и определяющих этапов проектирования информационной системы. Как правило, длительность предпроектного обследования объекта автоматизации находится в пределах от 1 до 2 недель. В течение этого периода бизнес-аналитиком исследуется не более 2-х или 3-х видов деятельности предприятия. Также выполняется сбор

информации для создания полной бизнес-модели организации на основании изучения документированных информационных потоков и функций структурных подразделений, а также проведение анкетирования, опросов и интервью.

На первоначальном этапе работы по обследованию объекта автоматизации представлены в виде комплекта документов и включают:

1. Сводную информацию о деятельности объекта автоматизации, в том числе информацию о финансовой, управленческой и производственной деятельности, а также отчетности и учетной политике.
2. Регулярный документооборот объекта автоматизации, включающий реестры входящий, внутренний и исходящей информации.
3. Сведения об ИТ-инфраструктуре предприятия, а также ответственных лицах.

Результатом предпроектного обследования объекта автоматизации выступает отчет по экспресс обследовании объекта автоматизации.

Входящие в отчет сведения могут быть представлены в виде текстового описания, табличных или графических данных.

На основании проведения предпроектного обследования объекта автоматизации выявляются требования к будущей системе, определяется структура организации, целевые функции, производится анализ существующих средств автоматизации предприятия.

Полученная информация в результате предпроектного обследования объекта автоматизации в дальнейшем анализируется с применением методов объектного и структурного анализа, построении моделей деятельности организации.

2.4 Референтная модель

Референтная модель – это модель, отражающая типовой бизнес-процесс какого-либо предприятия и ее можно использовать для разработки или реорганизации бизнес-процесса других предприятий.

Можно сказать, что референтная модель является эталонной моделью организации бизнеса. Ее можно использовать для выбора направлений совершенствования деятельности организации, внедрения или разработки информационных систем.

На основании референтной модели выполняется разработка собственной модели с рекомендуемым шаблоном процессов и функций, приведенным на референтной модели.

Референтная модель представлена в виде логически взаимосвязанных функций, для каждой из которых можно указать исполнителя, входные и выходные документы.

2.5 Постановка целей и задач разработку ИС

При разработке любой информационной системой основными участниками процесса являются постановщик задачи, который представляет интересы будущего пользователя и разработчик или специалист, выполняющий операции по разработке программного обеспечения.

Для решения простейших задач по постановке целей и формирования задач разработки ИС можно использовать программные средства общего назначения, позволяющие совместить эти две стороны в одном.

Любая постановка задачи включает следующие этапы:

- формулировка целей, то есть определение необходимости разработки программного обеспечения;
- моделирование, позволяющие получить представление о предметной области, определить пути решения задач и сформулировать желаемые результаты;
- словесное или лингвистическое описание процесса постановки задачи с перечислением исходных данных и получением желаемых форм представления результатов для того чтобы решить задачу;
- формализованное описание процесса с углублением уровня формализации;
- формулировка критериев эффективности работы разрабатываемой информационной системы, то есть количественных показателей, позволяющих достичь поставленных целей и разработка алгоритма решения, то есть описание последовательности действий, которые необходимо будет выполнить с входной информацией для того чтобы получить желаемое результаты.

2.6 Определение путей повышения эффективности объекта автоматизации

Для повышения эффективности объекта автоматизации предприятиями формируются различные направления в области автоматизации.

Многими предприятиями для повышения эффективности объекта автоматизации выполняется обновление материально-технической базы, производится обучение сотрудников работы с информационной системой, привлекаются внешние специалисты, оказывающие техническую поддержку по использованию разрабатываемого продукта.

Пользователями для повышения эффективности объекта автоматизации выполняется обновление сетевого оборудования, персональных компьютеров и серверов для хранения баз данных, чтобы в

дальнейшем увеличить скорость обработки запросов и ускорить процессы принятия управленческих решений.

За счет проведения комплексной автоматизации объекта:

- снижается трудоемкость, поскольку большую часть работы выполняет информационная система, а для сотрудников создаются резервы времени на разработку инновационных решений;
- снижаются сроки выполнения работ в целом;
- повышается безопасность использования оборудования, снижаются риски появления ошибок;
- повышается эффективность использования оборудования, а также уровень качества работы специалистов;
- систематизируется документооборот предприятия за счет накопления данных в базе данных.

За счет автоматизации, обновления аппаратного и программного обеспечения в целом повышается конкурентоспособность и прибыль объекта автоматизации, а также профессиональный уровень ИТ-специалистов.

Вопросы для самопроверки по теме 2

1. Перечислите основные этапы разработки проекта
2. Перечислите и охарактеризуйте основные принципы проектирования ИС
3. Какие документы необходимо использовать для обследования объекта автоматизации?
4. Какую модель можно отнести к эталонной модели и почему?
5. Какие этапы включает постановка задачи?
6. Чего позволяет достичь комплексная автоматизация предприятия?

Тема 3. Проектирование информационного обеспечения ИС: построение модели и операционной диаграммы информационных потоков

3.1 Порядок построение операционной диаграммы информационных потоков

Для составления структурных схем информационных систем применяется графический метод операционных диаграмм. С помощью операционной диаграммы отражаются основные операции, выполняемые в границах информационной системы, а также взаимосвязи между этими операциями и объектами, находящимися вне системы. Они позволяют представить информационную систему с различным уровнем детализации.

Для дальнейшей детализации проекта информационной системы на каждом блоке составляется отдельная диаграмма. Основным правилом составления диаграмм является то, что потоки данных, которые выходят за границы блока, не должны иметь обрывов.

Если вводится какое-либо изменение, его отражают только на диаграмме этого уровня и оно не распространяется на диаграммы верхних уровней. Однако, в случае изменения в содержании потока данных, которые пересекает границу операции, вводится необходимая корректировка на операционной диаграммы предшествующего уровня. Это создает непротиворечивость и согласованность между разработанными спецификациями.

На операционных диаграммах могут быть отображены все материальные, информационные и управляющие потоки. При этом информационные потоки должны соответствовать движению документов, учетных данных и отображать информацию, например, по реализации заказа.

Управляющие потоки наиболее часто представляются в виде внешних управляющих воздействий. Для удобства проведения анализа на операционных диаграммах информационные, управляющие и материальные потоки отображаются различными типами линий. Для отображения информационных потоков используют тонкие непрерывные линии, а для материальных потоков – жирные непрерывные линии. Управляющие потоки отображают пунктирными линиями.

Вместе с потоками на операционной диаграмме необходимо показать их наименование. Причем для потоков данных должны быть показаны не только названия передаваемых документов, отражает содержание последних.

Как правило, указывается название всех потоков, что позволяет исключить ошибки и дублирование, а при разветвлении потока приводится наименование каждой ветви. Если ветвь потока не имеет название, то по умолчанию принимают, что она принадлежит исходному потоку.

Для обозначения операций на операционных диаграммах используются прямоугольники. Любая операция связана с преобразованием одного или нескольких входных потоков в один или несколько выходных потоков. Это позволяет выполнить фиксацию дополнительных сведений о разрабатываемой информационной системе.

В некоторых случаях необходимо сохранять данные в течение некоторого промежутка времени. Хранилища на операционных диаграммах отражаются в виде овалов. При этом внутри каждого такого овала переводится название соответствующего объекта, а при необходимости и его числовой идентификатор. Для упрощения диаграммы в некоторых случаях выполняется дублирование хранилищ с отметкой его в виде незамкнутого овала и добавлением вертикальной черты.

Границы исследуемой предметной области отражаются на композиционной диаграмме. При этом каждый овал представлен в виде определенного внешнего объекта или группы объектов, отражающих подразделение организации, поставщиков, покупателей.

На операционной диаграмме принят следующий порядок нумерации. Композиционная диаграмма верхнего уровня не имеет нумерацию, операции на диаграмме верхнего уровня нумеруются цифрами 1,2 и так далее. Каждую операцию можно разделить на отдельные составляющие, которые представляются на следующих уровнях.

Декомпозиционные диаграммы нижнего уровня имеют двойную нумерацию. Первая цифра отражает номер операции верхнего уровня, а второй номер – операции нижнего уровня.

3.2 Функциональная модель информационных потоков

Функциональная модель SADT (IDEF0) используется для того чтобы проанализировать функциональную структуру объекта, то есть выполняемые ею действия и связи между ними.

Основными элементами этого метода выступают графическое представление или блочное моделирование, включающее диаграммы, позволяющие отобразить функции блока, интерфейсы входа, выхода, управления и механизма.

Характеристика основных элементов диаграммы в нотации IDEF0 приведена в таблице 1.

Таблица 3.1 - Характеристика основных элементов диаграммы в нотации IDEF0

Элемент	Значение элемента	Требования к использованию элемента
Функциональный блок	Применяется для отображения в виде прямоугольника. С его помощью представляются функции, которые можно определить как процесс, действия или операции	1. Каждый функциональный блок имеет уникальный идентификационный номер, отображение которого производится в правом нижнем углу 2. Название функционального блока приводится в отглагольном наклонении
Интерфейсная дуга	Указывает на диаграмме как однонаправленная стрелка. С ее помощью представляются данные, материальные объекты, которые связываются с функциями	1. Каждая интерфейсная дуга должна иметь уникальное имя 2. Название интерфейсной дуги показано в виде оборота существительного 3. В качестве источника может использоваться только выходная стрелка блока

Результатом использования метода SADT (IDEF0) является модель, включающая диаграммы, глоссарий, различные фрагменты текста, имеющие ссылки. Среди особенностей этого метода является постепенное выделение различных уровней детализации.

Для построения модели в нотации SADT (IDEF0) выполняются следующие действия:

- осуществляется сбор информации об объекте автоматизации с определением границ;
- определяются цели и точки зрения модели;
- разрабатывается, обобщается диаграмма и выполняется ее декомпозиция;
- осуществляется критическая оценка с комментариями и лицензированием работы.

Разработка диаграммы начинается с ее представления в виде одного функционального блока или контекстной диаграммы, имеющего дуги, интерфейсы. Поскольку единственный блок отражает систему как единое целое, то его имя является обобщающим.

После этого контекстная диаграмма, декомпозируется на другой диаграмме с выделением нескольких блоков, соединенных между собой интерфейсами-дугами. Эти блоки позволяют определить основные подфункции исходного блока.

Диаграмма декомпозиции представлена в виде полного набора подфункций, каждый из которых представлен в виде блока, границы которого отображается с использованием интерфейсных дуг. Каждая из

этих подфункций может быть декомпозирована и определить большую детализацию блока.

Во всех случаях каждая подфункция может включать только те элементы, которые входят в исходную подфункцию. Кроме того, модель не может опустить каких-либо элементов, в том числе родительского блока его интерфейса, к нему нельзя добавить и из него ничего не должно быть удалено.

Следовательно, модель SADT (IDEF0) представлена в виде серии диаграмм, имеющих сопроводительную документацию, позволяет разбить сложный объект на составляющие части и представить его в виде блоков.

Детализация каждого из блоков может быть представлена в виде набора блоков на других диаграммах.

Метод моделирования IDEF3 позволяет моделировать последовательность выполнения действий и отразить взаимосвязь между ними в рамках реализации бизнес-процесса. Основой этой модели выступает сценарий процесса, представленный в виде последовательности действий и процессов определенной системы.

Нотация IDEF3 выступает стандартом, позволяющим документировать технологические процессы предприятия и описывать бизнес-процессы нижнего уровня.

В состав этой нотации входят объекты, представленные операторами логики, показывающими альтернативы принятия управленческих решений, а также линии, отражающие временную последовательность реализуемых работ в процессе реализации бизнес-процесса.

Диаграммы потоков данных DFD представлены в виде иерархии функциональных процессов, они связаны между собой потоками данных. Целью такого представления является демонстрация, как каждый процесс может преобразовать, а также определить отношения между этими процессами.

С помощью диаграммы потоков данных DFD можно получить представление в области преобразования процесса, выявить отношения между выделенными работами.

Вопросы для самопроверки по теме 3

1. Какие потоки отображаются на информационных диаграммах и какое их основное назначение?
2. Для чего используется функциональная модель SADT (IDEF0)?
3. В каких случаях применяется метод моделирования IDEF3?
4. Что можно получить за счет диаграммы потоков данных DFD?

Тема 4. Способы хранения данных. Ограничения «файловых» систем. Определение, преимущества (по сравнению с «файловыми» системами) и недостатки баз данных (БД)

4.1 Способы хранения данных

В настоящее время для хранения данных используются следующие способы:

- сетевое устройство, которое может быть представлено в виде жесткого диска для хранения информации с нескольких компьютеров;
- внешний жесткий диск - это тип хранения, который позволяет выполнить одноразовое резервирование данных с одного персонального компьютера;
- CD, DVD диски;
- облачные хранилища, которые являются новым способом хранения информации;
- USB-носители, которые позволяют не только накапливать данные, но и их переносить между различными устройствами.

Сетевые накопители NAS, отличаются большей практической ценностью и позволяют накапливать различного рода графическую, текстовую и табличную информацию. Аббревиатура NAS, расшифровывается как Network Attached Storage, что в переводе означает сетевой накопитель.

Как правило, в компьютерной сети применяются от двух и больше накопителей, позволяющих обеспечить защиту данных в случае выхода сетевого диска из строя, жесткий диск представлен в виде запоминающего устройства или устройства позволяющего хранить большие объемы информации.

В его основе находится принцип магнитной записи. В отличие от гибкого диска, жесткий диск может быть представлен в виде алюминиевых или стеклянных пластин, покрытых ферримагнитным материалом.

Расстояние между диском и головкой составляет несколько нанометров, а отсутствие механического контакта позволяет жесткому диску обеспечить долгий срок службы. В случае отсутствия вращения дисков, головки находится у шпинделя или за пределами безопасности.

В отличии от гибкого диска, жесткий диск позволяет накапливать больше информации. Принцип работы жестких дисков заключается в следующем: рабочая поверхность жесткого диска движется относительно считывающей головки. При подаче переменного электрического тока на катушку головки влияет переменное магнитное поле.

Облачное хранилище представлено в виде виртуального хранилища для накопления различной информации и предоставления ее в использование пользователям.

Среди преимуществ облачного хранилища следует отметить:

- возможность получения доступа к данным с любого персонального компьютера подключенного к сети Интернет;
- организация совместной работы с данными;
- резервное хранилище для информационных ресурсов предприятия.

В настоящее время наиболее популярными облачными хранилищами выступают такие как: Dropbox, One Drive, Google Drive, Яндекс. Диск и облака mail.ru.

Непосредственно с облачным хранилищем связаны облачные шлюзы, которые представлены в виде технологий, позволяющих предоставить облачную среду для размещения данных пользователей. Например, с помощью программного обеспечения, в облачной среде, можно накапливать больше количество информации и получить к ней доступ с удаленных компьютеров.

4.2 Ограничения файловых систем

Файловая система используется для организации порядка и хранения информации. За счет иерархии обеспечивается быстрый поиск информации, и ее сортировка. Способ упорядочивания информации позволяет оптимизировать функционирование компьютера, электронных носителей и мобильных устройств. За счет использования определенных видов файловых систем определяются форматы и способы хранения информации. При этом любые варианты можно выполнять на основании группировки файлов.

Файловую систему можно представить в виде системы алгоритмов и стандартов, которые направлены на предоставление доступа пользователей к данным, хранящихся на персональном компьютере.

Специалисты относят файловую систему к различным категориям программного обеспечения и считают ее частью операционной системы, но в тоже время независимым компонентом.

Применение определенных вариантов файловых систем, позволяет определить размеры и другие особенности.

Среди основных параметров файловой системой следует отметить:

- максимальные размеры файлов;
- атрибуты файла, например разграничение по типам: для чтения, скрытый, архивный, исполняемый;
- дополнительные сервисные функции такие как: шифрование и разграничения доступа.

С помощью файловой системы обеспечивается взаимосвязь между информационным носителем жесткий диск, оптический накопитель, флеш-память и API программным интерфейсом, который выполняют служебные функции и обеспечивает работу с различными приложениями, в том числе доступ файлам.

Обращение к файлу является обязательным элементом работы используемого программного обеспечения, функции которого связаны с определенным именем, размером и атрибутом файла.

Одной из наиболее ранних файловых систем, является файловая система FAT, которая появилась 70-х годах XX века. Она является достаточно простой в использовании, поэтому в настоящее время применяется достаточно часто. Разновидностями файловой системы FAT являются: FAT12, FAT16, FAT32.

Эта файловая система применяется в операционных системах Windows и для нее характерно исключение ограничений по размерам файла и диска, поддержка жестких ссылок и подкреплением алгоритмов шифрования и сжатия. Аналогом файловой системы FAT, является файловая система EXFAT, которая позволяет обеспечивать работу съемных флеш-носителей. FAT и FAT32 являются наиболее распространенными вариантами для работы с Flash-накопителями.

Среди преимуществ файловой системы EXFAT, следует отметить:

- возможность сокращения количества перезаписей одного сектора, и тем самым увеличение срока службы флеш-накопителя;
- максимальные размеры файлов и кластеров составляют 16 экстра байт и 32 мегабайт;
- характеризуется улучшенным распределением незанятого места за счет работы с Бит картами.

Файловая система UDF является универсальным дисковым форматом и ее особенностью является возможность работы независимо от операционной системы, что достаточно важно для обеспечения бесперебойной работы оптического носителя. Программное обеспечение отличается широкими возможностями ведения и записывания файлов на диске CDR и CDRW.

К функциям файловой системы UDF относится:

- наименование файлов, а также организация их взаимосвязи через программный интерфейс;
- отображение логической модели на физическом уровне, для того чтобы сформировать хранилище информации;
- поддержка устойчивости файловой системы и обеспечение защиты от перебоев в сети;
- защита конфиденциальной информации при наличии в системе нескольких пользователей.

В основе работы файловой системы находятся разные алгоритмы, которые определяются типом персонального компьютера, видом операционной системы и особенностями хранения. В настоящее время для работы под операционной системой Windows, наиболее часто применяется файловая система NTFS, предложенная корпорации Windows. В файловой системе NTFS все атрибуты файлов размещены на дисках и хранятся в скрытом системном файле. Файлы небольших размеров также размещены в системной конфигурации.

Файловая система NTFS отличается от других файловых систем более сложной структурой. В этом случае файлы представлены в виде перечня особых атрибутов.

Можно сказать, что файловая система NTFS имеет возможности, которые отсутствуют в файловой системе FAT. Она позволяет увеличить гибкость, надежность и защищенность системы в сравнении с FAT.

Среди возможностей файловой системы NTFS следует отметить:

- наличие средства разграничения доступа, которое позволяет выполнять управлением доступом на уровне каталогов и файлов с предоставлением средств, блокировки и доступа группам пользователям сети;
- средства шифрования, позволяющие разграничивать доступ и повысить безопасность файлов, а также обеспечивают защиту от несанкционированного доступа в операционной системе Windows;
- программный дисковый массив WRITE, который представлен в виде логических и физических дисков, имеющих одинаковые объемы и позволяющий выполнять дублирования файлов.

Файловая система NTFS создает возможности объединения одного или нескольких разделов, размещенных на одном или нескольких физических дисках. Для этого могут использоваться системы хранения баз данных больших размеров, которые не помещаются на одном диске или для того, чтобы создать каталог с большим количеством файлов, превышающих размеры физического диска.

Кроме этого файловая система NTFS создает возможности создания разреженных файлов, в которых включены области нулевых данных. Эти файлы имеют большие объемы, но при этом не занимают большого количества на диске. Многопоточные файлы, при необходимости в одном файле, написанном на диске NTFS.

Жесткие связи создают возможность назначения для одного физического файла нескольких различных имен с размещением этих имен в различных каталогах. При удалении связи не происходит удаление самого файла, только когда связи файла будут уничтожены, тогда и удаляется сам файл. Точки переопределения создают возможности, переопределение любого файла или каталога, при этом достаточно редко используется переопределенные файлы или каталоги.

Несмотря на достаточно большое количество возможностей, файловая система NTFS имеет ограничения в области использования размеров логического пространства, максимальный объем логического диска в NTFS равен: 18446744; Тбайт. Однако, этого достаточно для работы многих современных приложений. К тому же количество файлов находящихся в одном каталоге не ограничивается, что создает преимущество перед файловой системой FAT.

С точки зрения перспективности развития функциональных возможностей, обеспечения защиты данных и надежности, файловая система NTFS намного эффективнее, чем файловая система FAT. Однако если сравнить производительность этих файловых систем, то можно сказать, что она определяется различными факторами.

Принципы работы и внутренняя структура файловой системы FAT намного проще, чем NTFS, при работе с небольшими каталогами FAT может быстрее выполнить обработку операции, чем система NTFS. Однако если содержимое каталога незначительно, то оно будет полностью помещено в несколько записей или каталогов.

Можно сказать, что NTFS при поиске несуществующих файлов и каталогов и при обращении к файлам небольшого размера, а также в случае дефрагментации лучше, чем FAT система.

Для того чтобы увеличить производительность файловой системой NTFS, необходимо увеличить размеры кластеров. Однако, это может привести к не экономному использованию дискового пространства. В случае упаковки файлов размещенных на диске небольшого размера, производительность увеличивается, однако при работе с файлами большого размера уменьшается.

Таким образом, можно сказать, что более надежной является NTFS и ее ограничения несущественны. Наиболее серьезные препятствия создает необходимость использования операционной системы Windows.

Для того чтобы обеспечить нормальную работу операционной системы, необходимо минимальное количество объемной памяти 64 МБ. В случае работы с новыми версиями файловой системы, требования к аппаратному обеспечению компьютера увеличиваются.

4.3 Определение, преимущества (по сравнению с «файловыми» системами) и недостатки баз данных (БД)

С момента развития вычислительной техники и информационных технологий, были образованы два основных направления их использования. Первое направление связано с использованием вычислительной техники для осуществления численных расчетов, которые можно производить ручным способом. Становление этого направления

позволило идентифицировать методы численного решения сложных задач, развивать классы языков программирования которые ориентированы на запись численных алгоритмов, устанавливать обратную связь с разработчиками архитектур вычислительных систем.

Второе направление непосредственно связано с использованием средств вычислительной техники в информационных системах. Как правило, объем информациями, с которым приходится повседневно работать, достаточно усложняет структуру информационного обеспечения.

Классическими информационными системами выступают системы резервирования, банковские системы, позволяющие автоматизировать деятельность предприятия.

Можно сказать, что второе направление возникло немного позже, это связано с тем, что на первоначальном этапе персональные компьютеры ограничивались, в части памяти и это не позволяло долговременно хранить информацию, поскольку запоминающие устройства имели небольшие объемы.

С точки зрение по прикладной программы, файл является именной областью внешней памяти, в которую выполняется запись и из которой выполняется считывание данных. Правила именования файлов выступают способом предоставление доступа к данным, а структура этих данных определяется системой управления данными.

Система управления данными выполняет функции по распределению внешней памяти и отображению имен файлов, в соответствии с имеющимися адресами во внешней памяти и последующем обеспечением доступа к данным.

На начальном этапе проблемы управления информацией решались индивидуально для каждой системы. Для этого осуществлялись необходимые настройки над файловыми системами, за счет использования возможностей редакторов и компиляторов. Поскольку информационные системы имеют сложную структуру данных, то дополнительные индивидуальные средства позволяют управлять.

Файловые системы и базы данных направлены на поддержку совокупности данных, которая подлежат обработке, хранению, вводу и модификации. Файловые системы и базы данных зависят от возможностей операционных систем, в качестве физического объекта управления в них выступают внешние устройства.

Развитие и создание сложных систем привело к необходимости увеличения объема в запоминающих устройствах, а магнитные накопители в настоящее время заменены облачными хранилищами.

Файловые системы позволяют управлять набором данных без учета природы и назначения данных. При этом каждое прикладное программное обеспечение предусматривает свой набор данных, что создает

возможность многократного их дублирования и отображение в различных наборах.

Базы данных являются развитием файловых систем, в них учитывается природа хранимой информации, обеспечивается непротиворечивость и ее единство. Они выполняют функции организации совместного использования информации через прикладные продукты. За счет централизованного управления в базах данных обеспечивается повышенная надежность хранения информации, защита от аппаратных сбоев и программных ошибок, а также высокая эффективность работы за счет минимизации затраченного времени на обработку информации.

В свою очередь, файловые системы позволяют обеспечивать доступ к различным структурам наборов данным, управлять наборами данных на различном уровне, распределять внешнюю память, защищать информацию в наборах данных и обеспечивать коллективный доступ к ней.

Вопросы для самопроверки по теме 4

1. Какие существуют способы хранения данных?
2. Что такое облачные хранилища и в чем их преимущества?
3. Что характерно для файловой системы FAT?
4. Какие функции позволяет реализовать файловая система UDF?
5. В чем отличия файловой систем NTFS от других систем?
6. Какая файловая система позволяет создать больше возможностей FAT или NTFS и почему?
7. В чем отличия файловой системы от базы данных?

Тема 5. Системы управления базами данных (СУБД): определение, возможности, преимущества и недостатки. Примеры СУБД, их сравнительные характеристики

5.1 Определение СУБД и ее возможности

База данных – это система именованной совокупности данных, позволяющих отразить состояние объектов и их отношений определенной предметной области.

Также можно сказать, что база данных это – совместно используемый набор логически связанных данных, позволяющих удовлетворить информационные потребности определенное группы пользователей.

Предметная область – это реально существующая система, выступающая самостоятельной единицей. Полная предметная область может быть представлена в виде группы определенных объектов, однако на практике для информационных систем наибольшую значимость имеет предметная область отдельного предприятия или корпорации.

Система управления базами данных – это система программных и языковых средств, позволяющих создавать, вводить, поддерживать в актуальном состоянии, организовывать поиск неоднородной информации.

Набор средств, формирующий функциональные возможности системы управления базами данных приведен на рисунке 5.1.

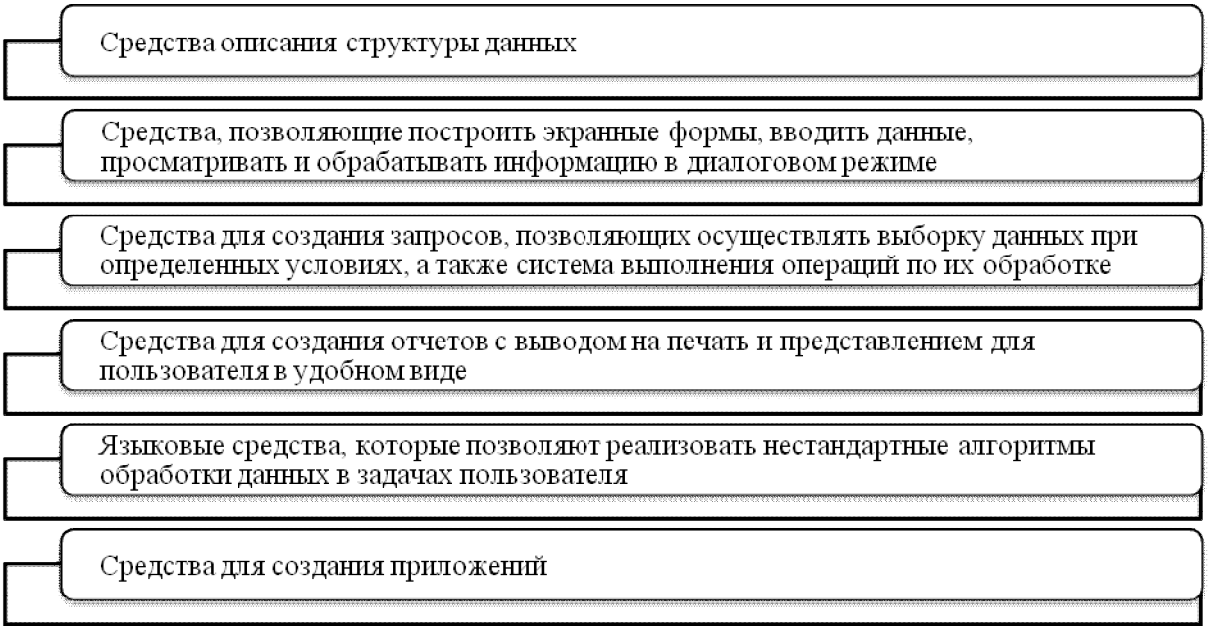

Рисунок 5.1 – Набор средств, формирующий функциональные возможности системы управления базами данных

Кроме вышеперечисленных средств, система управления базами данных может включать средства для обеспечения целостности, безопасности данных, выполнения операций по экспорту и импорту данных.

Современная система управления базами данных должна отвечать требованиям, приведенным в таблице 5.1.

Таблица 5.1 – Требования к системе управления базами данных

Требование	Описание
Масштабируемость	Обработка запросов пользователей должна выполняться с высокой скоростью данных
Доступность	Любой поступающий запрос должен всегда выполняться
Надежность	Должна иметь средства администрирования, автоматического конфигурирования, мониторинга событий
Средства защиты	Должна включать средства разграничения доступа, защиты от несанкционированного доступа

Таким образом, современная система управления базами данных должна иметь высокий уровень масштабируемости, доступности, надежности данных, а также включать средства защиты.

5.2 Преимущества и недостатки СУБД

На первоначальном этапе рассмотрим преимущества системы управления базами данных:

1. Контроль избыточности данных. Традиционные файловые системы не экономно расходует ресурсы внешней памяти за счет сохранения одних и тех же данных в нескольких файлах. В случае использования базы данных исключается избыточность за счет интеграции информации из файлов. Однако полностью избыточность информации в базе данных исключить нельзя, а можно лишь выполнять ее контроль. В одних случаях ключевые элементы дублируют для моделирования связей, а в других необходимо дублировать данные для того чтобы обеспечить повышенную производительность системы.

2. Непротиворечивость данных. Исключение избыточности, ее контроль позволяют сократить риски возникновения противоречивости.

3. Совместное использование данных. Файлы обычно принадлежат отдельным отделам или лицам, которые используют их для осуществления своей работы, а база данных может одновременно использоваться всеми подразделениями или зарегистрированными в ней пользователями.

4. Поддержка целостности данных. Целостность базы данных связана с непротиворечивостью и корректностью, хранимой в ней информации. Для обеспечения целостности применяют ограничения, то есть правила, позволяющие поддерживать непротиворечивость и непрерывность в обработке запросов пользователей.

5. Повышенная безопасность. Система защиты системы управления базами данных основана на парольной защите, которая в дальнейшем используются для идентификации пользователей, зарегистрированных в базе данных.

6. Улучшения показателей производительности. В системе управления базами данных предусмотрены различные стандартные функции, которые программист может использовать для разработки приложений.

7. Улучшенное управление параллельностью. В некоторых файловых системах при организации одновременного доступа к одному и тому же файлу возникают конфликты при обработке запросов. Для того, чтобы исключить эту ситуацию, в системе управления базами данных предусмотрен параллельный доступ.

8. Развитые системы резервного копирования и восстановления. Ответственность за обеспечение защиты данных от сбоев аппаратного и программного обеспечения в файловой системе возложена на пользователя. В системе управления базами данных для решения этого вопроса предусмотрены средства, позволяющие сокращать объем потерь информации от возникновения различных сбоев.

Недостатки системы управления базами данных:

1. Сложность. Для того чтобы обеспечить полную функциональность, современная базами база данных сопровождается различным программным обеспечением. Для того, чтобы использовать все преимущества системы управления базами данных, проектировщики и разработчики, а также конечные пользователи должны понимать ее функциональные возможности. Непонимание этих принципов приводит к неудачным результатам проектирования и серьезным последствиям.

2. Размеры. Чем более функциональной является система управления базами данных, тем сложнее управлять программным продуктом, который занимает достаточно много места на диске, и это приводит к нагрузке на оперативную память и снижению эффективности работы пользователя.

3. Стоимость. В зависимости от функциональности и возможностей, стоимость системы управления базами данных увеличивается.

4. Централизации ресурсов приводит к повышению уязвимостей и зависимости от работоспособности системы управления базами данных, поскольку работа всех пользователей и приложений определяется ее готовностью к работе.

5.3 Примеры СУБД и их сравнительные характеристики

Рассмотрим наиболее популярные системы управления базами данных, которые представлены в виде коммерческих продуктов и не коммерческих продуктов.

1. Система управления базами данных Oracle 12c – это система управления базами данных, которая используется в настоящее время в облачных средах и может быть размещена, как на одном сервере, так и на нескольких.

Среди достоинств системы управления базами данных Oracle 12c следует отметить то, что в этом программном продукте реализованы возможности управления базами данных не только в физической среде, но и в облачной среде.

Недостатком системы управления базами данных Oracle 12c является высокая степень потребления вычислительных ресурсов, поэтому существует необходимость использования достаточно мощных производительных серверов.

2. Система управления базами данных MySQL – это система управления базами данных для работы с веб-приложениями. Она является открытым программным обеспечением, ее версии постоянно обновляются и улучшаются с усилением безопасности данных. В настоящее время существуют коммерческие версии этой системы, имеющие расширенный функционал, позволяющий реализовывать крупные веб-проекты.

Гибкость системы управления базами данных MySQL обеспечивается поддержкой работы с различными форматами данных, организации полнотекстового поиска, транзакций на уровне отдельных записей.

Достоинства: открытое программное обеспечение, документальная поддержка, многофункциональность, поддержка набора пользовательских интерфейсов, возможна интеграция с другими СУБД.

Недостатки: не поддерживает механизмы создания инкрементных резервных копий, не поддерживаются форматы XML и OLAP.

3 Система управления базами данных MS SQL Server – это одна из наиболее известных систем управления базами данных, позволяет обрабатывать данные не только с локальных серверов, но из облачных серверов. При этом существует возможность комбинирования различных серверов одновременно.

После выпуска последней версии MS SQL Server 2016 система управления базами данных, она адаптирована под работу не только под операционную систему Windows, но и системой Linux.

Особенностью последней версии системы управления базами данных MS SQL Server возможность реализации временной поддержки данных (temporal data support) для отслеживания изменений с течением времени, а также механизм динамической маскировки данных (dynamic data masking),

позволяющей авторизированным пользователям получить доступ к конфиденциальной информации.

Достоинствами системы управления базами данных MS SQL Server являются: простота в эксплуатации, стабильность и скорость обработки данных, возможность регулировки и отслеживания уровней производительности, получение доступа с мобильных приложений и взаимодействие с другими программными продуктами корпорации Microsoft.

Недостатками СУБД MS SQL Server являются высокая цена, направленность на корпоративных клиентов.

Вопросы для самопроверки по теме 5

1. Что представляет из себя база данных?
2. Что такое система управления базами данных?
3. Какие средства позволяют сформировать функциональные возможности системы управления базами данных?
4. Перечислите основные требования к системе управления базами данных?
5. В чем заключаются преимущества системы управления базами данных?
6. Какие система управления базами данных имеет недостатки?
7. Какая система управления базами данных ориентирована на обработку данных на базе крупных предприятий?

Тема 6. Концептуальное моделирование

В процессе концептуального моделирования выполняется разработка концептуальной модели, которая является абстрактной моделью, позволяющей определить структуру моделируемой системы, ее элементы, свойства и причинно-следственные связи.

В результате разработки концептуальной модели формируются элементы, процессы и характеристики, позволяющие охарактеризовать определенную предметную область.

6.1 Модели данных

Модель данных можно представить в виде сочетания нескольких компонентов:

- структурной части, то есть набора правил, на основании которых ведется разработка базы данных;
- управляющей части, в которой определены типы операции с данными.

В состав модели данных также входят:

- операции извлечения и обновления данных;
- операции изменения структуры базы данных;
- набор поддержки целостности данных, который позволяет определить корректность используемой в базе данных информации.

Целью построения модели данных является представление в ней данных в понятном виде.

В отношении архитектуры ANSI-SPARC можно определить три связанные модели данных:

- внешняя модель данных, которая позволяет отобразить представление для каждого существующего в организации типа пользователей;
- концептуальная модель данных, которая отображает логическое представление данных вне зависимости от типа системы управления базами данных;
- внутренняя модель данных, которая отображает концептуальную схему на понятном для системы управления базами данных языке.

Также выделяют объектную модель данных (позволяет описать данные на внешнем и концептуальном уровне) и физическую модель данных.

Система управления базами данных может быть представлена в виде иерархической, сетевой и реляционной моделях данных.

Иерархическая модель представлена в виде совокупности элементов, которые связаны между собой по определенным правилам. Графическое представление иерархической модели представлено в виде развернутого дерева. К основным понятиям иерархической модели относятся связь, уровень элемента.

Каждый узел на более низком уровне связывается с другим узлом, который размещен на верхнем уровне. Таким образом, образуется иерархическое дерево, в котором выделяется вершина или корень дерева.

Аналогичной иерархической модели является сетевая модель, отличием которой является только то, что элемент может быть связан с другим элементом.

Наиболее распространенной является реляционная модель, представленная таблицами реляционной базы данных, включающими записи, столбцы, а также поля. Рассмотрим ее далее более детально.

6.2 Реляционная модель данных

Основными элементами реляционной модели данных выступают:

- отношения – это двухмерные таблицы, в которых размещены различные данные;
- сущность – это объект, описание о которого размещено в базе данных;
- атрибуты – это свойства сущности, соответствующие столбцам таблицы;
- домен – множество возможных значений атрибута отношения.
- схема отношения или заголовок отношения – это список имен атрибутов.

Каждая сущность реляционной модели данных имеет первичный ключ или ключ отношения, позволяющий ее идентифицировать. Первичный ключ является обязательным и его значения не изменяются. Каждая сущность может иметь и другие ключи, получившие название альтернативных ключей.

К достоинствам реляционной модели данных относятся:

- доступность и простота для пользователя;
- строгие правила при осуществлении проектирования базы данных, основанные на математическом аппарате;
- полная независимость данных, которая позволяет минимизировать объемы данных в реляционной базе данных;
- организация запросов при написании прикладного программного обеспечения исключает необходимость описания организации база данных во внешней памяти.

Недостатками реляционной модели данных являются:

- не всегда предметную область можно представить в виде таблиц;
- логическое проектирование включает достаточно много таблиц, что создает громоздкость структуры данных и сложности в ее понимании;
- база данных предусматривает выделение больших объемов памяти;
- относительно низкая скорость получения доступа к данным.

Реляционная модель данных представлена в виде структурной, манипуляционной и целостной частей.

Структурная часть реляционной модели позволяет определить, что в качестве единственной структуры данных выступает нормализованные парные отношения. При этом отношения достаточно удобно представлять в форме таблицы, в которой каждая строка выступает кортежем, а каждый столбец в виде атрибута соответствующему определенному домену.Этот неформализованный подход к понятию отношений образует форму представления, в которой реляционная база данных представлена в виде конечного набора таблиц.

Манипуляционная часть реляционной модели позволяет использовать два фундаментальных механизма манипулирования данными, такие как реляционную алгебру и реляционное исчисление.

К основным задачам манипуляционной части реляционной модели относится выполнение требований к использованию языка реляционных баз данных, который имеет не меньшую мощность и выразительность, чем реляционная алгебра или реляционное исчисление.

Целостная часть реляционной модели определяет требования к выполнению целостности сущностей и ссылок. Первое требование основано том на том, что любое отношение должно включать первичный ключ, а второе – для каждого значения внешнего ключа может быть указана ссылка, то есть должен быть найден кортеж, имеющий такие же значение что и у первичного ключа или внешнего ключа.

6.3 Структурированный язык запросов» SQL

Структурированный язык запросов SQL позволяет выполнять различные операции с данными представленными в виде логически взаимосвязанных таблиц.

SQL является информационно-логическим языком, предназначенным для описания, изменения и извлечения данных, хранимых в реляционных базах данных.

SQL был основным способом работы пользователя с базой данных и позволяет выполнять набор операций, приведенный на рисунке 6.1.

Рисунок 6.1 – Возможности SQL

При всех своих изменениях, SQL остается единственным механизмом связи между прикладным программным обеспечением и базой данных. В то же время, современные СУБД, а, также, информационные системы, использующие СУБД, предоставляют пользователю развитые средства визуального построения запросов.

Каждое предложение SQL – это запрос или обращение к базе данных, которое приводит к изменению в базе данных.

В соответствии с тем, какие изменения происходят в базе данных, различают следующие типы запросов, приведенные на рисунке 6.2.

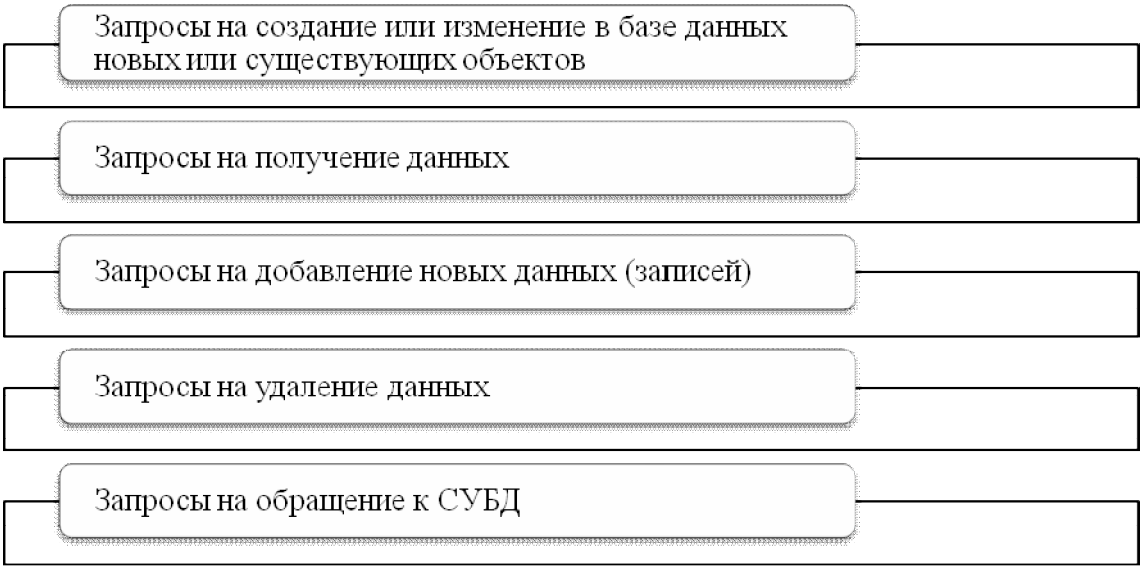

Рисунок 6.2 – Типы запросов для организации работы СУБД

Основным объектом хранения реляционной базы данных является таблица, поэтому все SQL-запросы – это операции над таблицами. В соответствии с этим, запросы делятся на:

- запросы, оперирующие самими таблицами (создание и изменение таблиц);
- запросы, оперирующие с отдельными записями (или строками таблиц) или наборами записей.

Каждая таблица описывается в виде перечисления своих полей (столбцов таблицы) с указанием:

- типа хранимых в каждом поле значений;
- связей между таблицами (задание первичных и вторичных ключей);
- информации, необходимой для построения индексов.

Запросы первого типа, в свою очередь, делятся на запросы, предназначенные для создания в базе данных новых таблиц, и на запросы, предназначенные для изменения уже существующих таблиц.

Запросы второго типа оперируют со строками, и их можно разделить на запросы следующего вида:

- вставка новой строки;
- изменение значений полей строки или набора строк;
- удаление строки или набора строк.

Самый главный вид запроса — это запрос, возвращающий (пользователю) некоторый набор строк, с которым можно осуществить одну из трех операций:

- просмотреть полученный набор;
- изменить все записи набора;
- удалить все записи набора.

Таким образом, использование SQL сводится, по сути, к формированию всевозможных выборок строк и совершению операций над всеми записями, входящими в набор.

Таким образом, использование SQL сводится, по сути, к формированию всевозможных выборок строк и совершению операций над всеми записями, входящими в набор.

Различают следующие основные операторы SQL: операторы определения данных DDL, операторы манипулирования данными DML, а также и операторы управления транзакциями TCL.

Операторы определения объектов базы данных позволяют создавать, удалять, изменять базу данных, таблицы в базе данных, а также выполнять операции с доменами баз данных и представлениями.

Основные операторы определения объектов базы данных DDL и их назначение приведены в таблице 6.1.

Таблица 6.1 - Операторы определения объектов базы данных DDL

Название	Значение
CREATE SCHEMA	Создание схемы базы данных
DROP SHEMA	Удалить схему базы данных
CREATE TABLE	Создать таблицу
ALTER TABLE	Изменить таблицу
DROP TABLE	Удалить таблицу
CREATE DOMAIN	Создать домен
ALTER DOMAIN	Изменить домен
DROP DOMAIN	Удалить домен
CREATE COLLATION	Создать последовательность
DROP COLLATION	Удалить последовательность
CREATE VIEW	Создать представление
DROP VIEW	Удалить представление

Операторы манипулирования данными DML позволяют отобрать, добавить и изменить строки в таблице базы данных, а также зафиксировать и откатить внесенные изменения, как показано в таблице 6.2.

Таблица 6.2 – Операторы манипулирования данными DML

Название	Значение
SELECT	Отобрать строки из таблиц
INSERT	Добавить строки в таблицу
UPDATE	Изменить строки в таблице
DELETE	Удалить строки в таблице
COMMIT	Зафиксировать внесенные изменения
ROLLBACK	Откатить внесенные изменения

Операторы управления транзакциями позволяют создавать и удалять ограничения, предоставлять и отменять привилегии пользователю или приложению на манипулирование объектами и их назначение приведено в таблице 3.3.

Таблица 6.3 – Операторы управления транзакциями TCL

Название	Значение
CREATE ASSERTION	Создать ограничение
DROP ASSERTION	Удалить ограничение
GRANT	Предоставить привилегии пользователю или приложению на манипулирование объектами
REVOKE	Отменить привилегии пользователя или приложения

Так как существует общепринятый стандарт языка SQL, многие разработчики СУБД стараются придерживаться его. Использование базой данных языка SQL является одним из факторов, определяющих ее

успешность и распространенность. Существуют различия в синтаксисе между конкретными СУБД, но в большинстве случаев запросы из одной СУБД могут быть перенесены на другую с минимальными изменениями.

Наличие стандартов и набора тестов для определения совместимости конкретной реализации SQL к общепринятому стандарту заметно способствует унификации языка.

К основным недостаткам языка SQL относятся неопределенные значения, возможность дублирования, отсутствие поддержки свойства «=» и высокая избыточность. Современный SQL считается сложным для освоения. Сейчас с ним работают в основном программисты, хотя изначально он задумывался как язык, с которым сможет работать конечный пользователь.

6.4 Использование механизма «представлений» (View) в СУБД

Механизм представлений выступает одним из средств языка SQL, он позволяет скрыть структуру базы данных от пользователей за счет определения представлений базы данных, которые относятся к некоторым хранимым в базе данных запросам, имеющим именованные столбцы, а для пользователя ничем не отличаются от базовой таблицы базы данных и учитывают технические ограничения.

Любая реализация должна быть гарантированной, что состояние представленной таблицы точно соответствуют состоянию базовых таблиц, на которых определено представление. Обычно вычисления представляемой таблицы выполняются один раз с использованием представлений.

В стандарте SQL-оператор для определения представлений имеет следующий синтаксис:

View Definition: = Create View Table name View column list as SQL specification with Check Option View column list column name column name. Определяемое представление таблицы V является изменяемой, то есть по отношению к ней можно использовать команды DELETE и UPDATE в том случае, если выполняются следующие условия:

- в списке выборки не указывается команда DISTINCT;
- арифметическое выражение в списке выборки представлено в виде одной спецификации столбца и спецификация этого столбца может использоваться не более одного раза. В команде FROM указывается только одна таблица, которая выступает базовой таблицей или изменяемой представленной таблицей;
- в условиях выборки команда WHERE не используется под запросы;

- в табличном выражении отсутствуют команды GROUP BY и HAVING;

- если в выборке спецификации запросов указано хотя бы одно арифметическое выражение, включающее не одну спецификацию столбца, или если одно имя столбцов включено в список выборки более одного раза, определение представления включает список имен столбцов представленной таблицы.

Более явно можно выполнить именования столбцов в представленной таблице, если эти имена наследуются от столбцов таблиц, указанных в команде FROM. Требования WITH CHECK OPTION в определении представления указывает на определение изменяемой представленной таблицы, которое представлено в виде спецификации запроса, включающего команду WHERE. При наличии этого требования не допускаются изменения таблицы, которые приводят к появлению в ней базовых строк, невидимых представленной таблице. Если WITH CHECK OPTION в определении представления отсутствует, такой контроль не осуществляется.

В соответствии со спецификацией языка SQL контроль прав доступа пользователя осуществляется на основе механизма привилегий. Фактически этот механизм состоит в том, что и для выполнения действия с таблицами пользователь должен иметь определенную привилегию. Пользователь, создавший таблицу, автоматически выступает и ее владельцем. В число этих привилегий входят привилегии на передачу всех данных другому пользователю, включая привилегии на передачу привилегий.

Запрос выступает командой, которая указывается в программе или вводится с командой строки с целью осуществления поиска в базе данных и сообщение ей, что полученная информация вынесена в память таблицы. Как правило, запросы выступают частью языка DML. Однако запрос не позволяет полностью изменить информацию в таблицах, а просто создает возможности для пользователя просмотра запросов в виде самостоятельной категории среди команд DML.

Самой простой формой является команда SELECT, которая позволяет инструктировать базу данных с целью извлечения из нее данных. Команда SELECT начинается с ключевого слова SELECT, а затем сопровождается пробелом, после этого следует список имен столбцов, которые разделяются между собой запятыми. После SELECT указывается ключевое слово FROM, которое сопровождается пробелом с указанием имени таблицы, запрос по которой осуществляется. В заключение ставится точка с запятой, которая указывает на окончание запроса и готовность выполнения команды.

Оператор SQL является строкой на языке и SQL, который передается на обработку системе управления базами данных. Содержание строки

включает зарезервированные слова SQL, например, команды SELECT и UPDATE являются зарезервированные словами и не могут использоваться как названия таблиц.

Оператор должен быть всегда эквивалентен SQL, например, SELECT «название отдела» FROM «отдел». Только этот оператор может быть выполнен, что приведет к сообщению, запрашиваемого дополнительную информацию.

Вопросы для самопроверки по теме 6

1. Из каких основных компонентов состоит модель данных?
2. Какие можно выделить модели в отношении архитектуры ANSI-SPARC?
3. В виде каких моделей может быть представлена СУБД?
4. Перечислите основные компоненты реляционной модели данных
5. В чем заключаются достоинства реляционной модели данных?
6. Для чего предназначен структурированный язык запросов SQL?
7. Перечислите операторы определения объектов базы данных DDL
8. Какие операторы относятся к операторам манипулирования данными DML?
9. Для чего применяются операторы управления транзакциями TCL?
10. Для чего применяется механизм представлений?

Тема 7. Этапы проектирования БД: концептуальное, логическое и физическое проектирование. Соответствие этапов моделирования данных и элементов архитектуры ANSI-SPARC

7.1 Этапы проектирования БД

Этапы проектирования баз данных включают в себя три основных этапа: концептуальное, логическое и физическое проектирование.

Концептуальное проектирование баз данных представляет собой процесс создания модели, которая используется на предприятии и не зависит от физических аспектов ее представления. Первый этап процесса проектирования базы данных получил название концептуального проектирования базы данных, и он состоит в том, что создается концептуальная модель данных для анализа предметной области. Эта модель данных создается с использованием информации, которая отражена в спецификациях требований пользователей. Концептуальное проектирование базы данных не зависит от детализации ее реализации, выбранного типа целевой системы управления базами данных, прикладного программного обеспечения, языков программирования, вычислительной платформы, а также любых других особенностей физической реализации.

При разработке концептуальной модели система полностью подвергается тестированию и проверяется на выполнение требований пользователя. Созданная таким образом концептуальная модель выступает источником информации и в дальнейшем используется на этапе логического проектирования базы данных.

Логическое проектирование базы данных предусматривает создание модели, которая используется на предприятии на основе выбранной модели организации данных, но в ней не учитывается тип целевой системы управления базами данных, а также другие физические аспекты реализации.

Второй этап проектирования базы данных получил название логического проектирования. Его цель заключается в создание логической модели для исследуемой части предприятия. Концептуальная модель данных, созданная на предыдущем этапе, преобразовывается и уточняется в логическую модель данных. Логическая модель данных позволяет учитывать особенности выбранной модели организации данных в целевой системе управления базами данных, например, в реляционной модели.

Если концептуальная модель не зависит от физических аспектов, то в логической модели данных на основе выбранной модели создается целевая система управления базами данных. Другими словами, на этом этапе должно быть известно, какая система управления базами данных будет являться целевой, а какая будет игнорироваться. В качестве целевой системы базы данных может быть выбрана как объектно-ориентированная система, так и иерархическая, сетевая или реляционная система управления базами данных.

В процессе разработки логической модели данных разработчиками выполняется постоянное тестирование и проверка выполненных действий на соответствие требований пользователей. Для проверки правильности логической модели специалистами применяется метод нормализации, который позволяют получить гарантию, что отношения, которые введены в модели данных, не имеют избыточность данных, кроме этого, логическая модель данных поддерживает все необходимые для пользователя транзакции.

Логическая модель данных выступает источником информации и на этапе физического проектирования создает возможности разработчику посредством поиска достигать поставленные цели, что достаточно важно для эффективного проектирования.

Логическая модель данных имеет достаточно важную роль на этапе сопровождения и эксплуатации информационной системы. Поэтому необходимо достаточно много внимания уделять организационному сопровождению, которое должно поддерживаться в актуальном состоянии.

7.2 Трехуровневая архитектура ANSI-SPARC: схема, назначение, уровни представления данных, примеры

Трехуровневая архитектура ANSI-SPARC позволяет описать логическую организацию системы и представить ее с точки зрения данных пользователя.

В состав архитектуры входит три основных уровня: внешний, концептуальный и внутренний, каждый из которых имеет различные уровни представления данных. Взаимосвязь между уровнями архитектуры ANSI-SPARC приведена на рисунке 7.1.

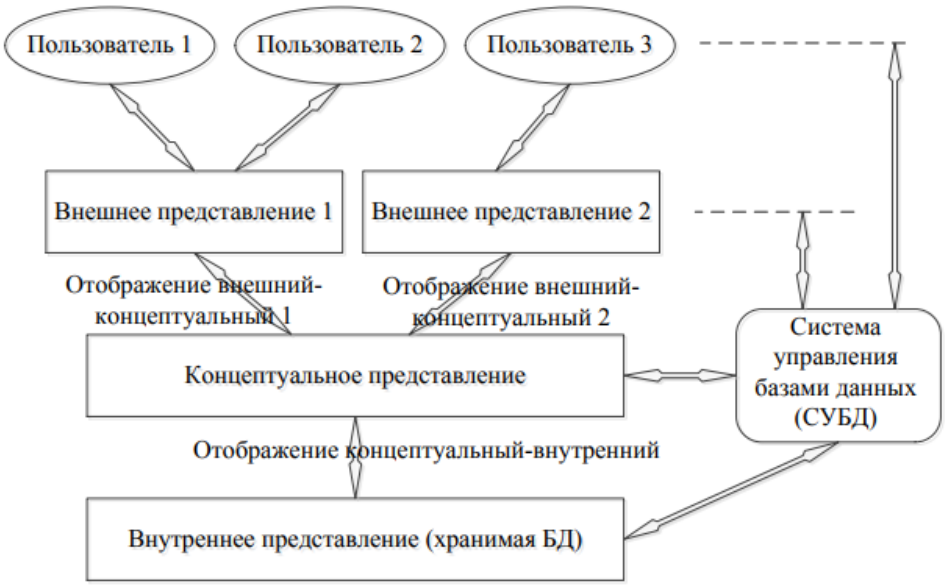

Рисунок 7.1 – Взаимосвязь между уровнями архитектуры ANSI-SPARC

Внутренний уровень используется для физического хранения данных, а внешний – направлен на пользователя и позволяет представить способы представления данных для отдельных пользователей. Концептуальный уровень является промежуточным уровнем и на нем представлены данные в абстрактном виде по отношению к другим уровням системы.

На внешнем уровне пользователи распределены по группам с выделением прикладных программистов, администраторов и конечных пользователей. Каждый из этих пользователей используют свой язык общения с системой управления базами данных.

Концептуальный уровень включает одно представление. Концептуальное представление позволяет получить информацию по всей базе данных в абстрактной форме, которая включает множество экземпляров различных типов концептуальных записей.

При этом концептуальная запись не обязательно должна соответствовать внешней записи и хранимой записи.

Концептуальное представление содержимого базы данных включает дополнительные средства, такие как правила безопасности, обеспечения целостности, средства контроля защиты данных.

Администратором данных на основании характеристик предметной области формируются база, которая сохраняется в системе и в дальнейшем выступает основой концептуальной схемы.

Внутренний уровень, как и концептуальный уровень, включает только одно представление, которое позволяет описать все данные, связанные с хранением данных в базе. В его состав входят экземпляры для описания каждой внутренней записи.

Внутреннее представление можно описать с использованием внутренней схемы, определяющей различные типы хранимых записей, способы представления этих хранимых полей, физическую последовательность, а также различные индексы, отображающие служебную структуру и ускоряющие поиск данных.

В архитектуре ANSI-SPARC существует два уровня отображения данных:

- отображение концептуального уровня на внутренний уровень, что позволяет установить соответствие между хранимой базой данных и концептуальным представлением. В случае изменения структуры данных изменяется отображение концептуально внутреннего уровня.

- отображение внешнего уровня на концептуальный уровень, что позволяет установить связи между концептуальными и внешними представлениями.

7.3 Соответствие этапов моделирования данных и элементов архитектуры ANSI-SPARC

В настоящее время для проектирования баз данных применяется трехуровневая архитектура, которая отличается от архитектуры ANSI-SPARC. Проектирование базы данных включает три основных уровня: концептуальный, логический и физический.

В соответствии с уровнями современного моделирования и элементами архитектуры ANSI-SPARC выделяют локальные концептуальные модели пользователей, которые объединены в единой концептуальной модели базы данных. В современной трехуровневой архитектуре слияние моделей пользователей в одну модель базы данных осуществляется на стадии логического проектирования, на которой создаются локальные логические модели базы данных отдельных пользователей, а затем они образуют глобальную логическую модель базы данных.

Модель концептуального уровня относится к инфологической модели предметной области, в ней представлено описание предметной области без ориентации на используемые в программе технические и программные средства. Представление базы данных на концептуальном уровне имеет свойство уникальности.

Инфологическая модель относится к человеко-ориентированной модели, полностью независимой от параметров среды и способов хранения информации. В качестве среды может использоваться память персонального компьютера или человека, поэтому инфологическая модель

не изменяется до тех пор, пока не будут происходить изменения в окружающей среде. Инфологическая модель представлена в виде интегрированных концептуальных требований всех пользователей к определенной предметной области.

На основании инфологической модели затем формируется даталогическая модель данных, которая позволяет описать язык определенной системы управления базами данных, а на основании даталогической модели формируется физическая модель данных, которая используется для описания хранимых данных в базе данных. Даталогическая модель выступает моделью логического уровня, и в ней представлены логические связи между элементами базы данных. Для разработки этой модели используется язык описания данных (ЯОД) в конкретной системе управления базами данных.

Этап разработки даталогической модели получил название даталогического проектирования. Описание логической структуры базы данных на языке систем управления базами данных получил название схемы.

При разработке логической структуры базы данных выполняется преобразование исходной инфологической модели в модель данных, которая поддерживается определенной системой управления базами данных, а затем проверяется адекватность полученной модели на отображение предметной области. Для любой предметной области можно использовать различные проектные решения и затем ее отобразить в даталогической модели.

Методика проектирования должна использоваться для выбора наиболее подходящих проектных решений. В даталогической модели отображаются логические связи между информационными данными концептуальной модели. При переходе от инфологической модели к даталогической используется вся информация, позволяющая получить представление о предметной области, а также выполнить проектирование базы данных. Это не указывает, что все сущности, которые зафиксированы в инфологической модели явным образом отображаются в даталогической модели. Прежде чем выполнить разработку даталогической модели, необходимо определить информацию для ее создания.

На этапе даталогического проектирования реализуется два направления:

- фактографическое направление, ориентированное на представление структурированной информации в виде реляционной, иерархической сетевой моделей данных;
- документальное направление, позволяющие выявить слабоструктурированные данные и представить их в виде документальных моделей или семантических сетей.

В соответствии с этими направлениями в проектировании баз данных выделяют документальные и фактографические. Различным пользователям в информационной модели отвечают различные подмножества, получившие название внешних моделей пользователей. Следовательно, внешняя модель представлена в виде концептуальных требований определенного пользователя и отражается в логической модели, определяющей эти представления. Спроектированная внешняя модель позволяет отобразить информационную модель предметной области, которая впоследствии будет использоваться для разработки автоматизированной системы управления.

На фазе физического моделирования разработчикам создается модель, оптимизированная под конкретное приложение и системы управления базами данных, с ее привязкой к даталогической модели и среде хранения. Эта модель впоследствии применяется на практике, и с ее помощью определяются запоминающие устройства, способы физической реализации. Физическую модель предметной области также называют внутренней моделью или схемой данных.

Даталогическая и физическая модели относятся к компьютероориентированной модели, с их помощью система управления базами данных выполняет управление программным обеспечением, предоставляет доступ пользователей к данным без учета физического расположения этих данных. Необходимые данные запоминаются на внешних устройствах. С помощью физической модели данных определяются способы размещения информации на носителях, а также возможность организации к ним доступа.

Между логическим и физическим проектированием устанавливается обратная связь, поскольку решение, принимаемое на этапе физического проектирования, направлено на повышение производительности системы, позволяет влиять на структуру логической модели.

Ключевым моментом проектирования баз данных является разработка механизмов обеспечения защиты базы данных в соответствии с требованиями пользователей.

Таким образом, на концептуальном уровне проектирования используются представления баз данных, она позволяет определить требования к безопасности в данных.

На уровне физического проектирования баз данных реализовываются механизмы обеспечения защиты, разработанные на предыдущих этапах проектирования, с последующей регистрацией пользователей баз данных, обеспечением прав доступа к данным, реализацией механизмов представлений.

Современная трехуровневая модель проектирования баз данных включает концептуальный логический и физический уровни, так же, как и

трехуровневая архитектура ANSI-SPARC, она обеспечивает независимость данных, а также дальнейшую возможность развития базы данных без нарушения работы существующего программного обеспечения.

Вопросы для самопроверки по теме 7

1. Какие этапы используются для проектирования баз данных?
2. Какие работы выполняются при разработке концептуальной модели?
3. На каком этапе наиболее важной является логическая модель?
4. Для чего предназначена трехуровневая архитектура ANSI-SPARC?
5. Модель, какого уровня относится к инфологической модели?
6. Какая модель формируется на основании инфологической модели?
7. Какие направления реализуются на этапе даталогического моделирования?

Тема 8. Концептуальное проектирование БД

9.1 Основные понятия концептуального проектирование БД и его этапы

Концептуальное проектирование баз данных выступает первой фазой проектирования баз данных и основано на анализе предметной области с последующей разработкой концептуальной модели данных.

Проектирование сложных баз данных с большим количеством атрибутов основано на использовании нисходящих подходов. Эти подходы предусматривают разработку модели данных, включающей несколько высокоуровневых сущностей и связей, а затем работа продолжается с уточнением низкоуровневых сущностей, связей и относящихся к ним атрибутов.

Нисходящий подход позволяет продемонстрировать концептуальной модели технологии высокоуровневого моделирования данных, предложенные членом. Модель «сущность-связь» является семантической моделью и связана со смысловым содержанием данных, независимо от их представления на персональном компьютере.

При разработке общей концептуальной модели данных выделяются следующие этапы:

- определение локальных представлений, которые соответствуют независимым данным. Такие представления представляются в виде подзадач;
- формулируются сущности, с помощью которых выполняется описание локальной предметной области проектируемой базы данных, описываются атрибуты, входящие в структуру каждой сущности;
- выделяются ключевые атрибуты;
- составляется спецификация между сущностями;
- удаляются избыточные связи;
- анализируются и добавляются не ключевые атрибуты;
- объединяются локальные представления.

Созданная концептуальная модель данных предприятия относится к источнику информации для фазы логического проектирования базы данных.

Первая фаза проектирования баз данных основана на создании анализируемой предметной области и представлении ее в виде концептуальной модели данных. При построении используется специальный порядок.

На первоначальном этапе разрабатывается подобная модель пользовательских представлений, которые затем преобразуются в

концептуальную модель данных. Концептуальное проектирование предусматривает разработку концептуальной схема базы данных.

Модель «сущность-связь» выступает конечным этапом разработки концептуальной модели данных.

9.2. Модель «сущность-связь» (ER-модель). ER-диаграмма. Сущности, атрибуты, ключи, связи и типы связей между сущностями. Представление сущностей, связей и атрибутов на диаграммах

Сущность выступает некоторым объектом, который включает в свой состав экземпляры, которые отличаются между собой атрибутами, имеющими однозначную идентификацию.

Тип сущности является объектом или концепцией, которая характеризуется независимым существованием. Типы сущностей могут быть как физическими, так и абстрактными. Выделяют сильные независимые сущности и слабые сущности, зависящие от других типов.

Атрибут является свойством сущности, а домен атрибута – областью его возможных значений. Выделяют простые и составные атрибуты, а также производные атрибуты и первичные ключи.

Домен можно представить в виде абстрактного атрибута, на основе которого создаются доменные прародители. Каждый атрибут может быть определен только в одном домене, а во всех доменах может быть определено несколько атрибутов.

Первичный ключ отношения выступает подмножеством атрибутов, для которого характерны следующие свойства:

- уникальность, то есть в отношении не могут использоваться кортежи, имеющие одинаковые значения;
- не избыточность, поскольку никакое подмножество не имеет свойства уникальности.

Кроме этого, каждая сущность может включать альтернативный или потенциальный ключ, который не может быть первичным. Некоторые сущности имеют естественные ключи. Например, естественным идентификатором является атрибут, значение которого имеет отношение к определенной области.

Связь является взаимодействием между сущностями, для нее характерна мощность или степень связи, которая показывает, сколько сущностей входит в эту связь. Связь между двумя сущностями получила название бинарной связи.

Связь – это ассоциация между сущностями, в состав которой входит одна сущность каждого участвующего в связи типа сущности.

Выделяют различные связи между сущностями:

- рекурсивные связи – это связи, когда одна сущность принимает участие в нескольких других ролях;

- унитарные связи, которые включают связи для указания ролей.

Среди важных характеристик связи выступает тип связи или кратность. В ER-модели могут применяться связи типа «один-к-одному», «один-ко-многим». Степень участия позволяет определить, участвуют ли в связи экземпляры. Кроме того, она может быть обязательной и необязательной.

Концептуальная ER-модель – это семантическая модель и она выступает стандартным способом, позволяющим выделить сущности и отношения между ними. С ее помощью выполняется детализация хранилищ данных.

В состав ER-диаграммы входит информация о сущностях системы, способах их взаимодействия, а также информация по идентификации объектов, которые важны для предметной области, их свойствах, атрибутах и установленных отношениях между объектами. Во многих случаях информационная модель является сложной и включает в свой состав несколько объектов.

Пример построения ER-диаграммы приведен на рисунке 9.1.

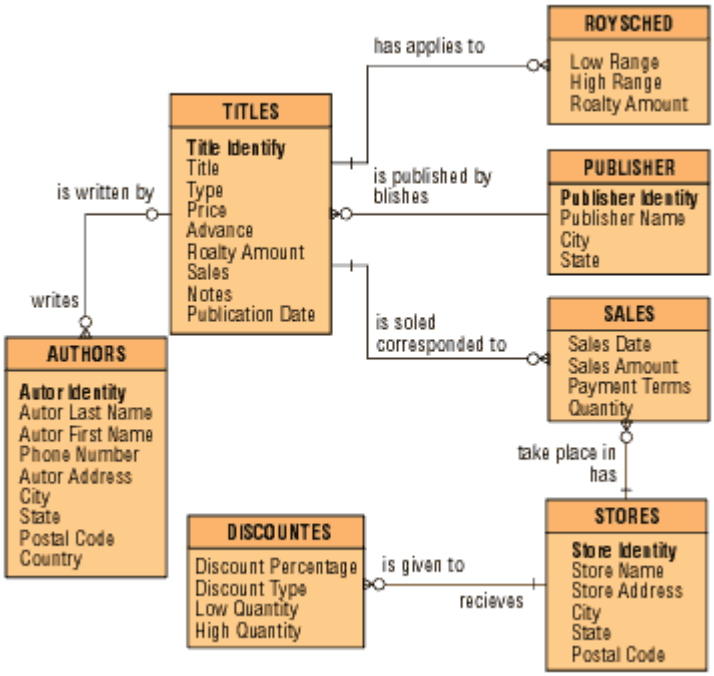

Рисунок 9.1 – Пример построения ER-диаграммы

В приведенном примере (рисунок 9.1) для отражения сущности используется прямоугольник, вверху которого отображается имя. Также в прямоугольник включены атрибуты, специальным шрифтом выделен первичный ключ. Отношения на ER-диаграмме читаются вдоль линии или слева направо или справа налево.

Вопросы для самопроверки по теме 9

1. Для чего проводится концептуальное проектирование баз данных?
2. Что такое сущность и какие выделяют ее типы?
3. Какое значение для сущности имеет атрибут?
4. Сущность обязательно должна иметь ключевое поле?
5. Какие существуют связи между сущностями?
6. Что представляет из себя ER-модель?

Тема 10. Логическое проектирование БД.

10.1 Сущность логического проектирования БД и этапы логического проектирования

Логическое проектирование

Логическое проектирование предусматривает создание схемы базы данных с использованием определенной модели, например, реляционной. Для реляционной модели логическая модель представлена в виде схемы отношений с указанием первичных ключей и связей между ними, а также внешних ключей.

При выполнении логического проектирования необходимо учитывать специфику концептуальной модели данных, но можно не учитывать особенности работы системы управления базами данных.

На этом этапе выполняется удаление связи «многие-ко-многим», сложных связей, рекурсивных связей, связей с атрибутами и множественных атрибутов. Также выполняется перепроверка связи типа «один-к-одному», удаляются избыточные связи, документируются отношения, проверяется модель с использованием правил нормализации, в отношении транзакций пользователей, а также создается диаграмма «сущность-связь».

При разработке логической модели необходимо определить требования к поддержке целостности данных, обсудить разработанную логическую модель с будущими пользователями. Также на этом этапе создается и проверяется глобальная логическая модель, проверяются возможности ее развития в будущем, а на заключительном этапе формируется окончательный вариант модели.

Следует отметить, что, если в концептуальной модели присутствие связи типа «многие-ко-многим» было возможным, то в логической модели эти связи должны быть заменены на связь типа «один-ко-многим».

Следует также отметить, что если в концептуальной модели возможно присутствие сложной связи, то в логической ее необходимо заменить за счет создания промежуточных сущностей. Сложные связи заменяются необходимым количеством бинарных связей, которые связаны с этой сущностью.

Множественными называют атрибуты, которые имеют несколько значений для одного экземпляра. Если в концептуальной модели существует множественный атрибут, то его преобразовывают за счет создания новой сущности.

Перепроверка связи типа «один-к-одному» в процессе определения сущности предусматривает создание двух разных сущностей, которые на

самом деле могут представлять один и тот же объект предметной области. В этом случае выполняется объединение двух сущностей в одну.

Если первичные ключи объединяемых сущностей различны между собой, то один из них выбирается в качестве первичного, а другой устанавливается в виде альтернативного ключа. Связь является избыточной, если одна информация может быть получена только через нее.

В завершении этого этапа формируется упрощенная локальная концептуальная модель, из которой удаляются все структуры, которые невозможно реализовать в системе управления базами данных. После этого выполняется документирование отношений для создания логической модели данных с использованием языка DBDL, который достаточно часто применяется в реляционных системах управления базами данных.

Можно сказать, что логическое проектирование является процессом конструирования общей информационной модели с использованием отдельных моделей данных пользователей. Эта модель является независимой от технических характеристик системы управления базами данных.

Основные этапы выполнения логического проектирования приведены на рисунке 10.1.

Рисунок 10.1 – Основные этапы выполнения логического проектирования

После выполнения основных этапов разработки логической модели на следующем этапе логического проектирования выполняются работы:

- выполняется слияние или объединение локальных моделей с созданием единой глобальной модели данных с последующим выполнением анализа сущностей и связей;

- осуществляется проверка разработанной глобальной логической модели с использованием правил транзакций и нормализации;
- проверяются возможности расширения модели в будущем;
- создается окончательный вариант диаграммы «сущность-связь»;
- выполняется обсуждение глобальной модели с пользователями.

Логическая модель позволяет описать предметную область, взаимосвязи, а также ограничение на данные. За счет определенной модели предметной области определяются границы, в пределах которых существует возможность развития логической модели данных.

На следующем этапе выполняется нормализация данных, на заключительном этапе создается логическая модель, которая в последующем используется для разработки физической модели.

10.2 Нормализация. Первая, вторая, третья нормальные формы

Проектирование базы данных на основании метода нормальных форм относится к операционному процессу и предусматривает последовательный перевод отношений из первой нормальной формы в другие нормальные формы более высокого уровня. При этом каждая следующая нормальная форма создает ограничение на определенный тип функциональных зависимостей, позволяет устранить аномалии при выполнении операций с базами данных и сохранить свойства предыдущих нормальных форм.

Выделяют следующие нормальные формы:

1. Первая нормальная форма (1NF);
2. Вторая нормальная форма (2NF);
3. Третья нормальная форма (3NF);
4. Усиленная третья нормальная форма или нормальная форма Бойса-Кодда (BCNF).

Отношение находится в первой нормальной форме (1NF), когда все атрибуты являются простыми, то есть они имеют одно единственное значение. Исходное отношение строится таким образом, чтобы была первая нормальная форма.

Для того чтобы перевести отношения в следующую нормальную форму, применяется метод декомпозиции без потерь. Этот метод позволяет обеспечить, чтобы запросы на выборку данных по исходным отношениям, которые получены в результате декомпозиции, позволяли получить одинаковый результат. Основной операции этого метода выступает операция проекции.

Отношение будет находиться во второй нормальной форме (2NF), если оно находится в первой нормальной форме, и каждый не ключевой

атрибут находится в полной функциональной зависимости от первичного ключа.

Для того чтобы устранить частичную зависимость и перевести отношения во вторую нормальную форму, необходимо использовать операции декомпозиции и разложения и тем самым создать проекцию без атрибутов, которая находится в частичной функциональной зависимости от первого ключа, а также построить проекцию части составного ключа и атрибутов, зависящих от этих частей. В результате будет получено два отношения находящиеся во второй нормальной форме.

Для того чтобы далее усовершенствовать отношения, необходимо их привести в третью нормальную форму (3NF). Отношение будет находиться в третьей нормальной форме, если оно отвечает требованиям второй нормальной формы, и каждый не ключевой атрибут не транзитивно определяется первичным ключом. Отношение будет находиться в третьей нормальной форме в том случае, если все не ключевые атрибуты отношения взаимно независимы и находится в полной зависимости от первичного ключа.

Для того чтобы доказать это утверждение, необходимо учесть, что все не ключевые атрибуты находятся в полной зависимости от первичного ключа, то есть соответствует требованиям второй нормальной формы (2NF). Взаимная независимость атрибутов означает отсутствие всякой зависимости между атрибутами отношения, в том числе и транзитивной зависимости между ними. Следовательно, второе определение третьей нормальной формы состоит в его первоначальном определении.

На практике построение третьей нормальной формы в большинстве случаев является достаточным условием. Поскольку привидение отношения к третьей нормальной форме позволяет устранить избыточное дублирование.

Если в отношении существует зависимость атрибутов составного ключа от не ключевых атрибутов, то необходимо использовать усиленную третью нормальную форму.

Отношение будет находиться в усиленной третьей нормальной форме или в форме Бойса-Кодда (BCNF), если оно отвечает требованиям третьей нормальной формы и в нем отсутствуют зависимости ключей составного ключа от не ключевых атрибутов.

10.3 Избыточность данных, аномалии, Функциональные зависимости между атрибутами

Избыточность данных – это такое состояние базы данных, при котором наблюдается в таблицах излишняя информация.

Кроме избыточности данных, при работе с базой данных возможны проблемы с аномалиями обновления, попытки рассмотреть аномалии в базе данных не являются достаточно удовлетворительными. Во многих работах аномалии характеризуются противоречиями между моделью предметной областью и физической моделью, которая поддерживается системой управления базами данных.

Аномалии возникают в том случае, когда достаточно сложно представить предметную область и отразить это в физической базе данных. Можно сказать, что аномалии выражаются в не адекватности модели данных и предусматривают создание ограничений. Наиболее часто выделяют следующие виды аномалий: аномалии вставки, обновления и удаления.

Причинами возникновения аномалии вставки является то, что производится хранение в одном отношении разнородной информации. Это приводит к тому, что логическая модель не отвечает модели предметной области, и база, основанная на такой модели, будет некорректно работать.

Причинами возникновения аномалий обновления является избыточность данных, которая возникает из-за того, что в таблице размещена разнородная информация. За счет возникновения таких аномалий усложняется разработка базы данных. База данных, основана на модели, будет работать только в случае внедрение дополнительного программного кода в виде триггера.

Причинами возникновения аномалий удаления является то, что в одном отношении размещена разнородная информация. За счет возникновения таких аномалий логическая модель не отвечает модели предметной области, а база данных, работающая на такой модели, будет работать неверно.

Анализ связей между сущностями в предметных областях позволяет выделить различные функциональные зависимости. Для того чтобы определить функциональную зависимость предметной области, наиболее часто возникают ситуации, что определить возможные ключи отношения невозможно. Различают полные и частичные функциональные зависимости.

Функциональная зависимость является частичной, когда для значения не ключевого атрибута находится значение некоторых атрибутов составного ключа. Полной функциональной зависимостью является то, когда для значения не ключевого атрибута находится значение всех атрибутов составного ключа. Не ключевой атрибут функционально находится в полной функциональной зависимости от составного ключа, если он функционально зависит от ключа и не находится в функциональной зависимости составного ключа.

Если не ключевой атрибут зависит частично от составного ключа, то такая функциональная зависимость называется неполной. В качестве

примера можно привести таблицу «преподаватель-предмет», содержащую данные о личном номере, предмете, фамилии и должности, окладе и часах. В этой таблице в качестве первичного ключа выступает пара атрибутов (личный номер и предмет), значение атрибута «количество часов» зависит от атрибута «предмет», то есть имеется частичная функциональная зависимость. Аналогично наблюдается и при зависимости «предмет» и «часы». Значение атрибута «фамилия» зависит от значения атрибута «личный номер», то есть в данном случае наблюдается полная функциональная зависимость «личный номер», «предмет», «фамилия».

Разбиение исходных отношений на основании функциональных зависимостей является одной из составляющих теории проектирования реляционных баз данных. За счет формирования схемы отношений, разбиения отношений по их атрибутам с учетом функциональной зависимости повышается качество схемы реляционной базы данных. Таким образом, за счет наблюдений и учета семантики данных выполняется разбиение возможных зависимостей с выделением определенных типов функциональных зависимостей.

Этот анализ выступает основой алгоритмов проектирования реляционных баз данных. На основании анализа зависимости между сущностями и предметными областями определяются частичная, полная и функциональная зависимости.

Вопросы для самопроверки по теме 10

1. Какое основное значение логического проектирования?
2. Перечислите основные этапы логического проектирования?
3. Какие задачи выполняют формы нормализации?
4. Какие проблемы возможны при проектировании базы данных?
5. В чем выражается избыточность данных?
6. Какие бывают аномалии и когда они возникают?
7. В чем сущность функциональных зависимостей?

Тема 11. Физическое проектирование БД. Жизненный цикл приложения БД

11.1 Выбор СУБД

Этап физического проектирования состоит в определении схемы данных, то есть создании физической структуры базы данных. Схема хранит в себе зависимости между физической структурой, которая поддерживается в выбранной системе управления базами данных. Физическая структура базы данных должна адекватно отображать логическую структуру, а также обеспечивать эффективное размещение данных с организацией быстрого доступа к ним.

Результаты этого этапа должны быть за документированы и представлены на языке определения данных DDL.

Выбор системы управления базами данных (СУБД) выполняется на основании различных критериев, таких как:

- тип модели базы данных, ее соответствие требованиям рассматриваемой предметной области;
- производительность системы управления базами данных, ее функциональные возможности;
- удобство и надежность системы управления базами данных при вводе в эксплуатацию;
- стоимость системы управления базами данных и необходимость внедрения дополнительного программного обеспечения.

В качестве примера можно привести реляционную систему управления базами данных Microsoft Access, представленной на рынке программного обеспечения корпорацией Microsoft.

СУБД Microsoft Access позволяет организовать работу с различными типам данных и одновременно использовать несколько таблиц, созданных в других системах управления базами данных. Также существует интеграции с офисным пакетом Microsoft Office для интеграции текстовых документов, электронных таблиц, презентаций. С помощью новых расширений также можно организовать взаимодействие с языком разметки и разметки гипертекста HTML.

Система управления базами данных Microsoft Access относится к многопользовательским приложениям и включает надежные системы обеспечения защиты от несанкционированного доступа к файлам. Сама база данных размещена в одном файле, но многими специалистами база данных разделяются на два файла: в одном файле хранятся объекты данных (таблицы и запросы), а другой файл используется для хранения форм, отчетов макросов и модулей.

Система управления базами данных Microsoft Access позволяет объединить сведения из различных источников и представить их в одной реляционной базе данных. Созданные запросы, формы и отчеты эффективно обновляются, в том числе можно организовать поиск необходимых данных с их анализом, печатью отчетов, построением диаграмм.

В системе управления базами данных Microsoft Access сведения из каждого источника размещены в отдельной таблице. При этом при работе с несколькими таблицами необходимо установить между ними связи.

Для того чтобы выполнить отбор данных или поиск, необходимо выполнить требования к созданию запросов. За счет запроса выполняется обновление или удаление одновременно нескольких записей, выполнение встроенных вычислений.

Для того чтобы просмотреть, вводить или изменить данные, применяются формы. В формах отобраны данные из нескольких таблиц или запросов, и существует возможность вывода полученных данных на печать.

В состав системы управления базами данных Microsoft Access входят различные мастера, которые позволяют ускорить процесс разработки базы данных.

11.2 Перенос логической модели в среду целевой СУБД средствами ERwin

После разработки концептуальной модели и создания логической модели в среде ERwin существует возможность получения SQL-кода, который в дальнейшем можно преобразовать в физическую базу данных.

Программа ERwin облегчает проектирование баз данных, для этого достаточно создать графическую ER-модель, которая отвечает требованиям нормализации, вести бизнес-правила и создать логическую модель, в которая будут отображаться все элементы атрибута, отношений и группировки. Также существует возможность использования программы ERwin для создания отдельных пользовательских свойств и ввода в модель дополнительной информации.

За счет изменения и редактирования диаграмм данных создается логическая модель, которая затем преобразуется в физическую модель, и таким образом автоматически создается физическая модель базы данных.

База данных может быть спроектирована без написания отдельных SQL-предложений, например, CREATE TABLE INDEX, поскольку физическая схема формируется с использованием описательной логической модели, то приложение сразу же документируется.

Средство проектирования ERwin позволяет выполнять процедуры обратного инжиниринга, то есть создать базу данных на основании модели. Следовательно, это позволяет получить представление о структуре содержания приложение.

Система ERwin поддерживает работу с реляционными базами данных, такими как Oracle, Microsoft SQL Server, DB2, IBM Informix и Microsoft Access. Одна и та же модель может быть создана для нескольких баз данных и перенесена на платформу целевой СУБД.

Программа ERwin также позволяет интегрировать базы данных в процессе разработки приложения. За счет возможности взаимодействия с архитектурой клиент-сервер в ERwin поддерживается серверная база и форма клиентской части, что позволяет ускорить процесс разработки приложений.

11.3 Работа с индексами в СУБД Microsoft SQL Server и методы обеспечения защиты данных

При создании физической модели в программе MS SQL Server необходимо учитывать индексы, которые представлены в виде средств, ускоряющих поиск необходимых данных за счет логического или физического их упорядочивания.

Индекс представлен в виде набор ссылок, которые упорядочены по определенному столбцу таблицы. Индексы также являются наборами уникальных значений для определенной таблицы, имеют ссылки на данные, они размещены в самой таблице и являются удобным внутренним механизмом системы управления базами данных SQL Server.

В среде SQL Server существует возможность использования алгоритмов поиска необходимого значения с определенной последовательностью данных.

Для того чтобы ускорить поиск данных, данные упорядочиваются и представляются в виде многоуровневой иерархической структуры с различными элементами.

В среде SQL Server можно использовать кластерный, не кластерный и уникальный индексы. Не кластерные индексы являются наиболее типичными представителями семейства индексов. В отличие от кластерных, они не перестраивает физическую структуру таблицы, а позволяют только организовать работу с ссылками на соответствующие строки.

Для того чтобы идентифицировать необходимую строку таблицы, не кластерный индекс использует указатели на информацию об

идентифицируемом номере файла, номере страницы, содержимому столбца.

Принципиальным отличием кластерного индекса от других типов является то, что при его определении в таблице физическое размещение данных необходимо перенастроить в соответствии со структурой индекса.

Также существует уникальные индексы. В этом случае уникальное значение в индексируемом столбце гарантируется уникальными индексами. При их наличии сервер не разрешает вставлять или изменять существующие значение, для того чтобы в результате этой операции в столбце появилось два одинаковых значения. Можно сказать, что уникальный индекс является надстройкой и реализуется в виде кластерного или не кластерного индекса.

Одной из важных частей проекта базы данных является разработка средств защиты. Защита имеет два аспекта: обеспечение защиты от сбоев и обеспечение защиты от несанкционированного доступа. Для того чтобы обеспечить защиту от сбоев на этапе физического проектирования, формируется стратегия резервного копирования. Для того чтобы защитить от несанкционированного доступа, каждый пользователь должен получить доступ к базе данных и привилегии к той части базы данных, с которой он будет работать.

11.4 Проектирование приложения БД

При проектировании приложений баз данных применяют интерфейс ODBC, который был предложен компанией Microsoft в виде открытого интерфейса для предоставления доступа к данным. Он позволяет выполнить проектирование приложения в среду взаимодействие с прикладной программой, которая получила название клиента или сервера с базой данных.

В основе интерфейса ODBC находится спецификация SLI интерфейса, а также структурированный язык запросов SQL, выступающий стандартам доступа к базам данным.

Интерфейс ODBC проектировался для того чтобы поддержать максимальные потребности приложений, обеспечить унифицированный доступ любого приложения, которые используют ODBC к различным источникам данных.

Для взаимодействия с базой данных приложение «клиент-сервер» вызывает функции интерфейса ODBC, которые реализованы в специальных модулях получивших название ODBC-драйверов.

Как правило, ODBC-драйверы являются DBL-библиотеками, при этом одна DBL-библиотека может поддерживать несколько ODBC-драйверов.

Архитектура ODBC включает в себя 4 основных компонента:

- приложение клиент, которое выполняет вызовы функций ODBC;
- менеджер файлов, который загружаются и освобождаются ODBC-драйверами, необходимыми для приложений клиентов;
- ODBC-драйвер, который выполняет обработку вызовов SQL функций за счет передачи SQL серверу, выполняемого SQL оператора. В приложении клиента выполняется вызванная функции, а также источник данных, который определяется, как определенная локальная или удаленная база данных.

Основным назначением менеджера драйверов является загрузка драйвера, который соответствуют подключаемому источнику данных, позволяет инкапсулировать взаимодействие с различными источниками данных через использование различных ODBC-драйверов.

ODBC-драйверы принимают вызовы функций, осуществляют взаимодействие с приложением клиентов и выполняются следующие задачи:

- управляют коммуникационными протоколами, между приложением клиентом и источником данных;
- управляют запросами к системе управления базами данных;
- выполняют передачу данных о приложениях клиентов с ODBC и обратно;
- возвращают приложение клиенту, стандартную информацию о выполняемых вызовах ODBC-функций в виде кода возврата;
- поддерживают работа с курсорами, и управляю транзакциями.

Приложение клиент одновременно может выполнять установку соединений с несколькими различными источниками, с использованием ODBC-драйвера, а также нескольких соединений с одним и тем же источником, на основе использования ODBC-драйвера

Средство OLEDB представлено в виде интерфейса, создает возможности получения доступа к приложению клиена, унифицированного доступа к различным источникам данных.

Можно сказать, что OLEDB является методом доступа к любым данным, через стандартный COM-интерфейс, не зависимо от типа данных и места их размещения. В качестве данных могут выступать простые документы, база данных таблицы Excel.

Средства, которые предоставляют доступ к источнику данных на основании технологии OLEDB, получили название OLEDB-провайдера. Программы клиенты, которые используют доступ OLEDB-провайдера относятся к потребителям данных.

В этом случае если существует только ODBC-драйвер для доступа к определенному источнику, то используется технология OLEDB. Спецификация OLEDB, позволяет описать набор интерфейсов, который реализуемый объектами OLEDB.

Каждый объектный тип представлен в виде набора интерфейсов, с помощью спецификации OLEDB определены наборы интерфейсов базового уровня, которые реализуются любыми OLE-провайдерами.

11.5 Жизненный цикл приложения БД

В настоящее время ключевая роль в достижении развития компьютерных систем, во многом определяется развитием программного обеспечения.

Этапы жизненного цикла приложений базы данных включают:

1. Планирование разработки базы данных, этот этап связан с подготовкой действий, позволяющих с максимальной возможностью реализовать этапы жизненного цикла приложений баз данных. На этом этапе происходит согласованность общей стратегии построения информационной системы, определяются основные компоненты с выделением требуемого объема данных, необходимых ресурсов и определения общей стоимости проекта. Для того чтобы поддержать планирование разработки база данных представлена в виде корпоративной модели, в которой отражаются наиболее важные данные и связи между ними, а также отношение к различным функциональным сферам деятельности.

Планирование разработки базы данных включает разработку стандартов, в которых определенно как выполняется сбор данных, каким будет их формат, какая необходима документация и как будет выполняться проектирование и реализация приложений. Разработка и сопровождение стандартов связываются с затратами времени, причем их первоначальное внедрение сопровождается значительными ресурсами.

Определение требований в системе связано с определением диапазона действий и границ приложений баз данных, определением состава пользователей и областей применения. Прежде чем приступать к проектированию приложений баз данных, необходимо установить границы предметной области, определить способы взаимодействия приложений с другими частями информационной системы. При этом данные границы должны охватывать не только разрабатываемую систему, ее пользователей, но и предусматривать перспективы ее развития.

2. Сбор и анализ требований пользователей. Данный этап относится к предварительному этапу концептуального проектирования баз данных в процессе, которого на основании спецификации требований выполняется анализ системы, выясняются необходимые подробности.

Необходимая для проектирования информация предусматривает реализацию из следующих способов:

- опрос отдельных сотрудников в области использования ПО наблюдения за деятельностью предприятия;

- изучение документов, которые используются для того чтобы выполнить сбор и представление информации;

- анкетирование сотрудников в области полученного опыта выполнения проектирования.

3. Проектирование баз данных. На этом этапе осуществляется создание проекта баз данных, который будет поддерживать функционирование предприятия и достигать его целей. Среди основных целей проектирования баз данных следует выделить представление данных и связей между ними, создание модели данных разработку предварительного варианта проекта.

4. Выбор целевой СУБД, которая будет поддерживать создаваемое приложение баз данных. Если СУБД не выбрано, то наиболее подходящим вариантом будет применение промежуточного приложения. Однако этот выбор можно осуществлять в любой другой момент до начала выполнения логического проектирования.

5. Разработка приложений. Этот этап связан с проектированием интерфейса пользователя и прикладного программного обеспечения, позволяющего взаимодействовать с базой данных, кроме проектирования разработчику должны быть известны функциональным возможностям системы.

6. Создание прототипов. Этот этап является необязательным и направлен на создание рабочей модели приложения баз данных, в которой реализована только часть требуемых возможностей, она отражает все функции готовой системы. Прототип приложения баз данных разрабатывается для того чтобы пользователи могли его использовать в своей работе и определить какие функциональные инструменты отвечают его назначению.

7. Реализация Этот этап включает в себя разработку внешних концептуальных и внутренних баз данных, а также прикладного программного обеспечения после реализации.

8. Конвертирование или загрузка данных. Данный этап предусматривает преобразование и загрузку старой системы в новую.

9. Тестирование. На этом этапе выявляются ошибки в программном обеспечении, проверяется его соответствие всем требованиям.

10. Эксплуатация и сопровождение. На этом этапе приложение база данных считается полностью разработанной и реализованной, система находится под постоянным наблюдением и поддерживается. Реализация этих изменений производится через повторное выполнение некоторых перечисленных выше этапов жизненного цикла.

Вопросы для самопроверки по теме 11

1. В чем заключается физическое проектирование БД?
2. Какие критерии используются для выбора базы данных?
3. В чем заключаются преимущества использования СУБД Microsoft Access?
4. Какое средство используется для переноса логической модели в среду целевой СУБД и в чем особенности этого этапа?
5. Для чего используются индексы в MS SQL Server?
6. Какой интерфейс применяют при проектировании приложений баз данных?
7. Какие основные компоненты входят в архитектуру ODBC?
8. Перечислите и охарактеризуйте основные этапы жизненного цикла приложений базы данных

Тема 12. Модели жизненного цикла разработки программного обеспечения

12.1 Каскадная модель ЖЦ. Достоинства и недостатки, применение

Каскадная классическая модель в настоящее время используется многими специалистами для разработки программного обеспечения. Она получила впервые свою практику применения с началом отладки и написания программного кода. Эту модель также называют еще моделью кодирования и устранения ошибок.

В настоящее время основными этапами классической каскадной модели выступают: планирование, формирование требований, анализ и проектирование, конструирование, интеграция, тестирование и поддержка эксплуатации, что видно из рисунка 12.1.

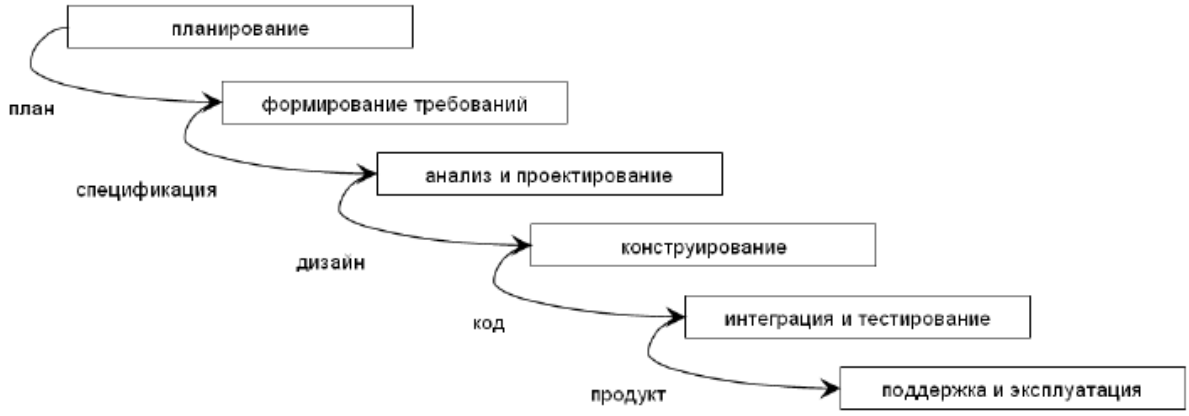

Рисунок 12.1 – Структура классической каскадной модели

Продолжение процесса рекомендуется реализовывать в виде упорядоченной последовательности, при этом должно выполняться требование, что каждая последующая фаза должна начинаться только тогда, когда завершена предыдущая фраза.

Каждая фаза имеет определенные параметры входа и выхода. В результате выполнения модели формируются внешние и внутренние данные проекта, документация на разработку программного обеспечения.

Сформированные документы по анализу требований передаются системным специалистам и разработчикам программного обеспечения. Программисты передают детальные технические характеристики программистам, а те – тестировщикам.

Переход от одной фазы на другую является формальным, поскольку на каждом этапе получается представление о разработке с проверкой качества. Как правило, прохождение стадий указывает на согласованность

командной разработки, поскольку текущая фаза завершается и может перейти на выполнение следующей фазы.

Отличительной характеристикой каскадной модели является то, что она представлена в виде формального метода, является разновидностью разработки «сверху вниз» и включает независимые фазы, которые должны обязательно выполняться последовательно.

Классическая каскадная модель впоследствии была доработана Уинстоном Ройсом, который учел взаимосвязанность этапов и необходимость возврата на предыдущие стадии.

В основе модифицированной модели находятся следующие стадии: исследование концепции, исследование системы, требования, разработка проекта, внедрение, установка, эксплуатация и поддержка, сопровождение и ввод в эксплуатацию, что видно из рисунка 12.2.

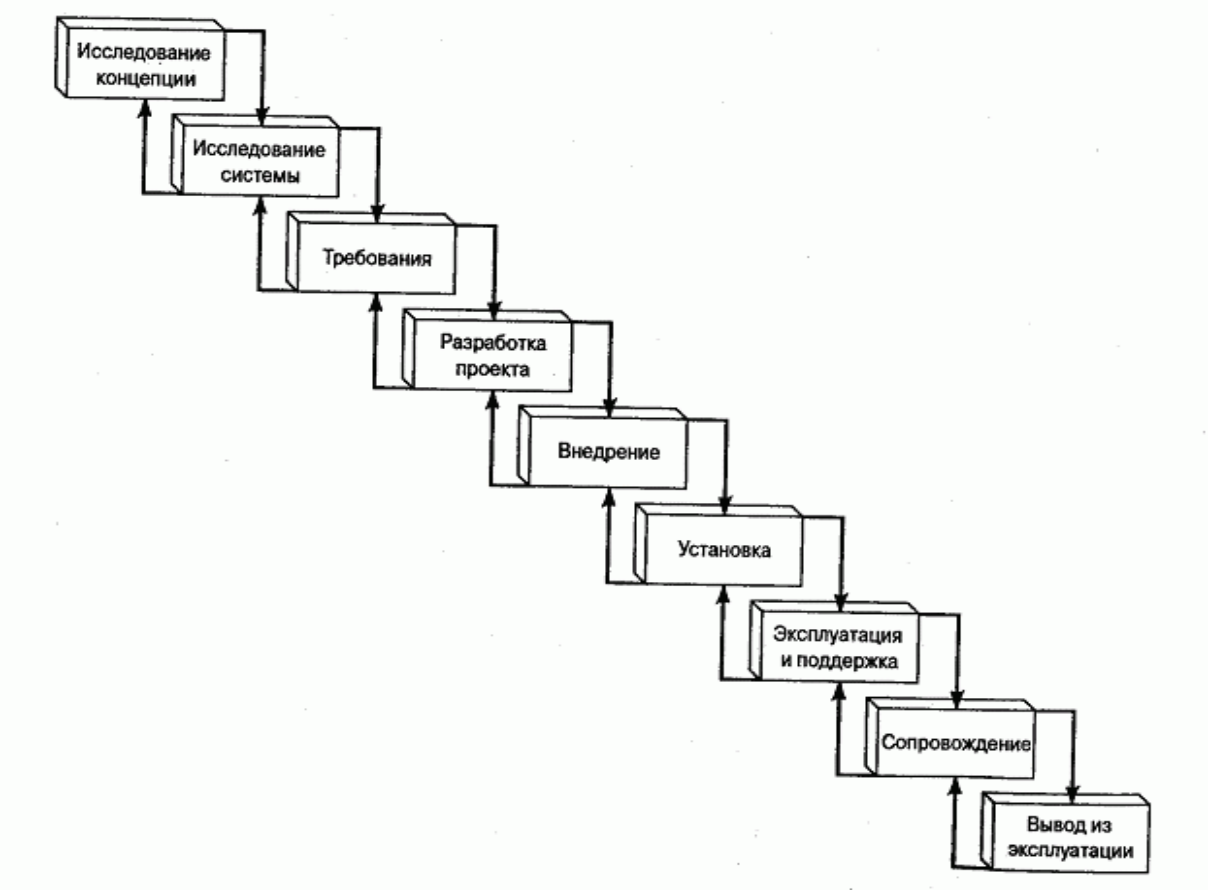

Рисунок 12.2 – Модифицированная каскадная модель

Этап исследования системы основан на исследовании требований на системном уровне. Процесс системного распределения многими разработчиками исключается, но формируются требования к аппаратному и программному обеспечению, функции, которые будут выполняться программным обеспечением в соответствии с общей архитектурой.

Процесс определения требований предусматривает формирование программных требований.

Процесс разработки проекта предусматривает разработку технических характеристик программной системы и включает этапы, связанные с разработкой структуры данных, архитектуры программного обеспечения, интерфейсных представлений.

Процесс реализации основан на эскизном описании программного обеспечения и превращении его в полнофункциональный программный продукт.

При этом выполняется разработка исходного кода, подготавливается документация и формируется база данных, которые находятся в основе физического преобразования проекта.

Если программный продукт представлен в виде прикладного пакета программного обеспечения, то к основным действиям относятся: установка и тестирование этого пакета. Если разработка программного продукта выполняется под заказ, то к основным действиям относятся: программирование и код тестирования.

Процесс установки включает установку программного обеспечения, его проверку и официальную приемку заказчиком. Процесс эксплуатации поддержки основан на запуске пользователем системы, в том числе, предоставлением технической помощи, обсуждением возникших вопросов с пользователем, регистрации запросов.

Процесс сопровождения основан на разрешении программных ошибок, неисправностей, выявлении сбоев, модификаций и внесении изменений. Процесс вывода из эксплуатации основан на прекращении работы с системой и замены ее функциональных областей.

Также возможно решение интегральных задач, в состав которых включается начало работы над проектом, проведение мониторинга, оценка и управление качеством, верификация и тестирование, менеджмент конфигурации, разработка документации и профессиональная подготовка на протяжении всего жизненного цикла.

Преимущества каскадной модели:

- каскадная модель достаточно проста в использовании и понятна конечным пользователям;
- каскадная модель является доступной для понимания, поскольку в ней основной целью является выполнение последовательно действий;
- каскадная модель предусматривает поэтапное выполнение задач.

Кроме перечисленных преимуществ следует отметить, что для каскадной модели характерна стабильность в требованиях, она представлена в виде шаблонов, которые включает в свой состав методы для выполнения анализа, проектирования, кодирования, тестирования программного обеспечения. Она позволяет выполнить требования контроля менеджмента проекта, она облегчает работу менеджера проекта в области составления плана.

При использовании каскадной модели можно отметить следующие недостатки:

- в основе модели находится последовательная линейная структура, поэтому попытки вернуться на обратные фазы или исправить какую-то проблему существенно увеличивают затраты на разработку программного обеспечения;
- она не несет смыслового характера и не является показателем для менеджера проекта;
- интеграция всех полученных результатов происходит внезапно на завершающей стадии работы модели.
- пользователи не могут получить подтверждение качества разработанного продукта до окончания всего процесса разработки;
- у пользователя нет возможности постепенного привыкания к системе;
- процесс обучения происходит на конце жизненного цикла.
- для каждой фазы определяются результаты и их можно считать завершенными только после разработки всей модели;
- возникает необходимость в жестком управлении, постоянного контроля этапов реализации модели. Модель основана на документации, а значит, количество документов является избыточным.

Весь программный продукт предусматривает выделение нескольких задач и жесткого распределения финансирования проекта. Из-за большого количества недостатков каскадная модель ограничивается в области применения, ее достаточно хорошо применять в областях, где требуется разработка небольших программных продуктов. Если компания имеет достаточный опыт построения разработки информационных систем, и проекты являются масштабными, то каскадную модель использовать неуместно. При всей критике этой модели следует также отметить, что модифицированная версия каскадной модели является менее жесткой, но включает больше этапов, которые позволяют повысить качество разработки программного обеспечения.

12.2 Поэтапная модель с промежуточным контролем ЖЦ. Достоинства и недостатки, применение

Поэтапная модель с промежуточным контролем жизненного цикла получила название также инкрементной модели, поскольку она направлена на разработку программного обеспечения в соответствии с линейной последовательности стадий, включающих несколько инкрементов или версий, то есть, предусматривает запланированное улучшение

программного продукта в процессе жизненного цикла разработки программного обеспечения.

К основным этапам разработки инкрементной модели относятся: разработка требований, проектирование, реализация, тестирование и ввод действия, что видно из рисунка 12.3.

Рисунок 12.3 – Поэтапная модель с промежуточным контролем жизненного цикла

Разработка программного обеспечения, как правило, выполняется с итерациями и предусматривает цикл обратной связи между этапами. Межэтапные корректировки позволяют учитывать реально существующее взаимовлияние результатов разработки на различных этапах, а также время жизни этапов.

На первоначальном этапе работы над проектом необходимо определить основные требования к системе с выделением более важных этапов и менее важных этапов, после этого начинает выполняться разработка системы. При этом используется принцип приращений.

Каждый инкремент добавляет системе определенную последовательность. При этом выпуск начинают с компонентов, имеющих высший приоритет. Когда часть системы определена, берется первая часть и начинается ее детализация с использованием наиболее подходящего процесса.

Также существует возможность возврата на предыдущий этап. Если часть система готова, то она предоставляется заказчику, который может ее использовать в работе. Это создает возможности клиенту уточнения требований для следующих компонентов, затем выполняется разработка следующих частей.

Ключевым этапом этого процесса является простая реализация подмножества требований к программному продукту и совершенствование модели на основании последовательных релизов.

Для жизненного цикла этой модели характерны сложность и комплексность систем, четкое видение системы со стороны заказчика и предоставление конечных результатов. Разработка версиями ведется по разным причинам, среди которых следует отметить:

- отсутствие у заказчика возможности финансирования всего дорогостоящего проекта;
- отсутствие у разработчика необходимых ресурсов для того, чтобы реализовать сложный проект в краткие сроки;
- необходимость выполнения требований поэтапного внедрения и освоения программного продукта конечными пользователями.

Внедрение всей системы сразу может вызвать у пользователей непринятие этой системы и привести к заморозке всего процесса перехода на новые технологии.

Достоинства применения поэтапной модели с промежуточным контролем жизненного цикла:

- ускоряются процессы получения отзывов о проделанной работе;
- у заказчиков существует возможность озвучивания своих комментариев в отношении готовых частей;
- заказчики могут быстро получить и освоить программное обеспечение;
- более гибкая модель в сравнении с использование с каскадной моделью.

Недостатки поэтапной модели с промежуточным контролем жизненного цикла:

- менеджеры должны постоянно учитывать прогресс процесса, в случае быстрой разработки не стоит создавать документооборот для каждого минимального изменения версии;
- структура системы ухудшается в случае добавления новых компонентов, а постоянные изменения нарушают ее структуру. Для того, чтобы избежать этого процесса, необходимо дополнительное время и денежные затраты.

Поэтапную модель с промежуточным контролем жизненного цикла применяют, если необходимо оперативно учитывать возникающие изменения и уточнять требования к программному обеспечению. При этом согласование результатов разработки с пользователями выполняются после завершения каждого этапа работ. Однако, пользователи получают программный продукт, который не отвечает их реальным потребностям.

12.3. Спиральная модель ЖЦ. Достоинства и недостатки, применение

Спиральная модель предусматривает выделение витков спирали, которые выполняются по очередности в соответствии с новой

выпускаемой версией продукта. При этом на каждом витке производится уточнение требований проекта, оценивается его качество и планирование работ для следующего витка.

Особое внимание необходимо уделить начальным этапам разработки, а именно, анализу и проектированию, где реализуемость тех или иных решений постоянно проверяется и обосновывается на основании создания прототипов.

Спиральная модель представлена в виде процесса разработки программного обеспечения, включающего в себя такие этапы, как проектирование, прототипирование и сочетает в себе преимущества входящей и нисходящей концепции, основанной на начальных этапах жизненного цикла, а именно, на анализе и проектировании.

Структура спиральной модели жизненного цикла разработки программного обеспечения приведена на рисунке 12.4.

Рисунок 12.4 – Структура спиральной модели жизненного цикла разработки программного обеспечения

Отличительной особенностью этой модели является учет рисков, которые влияют на организацию жизненного цикла.

Каждый виток спирали направлен на создание работоспособной версии системы. Это создает возможности уточнения требований, целей и характеристик проекта, определения качества разработки, планирование работы следующего витка спирали.

Следовательно, производится углубление, последовательно детализируются основные этапы проекта. В результате создается конечный вариант, который позволит удовлетворить все текущие потребности пользователей или заказчиков.

Разработка итерациями позволяет определить спиральный цикл создание системы. Неполное завершение этапа работ на каждом этапе предусматривает переход к следующему этапу после завершения текущего.

Главной задачей этого процесса является отображение пользователям работоспособного продукта, и тем самым активизируется процесс уточнения и дополнения требований.

К достоинствам спиральной модели относятся:

- создает возможности быстрого показа пользователям работоспособного программного продукта с активизацией процесса уточнения и дополнения требований;
- допускает внесение изменений в требования при разработке программного обеспечения, что характерно для многих проектов, в том числе и типовых;
- в модели предусмотрена возможность выполнять гибкое проектирование, поскольку в ней реализованы преимущества каскадной модели, а в то же время разрешены итерации по фазам проекта этой модели;
- создает возможности получения более надежной и устойчивой системы по мере развития программного обеспечения, своевременно выявить ошибки и слабые места, обнаружить и исправить ошибки на каждой итерации;
- модель создает возможность пользователям активного принятия участия в процессах планирования, анализа рисков, разработки, а также при выполнении оценочных действий. С внедрением модели снижаются риски заказчика, заказчик может с минимальными для себя финансовыми потерями завершить развитие неперспективного проекта;
- организована обратная связь по направлению от пользователей к разработчикам с выполнением высокой частоты на ранних этапах проекта и обеспечением необходимого продукта высокого качества.

Недостатки спиральной модели:

- если проект имеет низкую степень риска или небольшие размеры, то модель может быть достаточно дорогостоящей, анализ рисков после прохождения каждого витка спирали предусматривает большие затраты;
- жизненный цикл моделей имеет усложненную структуру, поэтому затрудняется применение разработчиками, заказчиками, менеджерами;
- спираль может продолжаться до бесконечности, поскольку каждая ответная реакция заказчика на созданную версию может порождать новый цикл, что отдаляет окончание работ над проектом;
- большое количество промежуточных циклов может привести к необходимости в обработке дополнительной документации.

Основной проблемой спиральной модели является выявление моментов перехода на следующий этап. Для того, чтобы решить эти

проблемы, вводятся временные ограничения на каждый из этапов жизненного цикла, и переход осуществляется на основании плана, даже если вся запланированная работа закончена. Планирование осуществляется с использованием статических данных, которые получены на предыдущих этапах проекта.

Областью применения спиральной модели является разработка проектов, в основе которых находятся новые технологии, новые серии продуктов.

Вопросы для самопроверки по теме 12

1. Чем отличается классическая каскадная модель от модифицированной?
2. В чем заключаются преимущества и недостатки каскадной модели?
3. В каких ситуациях лучше использовать каскадную модель?
4. Какие особенности можно выделить в поэтапной модели с промежуточным контролем жизненного цикла?
5. В чем заключаются преимущества и недостатки поэтапной модели с промежуточным контролем жизненного цикла?
6. В какой области применяется поэтапная модель с промежуточным контролем жизненного цикла?
7. Что отличает спиральную модель от других моделей и почему?
8. В чем заключаются преимущества и недостатки спиральной модели?
9. Для каких проектов применяется спиральная модель?

Тема 13. Стандарты, регламентирующие ЖЦ ПО и понятия автоматизированной системы

13.1 ГОСТ 34.601-90

Жизненный цикл программного обеспечения представляет собой период времени с момента принятия решения о необходимости разработки программного продукта до момента его изъятия из эксплуатации.

В стандарте ГОСТ 34.601-90 выделены следующие стадии и этапы разработки программного обеспечения:

1. Формирование требований к автоматизированной системе с выполнением обследования объекта и обоснованием необходимости разработки программного продукта, формированием требований пользователя, отчета о выполнении работ и заявок на разработку автоматизированной системы.
2. Разработка концепции автоматизированной системы с выполнением исследования объекта, проведением необходимых научно-исследовательских работ, разработка вариантов концепции и выбора варианта концепции, который удовлетворяет требованиям пользователей с оформлением отчета о проделанной работе.
3. Разработка технического задания и его утверждение на разработку автоматизированной информационной системы.
4. Эскизный проект, включающий этапы разработки и предварительных проектных решений и документации.
5. Технический проект, включающий разработку проектных решений по системе и ее компонентов, документооборот, его оформление на поставку комплектующих изделий, а также проектирование смежных частей проекта.
6. Разработка рабочей документации на автоматизированную систему и ее части с выполнением разработки и адаптации программного обеспечения.
7. Ввод в действие с подготовкой объекта автоматизации, персонала, комплектации автоматизированной системы, поставляемыми изделиями, включающими технические и программные средства, программно-технические комплексы, информационные системы.
9. Проведение предварительных испытаний и ввод в опытную эксплуатацию, а также проведение приемочных испытаний.
10. Сопровождение автоматизированной системы.

Этот стандарт не вполне подходит для разработки программного обеспечения в настоящее время.

13.2 Стандарт ISO/IEC 12207:1995

Стандарт ISO/IEC 12207:1995 был предложен Федеральным агентством по техническому регулированию и метрологии. Это стандарт, включает терминологию, которая позволяет установить общую структуру процессов жизненного цикла программных средств, и на которую можно ориентироваться в программной индустрии. В стандарте определены процессы, виды деятельности, задачи, которые используются при покупке программного обеспечения, а также при поставке разработки и сопровождения.

В стандарте выполнена группировка различных видов деятельности, которые используются в течение всего жизненного цикла программного обеспечения с выделением процессов, приведенных на рисунке 13.1.

Рисунок 13.1 – Основные процессы по стандарту ISO/IEC 12207:1995

Также выделяются основные и вспомогательные процессы. В состав основных процессов входит покупка, поставка, разработка, эксплуатация и сопровождение.

Вспомогательные процессы предусматривают документирование, управление конфигурацией, обеспечение качества, верификацию и аттестацию, совместную оценку, аудит и разрешение процесса.

Отдельной группой являются организационные процессы, которые предусматривают управление действиями, задачами, создание инфраструктуры, а также предусматривает усовершенствование жизненного цикла и обучение персонала.

В состав каждого процесса включены действия. Например, в процесс покупки включают действия по инициированию покупки, подготовке заявочных предложений, подготовке и корректировке договора на покупку, выполнения контроля деятельности поставщика, а также приемку и завершение работ.

В состав каждого действия включены и задачи. Например, в подготовку заявочных предложений входят такие задачи, как формирование требований к системе, подготовка списка программных продуктов, установка соглашений и условий, а также описание технических ограничений.

13.3 ГОСТы 19.ххх, 34.ххх

ГОСТ 19.ххх позволяет установить целевое назначение, область применения, классификацию и правила обозначения стандартов входящих в единый комплекс.

В стандарте ГОСТ 19.ххх приводятся основные понятия системы программной документации, представленной в виде комплекса государственных стандартов, позволяющих установить взаимосвязанные правила для разработки, оформления и сопровождения программной документации.

В стандарте установлены требования, позволяющие регламентировать разработку, сопровождение, изготовление и эксплуатацию программного обеспечения с обеспечением унификации программных продуктов для того, чтобы организовать взаимообмен и применять разработанные программы в новых системах со снижением трудоемкости и повышением эффективности разработки, сопровождением, изготовлением, эксплуатацией программных изделий, а также автоматизацией и хранением программной документации.

Процесс сопровождения программы включает такие процессы, как анализ функционирования, развития и совершенствования программного обеспечения, а также внесение изменений и устранение ошибок.

В состав ГОСТ 19.ххх входят:

- общее положения;
- виды программ и программных документов;
- стадии разработки;
- обозначение программ и программных продуктов.

Также в стандарте ГОСТ 19.ххх приводятся основные надписи, отображаются требования к программным документам, выполненным печатным способом.

Достаточно много внимания в стандарте ГОСТ 19.ххх уделяется разработке технического задания, требованиям к его содержанию и оформлению, спецификациям, программам и методикам испытаний, текстам программ, описанию программ, ведомости держателей подлинников. Многими специалистами для разработки программных продуктов используются требования к выполнению пояснительной записки, оформлению формуляров, описанию применения требований, разработке руководства системного программиста и формированию требований оператора, а также описанию языка.

ГОСТ серии 34.ххх выступает государственным стандартом, позволяющим описать информационные технологии. В этом стандарте установлены термины и определения в области автоматизированных систем, которые используются в различных сферах деятельности человека, таких как проектирование, исследование, управление и направлены на переработку информации.

Настоящий стандарт не применяется в отношении систем, позволяющих выполнять обработку, изготовление или сборку и транспортировку каких-либо изделий.

Понятия, определенные ГОСТ серии 34.ххх обязательны для применения во всех видах документации по автоматизированным системам.

Основными понятиями стандарта выступают: автоматизированная система, включающая систему, персонал и комплекс средств, позволяющих автоматизировать деятельность, реализовать информационную технологию для того, чтобы выполнить установленные функции.

Вопросы для самопроверки по теме 13

1. Какие выделены стадии и этапы разработки программного обеспечения в ГОСТ 34.601-90?
2. Для чего предназначен стандарт ISO/IEC 12207:1995?
3. Что позволяют установить ГОСТ 19.ххх?
4. Для чего применяют ГОСТ серии 34.ххх?

Тема 14. Стандарты проектной деятельности

14.1 Стандарт ISO 21 500:2012

Стандарт ISO 21 500:2012 выступает руководством по управлению проектами и был опубликован международной организацией по сертификации ISO. В нем представлены элементы управления проектами, описаны процессы и схемы проектного менеджмента, которые достаточно важные и влияют на эффективность выполнения проекта.

Основным назначением стандарта ISO 21 500:2012 является:

- улучшение понимания практик и принципов в проектном менеджменте всеми участниками процесса управления проектами;
- описание наилучших способов влияния по ведению проекта;
- задание основы для улучшения проектной деятельности в организации;
- основа для разработки корпоративных и национальных стандартов в области управления проектами.

В стандарте ISO 21 500:2012 описывается высокоуровневая модель управления проектами. Эта модель адаптируется к любой предметной области и может использоваться на любых предприятиях, которыми выполняется реализация проектов.

Структура стандарта ISO 21 500:2012 позволяет описать области управления проектами и выделить терминологию. В стандарте выделены ключевые термины и определения в области проектного менеджмента.

Указанные термины рекомендуется использовать в области управлении проектами.

К ключевым понятиям стандарта относятся: проект, управление проектами, стратегии организации, окружение проекта, политика проектной деятельности, проектная и операционная деятельность, заинтересованные стороны, компетенции проектного персонала, жизненный цикл проекта, ограничение по проекту.

В стандарт ISO 21500 входит четыре уровня управления проектами:

- организационная среда, в состав которой входит миссия и стратегия организации, ее цели и политика. Проекты наиболее часто выступают одним способом, который позволяет достичь поставленных целей.
- окружение проекта, в состав которого входят факторы внешней и внутренней среды, непосредственно влияющие на осуществление проекта;
- организация проекта, в состав которой входит проекты, организованные на основании политики норм и требований.

Основное внимание стандарт ISO 21 500:2012 уделяет процессам управления проектами, которые позволяют выполнить:

- группировку процессов по управлению проектами с выделением инициализации, планирования, выполнения контроля и завершение;

- тематических групп, позволяющих систематизировать процесс управления проектами.

В случае сочетания тематических групп и групп процессов управления выполняется формирование отдельных рекомендуемых процессов, которые должно осуществлять руководство.

Следует отметить, что стандарт ISO 21 500:2012 позволяет описать универсальные процессы управления проектом без привязки к отрасли, в которой осуществляет свою деятельность организация. При этом все выделенные процессы управления проектом могут относиться как к отдельной части проекта, так и ко всему проекту в целом. В состав процессов управления проектом могут быть включены общие схемы, при этом необходимо соблюдать следующие требования:

- проект должен начинаться, когда исполняется этот проект организации;

- проект должен быть закончен, когда продукты проекта приняты или проект был преждевременно завершен.

Выбор того или иного способа определяется множеством факторов, таких, как цели, риски, масштабы, временные ограничения, опыт проектной команды, доступность ресурсов.

14.2 Стандарт ICB IMPA

Стандарт ICB IMPA – это стандарт, который позволяет описать международные требования к компетентности специалистов по управлению проектами, разработанные Международной Ассоциацией управления проектами.

Основным назначением стандарта ICB IMPA является отражение требований международных стандартов для повышения компетентности специалистов по управлению проектами, он выступает в качестве руководства для проведения профессиональной сертификации.

В состав структуры ICB IMPA входит так называемая диаграмма компетентности, в которую включены элементы для работы менеджера проекта.

В стандарте ICB IMPA АИС выделяются следующие виды компетенций:

- технические компетенции, позволяющие описать основополагающие элементы компетентностей, сущность управления проектами;

- поведенческие компетенции, позволяющие описать элементы, которые характеризуют поведение специалистов с позиции их компетенции в области управления проектами;

- концептуальные компетенции, позволяющее описать элементы окружения проекта. В состав этого направления входят компетентности, позволяющие охарактеризовать способности менеджера проекта, сфокусировать несколько проектов, создать систему отношений между линейными менеджерами.

В целом, в стандарте ICB IMPA АИС выделено 4846 компетенций, каждая из которых структурирована по трем группам компетенций.

Практической ценностью стандарта ICB IMPA АИС является то, что он может выступать в виде:

- основы для описания компетентностей различных специалистов в области управления проектами;

- методического руководства по созданию модели компетенций специалистов в области управления проектами;

- источника для разработки различных образовательных программ по подготовке специалистов в области управления проектами.

14.3 Руководство по своду знаний и управлению проектами PMBOK (PMBOK)

Руководство по своду знаний и управлению проектами PMBOK выступает в виде универсального стандарта, который может быть использован в виде справочных материалов в области управления проектами.

Структура стандарта PMBOK предусматривает выделение таких ключевых элементов как старт проекта, организация проекта, выполнение работ и закрытие проекта.

В соответствии с PMBOK проект должен выполняться через объединение нескольких ключевых процессов управления. Выделенные пять групп процессов представлены в виде управленческой сущности и на основании этих процессов формируются различные области знаний.

В пятой редакции PMBOK выделены следующие ключевые области:

- управление интеграцией проекта – это действия и процессы, позволяющие определить, уточнить, комбинировать, объединить и координировать различные процессы в области управления проектами;

- управление содержанием проекта – это процессы, позволяющие включить в проект ключевые работы, которые позволяют успешно завершить проект;

- управление сроками проекта – это процессы, которые позволяют обеспечить своевременное завершение проекта;

- управление стоимостью проекта – это процессы по управлению расходами и позволяет обеспечить завершение проектов в рамках утвержденного проекта;

- управление качеством проекта – это процессы исполняющей организации, а также политика, которая позволяет управлять качеством, предусматривает установление процедур и правил, а также действий в области постоянного совершенствования процессов на протяжении всего цикла;

- управление человеческими ресурсами – это процессы по организации, управлению и руководству команды проекта;

- управление коммуникациями проекта – процессы, которые позволяют своевременно создавать, собирать, распространять, хранить, получать, использовать информацию по проекту;

- управление рисками – это процессы, которые позволяют повышать вероятность возникновения благоприятных событий и снизить вероятность возникновения неблагоприятных факторов;

- управление поставками проекта – это процессы по покупке или приобретению необходимого программного обеспечения, которое производится не исполняющей проектной организации;

- управление заинтересованными сторонами проекта – это процессы, которые позволяют идентифицировать лиц, влияющих на проект или находящихся под его влиянием.

В состав каждой области знаний включаются только те процессы, реализация которых позволяет реализовать в оговоренные сроки выделенный бюджет.

Практической значимостью стандарта PMBOK является то, что его можно использовать в виде основы, позволяющей унифицировать проектную деятельность организации, также он может быть полезен для организаций, создающих общие подходы в области управления проектами.

14.4 ГОСТ Р 54 869-2011

ГОСТ Р 54 869-2011 позволяет установить требования к процессам управления проектами. Он используется для описания любой предметной области и включает в себя требования, которые могут применяться по отношению к физическим и юридическим лицам.

Основным назначением ГОСТ Р 54 869-2011 является установление требований в области управления проектом для того, чтобы обеспечить эффективное достижение его целей.

В стандарте ГОСТ Р 54 869-2011 приведены требования, которые применяются в отношении организации любого рода проектной деятельности, позволяют повысить ее эффективность, усовершенствовать процесс управления проектами.

В структуре стандарта ГОСТ Р 54 869-2011 выделены общие рекомендательные требования в области управления проектами с выделением инициализации, планирования, организации, исполнения и контроля.

В состав ГОСТ Р 54 869-2011 входят следующие ключевые разделы:

1. Инициализация проекта с формальным описанием проекта.

2. Процессы планирования проекта с привидением описания образа наиболее вероятностных искусственных действий в области управления проектом.

3. Процесс организации исполнения проекта с выделением разработанного плана действий.

4. Процесс контроля исполнения проекта с проверкой соответствия процессов и продуктов установленным требованиям.

5. Завершение проекта с формальным закрытием проекта.

6. Требования к управлению документами проекта с выделением общих требований, которые предъявляются к документообороту.

Кроме указанных разделов в ГОСТ Р 54 869-2011 выделяются:

- заказчик проекта, выступающий в виде юридического или физического лица,
- куратор проекта, отвечающий за обеспечение проекта ресурсами.

Практической значимостью стандарта ГОСТ Р 54 869-2011 является возможность его использования для формирования требований в проектной деятельности.

14.5 NCB (НТК), корпоративные стандарты

Национальные требования к компетентности специалистов по управлению проектами NCB (НТК) выступают стандартом, позволяющим описать национальные требования и учитывать накопленный опыт в области управления проектами.

В NCB (НТК) системная модель представлена в виде развернутых структур, в состав которой входят проектное управление, сгруппированные по трем блокам:

- объекты управления;
- субъекты управления;
- процессы управления.

В NCB (НТК) описано 55 элементов компетентности и каждый из этих элементов включает речевое определение, свод знаний, возможные шаги процесса, тему изучения, критерии оценки компетентности по уровням.

Практической ценностью стандарта NCB (НТК) является то, что его можно использовать в виде основы для обеспечения терминологического единства в области управления проекта.

NCB (НТК) выступает источником, позволяющим систематизировать описание модели, определить потенциальные задачи в области управления проектами, а также он является методологической основой, на основании которой можно создавать модели компетенций для специалистов по управлению проектами.

Вопросы для самопроверки по теме 14

1. Для чего применяется стандарт ISO 21 500:2012?
2. Какие виды компетенций перечислены в стандарте ICB IMPA АИС?
3. Какие ключевые области выделены в пятой редакции PMBOK?
4. В чем заключается практическая значимость PMBOK?
5. Что позволяет установить ГОСТ Р 54 869-2011?
6. В чем заключается практическая ценность стандарта NCB (НТК)?

Тема 15. Процессы ЖЦ ПО: основные, вспомогательные, организационные

15.1 Основные процессы

В соответствии с жизненным циклом программного обеспечения выделяются три группы процессов:

- 5 основных процессов, включающих приобретение, поставку, разработку, эксплуатацию и сопровождение;
- 8 вспомогательных процессов, которые позволяют обеспечить выполнение основных процессов и включают в себя документирование, управление конфигурацией, обеспечение качества, аттестации, верификации, тестирование, совместной оценки, проведения аудита, разрешения проблем.
- 4 группы организационных процессов, включающих управление, создание инфраструктуры, усовершенствование и обучение.

В состав основных процессов жизненного цикла программного обеспечения входят следующие процессы:

- процесс приобретения, он включает действие и задачи заказчика, который осуществляет покупку программного обеспечения. В состав этого процесса входят действия, связанные с инициализацией приобретения, подготовкой заявочных предложений, корректировки договора, выполнения контроля деятельности поставщиков, приемки и завершения процессов.
- процесс инициализации включает в себя задачи по определению заказчиком своих потребностей в покупке, разработки и усовершенствования системы, анализа требований в системе, формирования решения относительно покупки, проверки наличия необходимой документации, подготовку и утверждение плана работ.

Также к основным процессам относятся формирование требований к программному обеспечению, принятие условий соглашения, технические ограничения.

Заявочные предложения направлены на выбор поставщика, которым выступает организация, с которой выполняется заключение договора с заказчика на поставку программного обеспечения.

Процедура выбора поставщика включает критерии оценки выбора поставщика, подготовку и заключение договора с поставщиком, внесение изменений. Контроль деятельности поставщика осуществляется в соответствии с действиями, которые предусмотрены в процессах совместной оценки и аудита.

В процессе приемки выполняется подготовка, выполняются необходимые тесты. Завершение работ по договору осуществляется на основании удовлетворения всех условий приемки.

В процесс поставки входят действие, задачи, которые выполняются поставщиком. Он предоставляет заказчику программный продукт или услугу. В этот процесс входят действия по инициализации, поставке, подготовке ответов на заявочные предложения, подготовке договоров, планированию, выполнению контроля, проверки и оценки поставки и завершение работ. Инициирование поставки заключается в рассмотрении поставщиком заявочных предложений и принятия решений, выполнения поставленных требований и условий.

Процесс планирования включает задачи, связанные с принятием решений поставщиком относительно выполнения работ своими силами или с привлечением субподрядчика.

Разработка поставщиком плана проекта включает в себя организационную структуру проекта, ограничение ответственности, технические требования к среде и ресурсам, управление субподрядчиками.

Результаты тестирования компонентов документируются, обновляются, составлением плана интеграции.

Интеграция программного обеспечения предусматривает сборку разработанных компонентов в соответствии с планом интеграции и тестированием этих компонентов. После интеграции система подвергается квалификационным тестированиям на соответствие совокупности требований к ней, при этом также производится оформление и проверка полного комплекта документации.

В процессе установки выполняется проверка работоспособности программного обеспечения и баз данных. Если устанавливаемое программное обеспечение можно заменить на используемую систему, разработчиком обеспечивается параллельное функционирование системы в соответствии с договором. Приемка программного обеспечения предусматривает анализ результатов квалификационного тестирования и документирование результатов оценки.

Процесс эксплуатации включает действия и задачи оператора, организацию эксплуатации системы. В этот процесс включены действия, связанные с подготовкой, проведением эксплуатационного тестирования, сдачей системы в эксплуатацию, поддержкой пользователей.

Подготовительные работы включают в свой состав работы по планированию действий, работ, определению процедур локализации, разрешению проблем.

Эксплуатационное тестирование выполняется для каждой очередной редакции программного продукта, после чего этот программный продукт передается в эксплуатацию.

15.2 Вспомогательные процессы

Вспомогательные процессы жизненного цикла программного обеспечения включают в себя процесс документирования, который представлен в виде формального описания информации, которая создана в течение жизненного цикла программного обеспечения. В состав этого процесса входят действия, позволяющие планировать, проектировать, разрабатывать, выпускать, редактировать, распространять и сопровождать документы, которые необходимы для всех заинтересованных лиц таких, как технические специалисты, руководство и пользователи системы.

В процесс документирования входят действия, связанные с подготовкой, проектированием и разработкой, выпуском документации и ее сопровождением.

К вспомогательным процессам также относится процесс управления конфигурацией, основанный на использовании технических и административных процедур. Он позволяет определить состояние компонентов программного обеспечения в системе, управлять модификациями программного обеспечения, описывать и подготавливать отчеты о состоянии компонентов и запросов.

В процесс управления конфигурацией входят действия, связанные с подготовкой работ, идентификацией конфигурации, выполнением контроля конфигурации, учетом состояния конфигурации, оценкой конфигурации, управление выпуском и поставкой. Подготовительные работы основаны на планировании и управлении конфигурацией. Идентификация конфигурации позволяет установить правила, которые идентифицируются и позволяют различать компоненты программного обеспечения и их версии.

Кроме этого, каждый компонент и его версия соответствует однозначно обозначенному комплекту документации, в результате которого создается база для однозначного выбора и манипулирования версиями компонентов программного обеспечения с использованием ограниченной и упорядоченной системы символов, позволяющих идентифицировать различные версии программного обеспечения.

Контроль конфигурации основан на систематической оценке предусмотренных модификаций программного обеспечения и координированной их реализации с учетом эффективности каждой модификации и затрат на выполнение. Он позволяет обеспечить контроль состояния и развивать компоненты программного обеспечения, а также реально изменяющиеся компоненты и комплект на документацию.

Учет состояния конфигурации основан на регистрации состояния компонентов программного обеспечения, подготовке отчетов обо всех реализованных и отвергнутых модификациях версий компонентов. Совокупность отчетов позволяет обеспечить однозначное определение

текущего состояния программной системы и входящих в ее состав компонентов, а также вести историю модификаций.

Оценка конфигурации основана на оценке функциональной полноты компонентов, а также определении их соответствия текущему состоянию и приведенному техническому описанию. Управление выпуском и процессы поставки основаны на изготовлении эталонных копий программного обеспечения и документооборота, поставке и хранении пользователем в соответствии с порядком, который принят в организации.

Процесс обеспечения качества основан на гарантиях того, что программное обеспечение и все процессы жизненного цикла соответствуют поставленным требованиям и составленным планам. Под качеством программного обеспечения предусматриваются свойства, позволяющие охарактеризовать способность программного обеспечения на выполнение требований пользователей. Для того, чтобы получить достоверную оценку по создаваемому программному обеспечению, в него вносятся изменения. Процесс обеспечения качества включает подготовительные работы, обеспечение качества процессов и продуктов, а также прочих показателей.

Подготовительные работы основаны на координации с другими вспомогательными процессами, а также планировании самого процесса с учетом используемых стандартов, процедур, методов и средств для обеспечения качества продукта. Обеспечение качества процесса основано на гарантии соответствия процессов жизненного цикла программного обеспечения методам разработки, среды разработки и квалификации персонала.

Процесс верификации основан на определении того, что программный продукт относится к результатам определенного действия, он полностью позволяет удовлетворить требования и условия, а также предшествующее действие. Для того, чтобы повысить эффективность верификации, должна как можно раньше быть интегрирована с использующимися такими процессами, как поставка, разработка, эксплуатация. Верификация выполняется с учетом степени независимости, профессиональных требований специалистов. Если процесс верификации выполняется непосредственно организацией, которая зависит от поставщика и разработчика, а также оператора или службы сопровождения, то этот процесс получил название независимой верификации.

В процесс верификации входят такие действия, как подготовительные работы, верификация. В ее процессе устанавливается непротиворечивость требований в соответствии выбранных процессов в условиях договоров, корректность описания.

Процесс аттестации основан на определении полноты соответствия заданных требований и программного продукта их функциональному

назначению. Аттестация предусматривает подтверждение и выполнение оценки достоверности тестирования программного обеспечения, в ее процессе гарантируется установление полного соответствия программного обеспечения требованиям и документации.

Процесс совместной оценки основан на оценке состояния работ по проекту и программного обеспечения, которая создается при выполнении этих работ. Он сосредоточен в основном на контроле и планировании ресурсами, материально-техническим обеспечением персонала и другими средствами. Оценка осуществляется как на уровне управления проектом, так и на уровне технической его реализации. Процесс совместной оценки предусматривает выполнение подготовительных работ, выполнение оценки управления проектом и техническую оценку.

Процесс аудита представлен в виде требований, условий и плана. Он осуществляется несколькими участниками, отраженными в договоре на проведение аудита. Можно сказать, что аудит является ревизией или проверкой, которая проводится компетентными органами для того, чтобы обеспечить независимую оценку степени соответствия программного обеспечения или процессов текущим требованиям. За счет аудита устанавливается соответствие между реальными работами и требованиями. Аудиторы не должны иметь прямую зависимость от разработчиков программного обеспечения, с их помощью определяется состояние работ, использование ресурсов.

Процесс разрешения проблем основан на анализе и решении проблем независимо от происходящих действий или выбранного источника. Для проведения проверки процесса разрешения проблемы основана подозрительная работе и разрешении проблем.

15.3 Организационные процессы

Организационные процессы включают в свой состав процесс управления, который включает задачи и действия, выполняемые стороной, отвечающей за управление выпуском продукта, реализацией задач, связанных с приобретением, поставкой, разработкой, эксплуатацией и сопровождением программного продукта.

В процесс управления включены действия по инициализации и определению области управления, планирования и выполнения контроля, проверки и оценки. При инициализации менеджер устанавливает необходимые для реализации проекта ресурсы такие, как персонал, технологии, оборудование.

Планирование предусматривает выполнение задач по составлению графиков, оценки затрат, выделению требуемых ресурсов, распределению ответственности, оценки рисков.

Процесс создания инфраструктуры включает в себя выбор и поддержку технологий и стандартов, а также аппаратных и программных средств, которые будут использоваться для разработки и эксплуатации программного обеспечения. За счет инфраструктуры устанавливаются требования к процессам. Процесс создания инфраструктуры включает подготовительные работы, непосредственно создание инфраструктуры и ее сопровождение.

Процесс усовершенствования основан на оценке, измерении, контроле усовершенствованных процессов. В этот процесс включаются действия по созданию процесса, его оценке и усовершенствованию процессов, связанных с повышением производительности труда, всех участников и специалистов, совершенствование методов управления, технологий, выбор средств для обучения.

Процесс обучения включает в себя первоначальное обучение и последующие с повышением квалификации персонала.

Содержание процесса обучения должно соответствовать требованиям к проекту. В нем должны учитываться все необходимые технические средства и ресурсы. Должны быть разработаны и представлены методические материалы, позволяющие получить пользователям представления о проекте, изучить его основные технологические этапы.

Вопросы для самопроверки по теме 15

1. Какие процессы входят в состав основных процессов жизненного цикла программного обеспечения?
2. Что позволяет получить интеграция программного обеспечения?
3. Какие процессы относятся к вспомогательным?
4. Процесс обучения к каким процессам относится?

Тема 16. Модели процесса разработки ПО

16.1 «Тяжелые» и «легкие» процессы разработки. Достоинства и недостатки

В основе «тяжелых» и «легких» процессов разработки находятся унифицированные процессы (Rational Unified Process, RUP) или экстремальное программирование (Extreme Programming, XP).

Технология RUP выступает примером «тяжелых» процессов, предусматривающих детальное описание, предусматривает поддержку собственной разработки исходного программного кода за счет реализации вспомогательных действий.

В качестве примеров подобных действий выступают разработка плана, технического задания многочисленных проектных моделей, проектной документации. Основной целью такого процесса является отделение успешных практик, разработки программных продуктов и их сопровождение от определенных людей у меня умеющих их применять на практике. За счет многочисленных вспомогательных действий создается поддержка успешного решения задач в области поддержки и конструирования сложных систем.

Для того чтобы достичь этого осуществляется иерархическое пошаговое детальное описание принимаемых в этой ситуации действий, для того чтобы в дальнейшем можно было обучить работников осуществлять эти действия аналогичным образом.

В процессе проекта разрабатывается промежуточные документы, позволяющие разработчику последовательно декомпозировать задачи и выделять более простые, эти документы являются основой для проверки правильности решений, которые принимаются на каждом этапе, с отслеживанием текущего хода работ и уточнения оценок ресурсов позволяющих получить желаемые результаты.

Технология RUP является достаточно сложным интерактивным процессом. В ней весь процесс выполнения работ направлен на достижение итоговых целей проекта и представлен в виде вариантов использования.

Варианты использования позволяют реализовать сценарии взаимодействия между пользователями. Разработка предусматривает выделение вариантов использования на каждом этапе с контролем степени приближении их к реализации. Основой является архитектура, в которую входят наборы компонентов для разработки программного обеспечения, а также выделяется ответственность каждого из компонентов.

Архитектура выступает одной из основ для того чтобы получить качественное программное обеспечение, а также планировать работы и оценки в терминах времени и ресурсов позволяющих достичь определенных результатов. Для этого используются набор графических моделей разработанных на языке UML.

В качестве процесса разработки выступают управляемые и планируемые итерации, объемы которых представлены в виде архитектуры.

В технологии RUP выделены четыре фазы жизненного цикла, в каждую из которых включено несколько итераций.

Первая фаза «Начало проекта» и ее целью является получение компромисса между всеми заинтересованными лицами, а также решение задач, связанных с выделением ресурсов. На этой стадии определяются основные цели проекта, бюджет, руководство, выбираются технологии и инструменты, ключевые исполнители, также выбираются технологии позволяющие достичь поставленных целей.

Фаза «Проектирование» основана на формировании требований, разработку стабильной базовой архитектуры продукта, которая позволяет решать различные задачи. Как правило, на эту фазу уходит около 30% времени и она занимает 20% трудоемкости одного цикла.

Третья фаза «Построение». Ее основной целью является детальная проработка требований и разработка системы, которая удовлетворяет этим требованиям, с проектированием архитектуры.

Фаза «Внедрение» направлена на обеспечение системы, предоставление доступа к системе конечных пользователей. На этой стадии разворачивается система в определенной рабочей среде, проводится ее тестирование, исправляются ошибки, учитываются требования пользователей.

Технология экстремального программирования XP является примером «легких» процессов. За счет методов фиксируются схемы действий, позволяющие обеспечить большую гибкость на каждом этапе разработки проекта. В ней также не используется дополнительные документы, не вносятся непосредственно изменения в готовую работающую программу.

Модель процесса по XP представлена в виде частой последовательности выпусков программных продуктов, при этом не обязательно, чтобы каждый выпуск программного обеспечения был представлен в виде полной цельной версии.

К основным принципам организации процесса по XP относятся:

- планирование, основанное на принципах, что разработка программного обеспечения позволит обеспечить диалог между потребностями и желаниями пользователей;
- метафора, отражающая сущность проекта;

- рефакторинг, связанный с процессом постоянного улучшения структуры программного обеспечения и позволяющий добавить новую функциональность;

- парное программирование, основанное на том, что один специалист осуществляет программирование, а другой осуществляет ее тестирование и упрощение структуры;

- коллективное владение программным кодом и участие заказчика в разработке;

- создание и использование стандартов кодирования, написание программного кода, тестирование, основанное на тестировании программного обеспечения.

Однако, в полном объеме XP использовать нельзя, поэтому внедряются отдельные практики, такие как парное программирование. совместное владение программным кодом, рефакторинг кода.

16.2. Гибкая методология разработки ПО (Agile)

Гибкая методология разработки ПО Agile включает в себя методы разработки, которые являются альтернативой формальным и тяжеловесным методологиям, подобным RUP.

Профессиональные программисты, желая не превращать процесс разработки программного обеспечения в сложный процесс и обеспечить свободу, предложили технологию высокой эффективности, получившей название технологии Agile.

Основными положениями гибкой методологии Agile является взаимодействие между процессами и программными средствами, работа программного обеспечения вместо использования сложной документации, организация взаимодействия между заказчиком, реализация реакции на изменения в соответствии с планом.

Гибкая методология Agile используются небольшими самоорганизующимися командами, включающими высококвалифицированных специалистов, ориентированных в своих действиях на бизнес.

Вопросы для самопроверки по теме 16

1. Какая технология выступает типичным примером «тяжелых процессов»?
2. В чем выражены особенности технологии RUP?
3. Какие можно выделить фазы в технологии RUP?
4. Какая технология является примером «легких» процессов?
5. Перечислите основные принципы организации процесса по XP
6. В чем выражаются особенности гибкой методологии разработки ПО Agile?

Тема 17. Основы объектно-ориентированного проектирования ПО

17.1 Принципы объектно-ориентированного проектирования

Для объектно-ориентированного проектирования характерно наличие концептуальной базы, представленной в виде объектной модели, для которой характерно абстрагирование, инкапсуляция, модульность, наследование и иерархия. Эти свойства являются главными характеристиками этого вида проектирования.

Абстракция позволяет выделить наиболее важные характеристики объекта, которые его отличают от других объектов и позволяют определить концептуальные границы с точки зрения наблюдателя.

Инкапсуляция является процессом, позволяющим отделить друг от друга элементы объекта, определить их устройство и поведение.

Модульность позволяет разложить систему на внутренние модули, причем иерархия выступает упорядоченной абстракцией, размещенной между уровнями.

С помощью наследования создается иерархия абстракций, в которых выделяются подклассы.

В основе объектно-ориентированных принципов находится разработка пользовательского интерфейса, позволяющего организовать взаимодействие между компьютером и оператором. Он выступает необходимой составляющей в информационной системе.

Среди основных принципов объектно-ориентированного проектирования следует выделить:

- скрытие от пользователя внутреннего строения информационной системы, что создает возможности концентрации на выполнении задач, организация взаимодействия между приложениями и объектами, то есть, у пользователя созданная возможность одновременной работы, как с приложением, так и с его объектами;
- возможность использования окон и представлений, при этом все основные компоненты размещаются в среде пользователя.

В объектно-ориентированном проектировании также выделяется понятие графических элементов управления, в качестве которых выступают кнопки, флажки и закладки. Операции, позволяющие выполнять манипулирования данными, осуществлять взаимосвязь с объектами и выделять интуитивно понятные действия с ними.

17.2. Унифицированный язык моделирования (UML)

В основе реализации объектно-ориентированных подходов к проектированию находится унифицированный язык моделирования.

Язык UML представлен в виде общецелевого языка визуального моделирования, предложенного для визуализации и спецификации, проектирования и документирования компонентов программного обеспечения, бизнес-процессов и других систем.

С помощью языка UML выполняется моделирование, которое может эффективно использоваться для создания логических, концептуальных и графических моделей сложных систем.

Конструктивное использование языка UML основано на понимании общих принципов моделирования сложных систем и выявлении особенностей процесса объектно-ориентированного проектирования.

При этом второстепенные детали опускаются для того, чтобы снизить сложность анализа и получения модели.

Диаграммы UML представлены в виде графического представления и с их помощью диаграмм создаются системы визуализации.

В языке UML выделены следующие типы диаграмм:

- диаграмма вариантов использования, позволяющая выполнять моделирование бизнес-процессов организации;
- диаграмма классов, позволяющая моделировать статические структуры классов системы и связей между ними. На этих диаграммах отображены классы, объекты, интерфейсы, кооперации и их отношения.
- диаграммы поведения взаимодействия, которые позволяют смоделировать процесс для обмена сообщениями между объектами. В состав диаграмм взаимодействия входят диаграммы последовательности и кооперативные диаграммы.
- диаграммы реализации включают в свой состав диаграммы компонентов, позволяющие моделировать иерархию компонентов и диаграммы размещения для моделирования физической архитектуры системы.

17.3. Паттерны проектирования ПО

Шаблон проектирования или паттерн представляет собой повторяемую архитектурную конструкцию, представленную в виде шаблона, позволяющего решить проблемы проектирования, для реализации определенного проекта наиболее часто шаблон не относится к законченным образцам, поэтому в дальнейшем его можно преобразовать в код и применять для решения задачи в различных ситуациях.

Объектно-ориентированные шаблоны позволяют установить взаимоотношения между классами и объектами, и выделить конечные классы или объекты которые будут использоваться в проекте.

В настоящее время наиболее популярными являются паттерны проектирования. Среди них выделяют архитектурные паттерны, паттерна проектирования и идиомы.

Задачей каждого паттерна является представление проблемы и ее решение и для этого используются различные форматы описаний.

Преимуществами использования паттернов в процессах проектирования являются:

- паттерны создают возможность накопления опыта экспертов и предоставления его рядовым разработчикам;
- паттерны позволяют создать словарь, в основе которого находится организация взаимодействия между разработчиками;
- паттерны упрощают реструктуризацию системы, в независимости от того используется он при проектировании или нет.

Правильно выбранные паттерны проектирования создают возможности сделать программный продукт более гибким, облегчить и модифицировать его работу, упростить работу с программным кодом и повторно его использовать выделяют.

Вопросы для самопроверки по теме 17

1. Что характерно для объектно-ориентированного проектирования?
2. Перечислите основные принципы объектно-ориентированного проектирования
3. Что представляет собой язык UML?
4. Какие виды диаграмм позволяет разработать язык UML?
5. Что такое паттерн и какие виды паттернов существуют?
6. В чем заключаются преимущества паттернов?

Тема 18. Проектирование пользовательского интерфейса и основные фазы по завершению проекта

18.1 Принципы проектирования пользовательских интерфейсов

За счет использования пользовательского интерфейса обеспечивается взаимодействие между персональным компьютером и пользователем, организуется обмен, а также вырабатывается ответная реакция.

При проектировании пользовательского интерфейса достаточно много внимания уделяется юзабилити, отражающей степень, как программный продукт должен использоваться пользователями для достижения определенных целей, эффективности, продуктивности и удовлетворенности пользователей.

Исследования в области взаимодействия пользователя и персонального компьютера показывают, что наиболее часто пользовательским интерфейсом должны быть реализованы следующие функции:

- возможность управления персональным компьютером за счет осуществления пользовательских действий, среди которых следует выделить инициацию, прерывание и отмену компьютерных процессов;
- вводить данные от пользователя с обеспечением обратной реакции системы;
- отображать данные пользователя и его поддерживать в процессе работы за счет организации обратной связи, сбора информации о случайных и ошибочных действиях пользователя.

Правильно спроектированный пользовательский интерфейс должен выполнять следующие требования:

- иметь низкий уровень вхождения, а также позволять быстро осваивать его пользователям с формированием устойчивых практических навыков;
- предоставлять информацию на естественном языке без демонстрации процесса внутренних вычислений;
- удовлетворять потребности пользователей без заострения внимания на процессах обработки данных.

Для того чтобы получить эффективный результат при разработке пользовательского интерфейса, применяют различные подходы к проектированию. Основными из них являются подходы, ориентированные на пользователя. Для этого подхода характерно активное вовлечение пользователей в процесс проектирования и тестирование с выделением пользовательских задач и требований, оптимальное распределение функций между технологиями и пользователями.

Подходы к проектированию интерфейсов основаны на утверждении, что пользователь должен адаптироваться к входящим в состав интерфейса инструментам.

Среди существующих подходов к проектированию пользовательского интерфейса следует выделить:

- целостно-ориентированный подход. В его основе находятся конечные результаты пользователей, которые достигаются за счет организации взаимодействия с программным обеспечением;
- подходы, ориентированные на данных, при проектировании интерфейса должна поддерживаться модель взаимодействия пользователя с системой, при которой первичными являются обрабатываемые данные, а не используемые для этого программные средства. Другими словами, этот подход основан на выявлении данных, с которыми осуществляется работа;
- использование подхода основанного на программных объектах. В основе этого подхода находится выделение абстрактного устройства для сохранения данных, которые используются для выполнения заданий пользователя. Документ должен быть доступен из различных приложений, которые используются для обработки и организации взаимодействия с пользователем;
- интерактивный подход - это метод, основанный на последовательных приближениях. Сущность интерактивного подхода основана на создании простейшего прототипа, позволяющего удовлетворить требования, как заказчика, так и самого разработчика;
- при разработке интерфейса необходимо учитывать подходы, основанные на выборе методов, назначения разрабатываемого программного обеспечения, целевой аудитории, времени и бюджета и разработки.

18.2 Элементы управления в пользовательском интерфейсе

В большинстве случаев в настоящее время применяется графический пользовательский интерфейс, который позволяет организовать взаимодействие между персональным компьютером и пользователем. Основными элементами этого стиля интерфейса выступают пиктограммы, окна, меню и указатели.

Графический интерфейс является обязательным элементом современного программного обеспечения, поскольку ориентирован на работу конечных пользователей.

К основным достоинствам графического интерфейса относится интуитивная понятность для пользователей и наглядность, а также

возможность использования программного обеспечения, позволяющего организовать работу в графической среде.

К основным элементам управления графического интерфейса относятся:

- метка, представленная в виде текста, который не подлежит изменению при работе пользователя с экраном и формой;
- текстовое окно – это поле для ввода информации произвольного вида;
- командная кнопка, представленная в виде объекта, позволяющего обеспечить передачу управляющих воздействий, например, кнопки OK, сохранить, отменить;
- кнопка переключатель, которая является элементом для альтернативного выбора или выполнения групповых команд;
- помеченная кнопка, относится к элементу, который позволяет выбрать несколько групп и команд из одноименных команд;
- окно-список, выступающее элементом, в состав которого включены альтернативные значения для выбора;
- комбинированное окно, выступающее элементом, в котором объединяются возможности окна-списка и текстовых окон, с возможностью внесения данных с клавиатуры;
- линейка горизонтальной прокрутки, относится к элементам, которые позволяют произвести быстрое перемещение внутри длинного списка или текста по горизонтали;
- линейка вертикальной прокрутки, относится к элементам, которые позволяют произвести быстрое перемещение внутри длинного списка или текста по вертикали.

Графический интерфейс позволяет поддерживать пользователям различные виды диалога. Диалог представлен в виде обмена информационными сообщениями и предусматривает выполнение операции по приему, обработке и выдаче сообщений в режиме реального времени. Диалог является двусторонним обменом информацией между пользователями и персональным компьютером.

В настоящее время выделяют следующие виды диалога:

- жесткий диалог - это вид диалога, при котором участники ведут обмен сообщениями в жестко заданных условиях с использованием определенных принципов;
- гибкий диалог - это вид диалога задается с использованием предписанных вариантов диалога, которые предъявляются пользователю в виде меню, он представлен в виде иерархической структуры, из которой выбирается направление решения задач;
- свободный – это вид диалога создает возможности обмена сообщениями произвольным образом.

Для того, чтобы организовать диалог в программном интерфейсе применяются меню, команда, шаблон и естественный язык. Реализация диалога в виде меню предусматривает вывод экран определенных функций.

В этом случае пользователем выбирается на экране монитора необходимая ему операция и осуществляется передача ее на исполнение. Меню выступает набором операций, которые могут выполняться персональным компьютером по определенной программе.

Шаблон является режимом взаимодействия конечного пользователя на каждом этапе восприятия информации. Диалог вида команда должен быть инициирован пользователем. При этом необходимо выполнить одно из допустимых действий.

Если перечень отсутствует на экране, то его можно вызвать с помощью специальной директивы или функциональных клавиш. Естественный язык является таким типом диалога, при котором запрос на ответ со стороны пользователя ведется на естественном языке. Пользователь в этом случае имеет возможность свободной формулировки задачи, с набором фраз, слов и синтаксиса языка.

У системы существует возможность уточнения формулировки пользователя. Разновидностью этого вида диалога является организация речевого общения.

Для построения пользовательских интерфейсов необходимо соблюдать принципы структуризация, простоты, видимости, обратной связи, толерантности и повторного использования.

Принцип структуризации основан на том, что пользовательский интерфейс должен быть структурирован, при этом отдельные его части должны быть связаны между собой, и в то же время независимы, то есть в них должны быть выделены классификационные признаки.

Принцип простоты основан на том, что наиболее распространенные операции должны выполняться достаточно быстро, при этом должны быть ясными ссылки на более сложные процедуры.

Принцип видимости основан на том, что функции, которые необходимы пользователю, должны быть отражены в пользовательском интерфейсе.

Принцип обратной связи состоит в том, что пользователя должны быть сообщения о действиях, которые выполняются внутри системы. При этом каждое из этих сообщений должно быть кратким, однозначным и написанном на языке, понятном пользователю.

Принцип толерантности основан на гибкости, возможности исправления ошибок пользователя. При этом ошибки должны сокращаться за счет возможности отмены или повтора действий.

Принцип повторного использования основан на том, что интерфейс должен учитывать возможности внешних и внутренних компонентов, которые должны быть унифицированным.

18.3 Разработка пользовательской документации

В пользовательской документации приводятся объяснения для пользователей, как использовать программный продукт. В состав этой документации входят документы позволяющие, получить пользователю представления о настройках системы, использовании пользовательского интерфейса, организации и взаимодействии с другими системами.

При разработке пользовательской документации необходимо учитывать требования обычных пользователей и администраторов.

Обычные пользователи используют руководство для того, чтобы найти поиск функций системы в соответствии с рассматриваемой предметной области. В качестве этих пользователей может выступать инженер, который занимается автоматизацией процесса проектирования технического устройства, бухгалтер, который с помощью автоматизированной системы получает первичные документы и формирует бухгалтерскую отчетность.

Руководство администратора направлено на отражение элементов управления, администрирования, разграничения прав доступа к информационной системе, а также выполнять определенные действия для того, чтобы поддерживать программный продукт в актуальном состоянии.

В состав пользовательской документации для большинства программных продуктов входят:

- общее функциональное описание системы, в котором приводится краткая характеристика функциональных возможностей программного продукта, она направлена на пользователя, который решает, насколько ему необходим этот программный продукт;
- руководство по инсталляции программного продукта, направлено на поддержку работы системных администраторов, в нем детализируется информация по требованиям конфигурации, инсталляции, защиты программного продукта;
- справочник по применению программного продукта направлен на работу оригинальных пользователей, в нем приводится необходимая информация по использованию программного продукта, представление в удобной форме деталей пользовательского интерфейса;
- руководство по управлению информационной системой, предназначено для администраторов, в нем приводятся сообщения, которые генерируются программным продуктом, среда взаимодействия с другими системами, возможность реагирования на сообщение.

Разработка пользовательской документации начинается после создания внешнего описания. Качество этого документа влияет на принятие программного продукта пользователем и дальнейшую его эксплуатацию.

Поэтому при разработке руководства пользователю необходимо достаточно детализировать все функциональные возможности программного продукта и возможности его использования в повседневной деятельности.

Для того чтобы обеспечить качество пользовательской документации, необходимо руководствоваться требованиями стандартов, в которых приводится описание к разработке пользовательской документации, требования к каждому из перечисленных документов.

Вопросы для самопроверки по теме 18

1. Какие функции должен позволять реализовать пользовательский интерфейс?
2. Какие предъявляются требования к пользовательскому интерфейсу?
3. Какие существуют подходы для разработки пользовательского интерфейса?
4. Охарактеризуйте основные элементы пользовательского интерфейса
5. В чем выражается значимость пользовательской документации?
6. Что входит в состав пользовательской документации?

Список использованной литературы

1. Варзунов А.В. Анализ и управление бизнес-процессами: учебное пособие - СПб: Университет ИТМО., 2016. - 112 с.
2. Влацкая И.В. Проектирование и реализация прикладного программного обеспечения: учеб. пособие / И.В. Влацкая, Н.А. Заельская, Н.С. Надточий. – Оренбург: Оренбургский государственный университет, 2015. – 270 с.
3. Грекул В.И. Проектное управление в сфере информационных технологий: учеб. пособие / В.И. Грекул. – М.: Бином, 2015. – 337 с.
4. Громов А.И. Управление бизнес-процессами: современные методы монография / А.И. Громов, А. Фляйшман, В. Шмидт. - М.: Юрайт., 2018. - 367с.
5. Долганова О.И. Моделирование бизнес-процессов: учебник / О.И. Долганова, Е.В. Виноградова, А.М. Лобанова. - М.: Юрайт, 2018. — 289 с.
6. Жданов С.А. Информационные системы: учебник / С.А. Жданов, М.Л. Соболева, А.С. Алфимова. - М.: Прометей, 2015. - 302 с.
7. Затонский А.В. Информационные технологии: разработка информационных моделей и систем: учеб. пособие / А.В. Затонский - М.: ИЦ РИОР: НИЦ ИНФРА-М, 2014 – 344 с.
8. Золотухина Е.Б. Моделирование бизнес-процессов: краткий конспект лекций / Е.Б. Золотухина. - М.: Инфра-М, 2017. - 79с.
9. Иващенко И.Г. Методы и средства проектирования информационных систем и технологий: учеб. пособие / И.Г. Иващенко. - М.:МГУП имени Ивана Федорова, 2015. – 160с.
10. Крахоткина Е.В. Методы и средства проектирования информационных систем и технологий: учеб. Пособие / Е.В. Крахоткина. - Ставрополь: СКФУ, 2015. – 152 с.
11. Коцюба И.Ю. Основы проектирования информационных систем: учеб. пособие / И.Ю. Коцюба. - СПб.: Университет ИТМО, 2015. - 206 с.
12. Кузнецова П.У. Информационные технологии: учебник для бакалавров / П.У. Кузнецова. – М.: Юрайт., 2015.- 441с.
13. Куняев Н.Н. Информационные технологии: учебник /Н.Н. Куняев. - М.: Логос 2016. - 408с.
14. Липаев В.В. Документирование сложных программных комплексов: учеб. пособие / В.В. Липаев. – Саратов: Вузовское образование, 2015. – 420 с.
15. Николаев Е.И. Базы данных в высокопроизводительных информационных системах: учебное пособие / Е.И. Николаев. - Ставрополь: СКФУ, 2016. – 163 с

16. Тельнов Ю.Ф. Инжиниринг предприятия и управления бизнес-процессами. Методология и технология: учеб. пособие / Ю.Ф. Тельнов, И.Г. Федоров. – М.: Юнити-Дана, 2015. – 207 с.
17. Ширяев В.И. Управление бизнес-процессами: учеб. пособие / В.И. Ширяев, Е.В. Ширяев. – М.: Финансы и статистика, 2014. – 464 с.
18. Цуканова О.А. Методология и инструментарий моделирования бизнес-процессов: практический курс / О.А. Цуканова - Санкт-Петербург., СПб: Университет ИТМО., 2017. - 56 с.
19. Шагрова Г.В. Методы исследования и моделирования информационных процессов и технологий: учеб. пособие / И.Н. Топчиев, Г.В. Шагрова. - Ставрополь: СКФУ., 2016. – 180с.
20. Шелухин О.И. Моделирование информационных систем: учеб. пособие для высших учебных заведений/ О.И. Шелухин. – Москва: Горячая линия-Телеком., 2016. – 516с.

Шамсутдинов Тимур Фаритович

Учебно-методическое пособие «Проектирование информационных систем»

Редактор

Корректор

Подписано в печать		Формат 60x84/
Заказ №	Печать ризографическая	Усл. печ. л
Тираж экз.	Бумага офсетная № 1	Уч. -изд. л

Издательство КГАСУ
420043, Казань, Зеленая, 1.